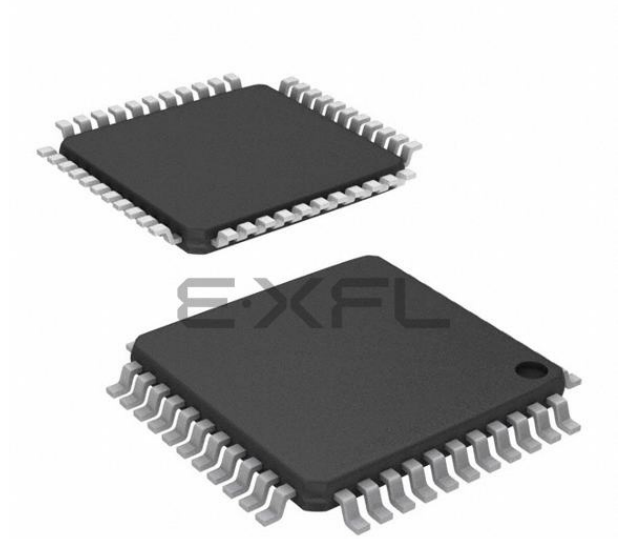


Welcome to E-XFL.COM

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	Not For New Designs
Core Processor	S08
Core Size	8-Bit
Speed	20MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	LVD, POR, PWM, WDT
Number of I/O	37
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 16x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LQFP
Supplier Device Package	44-LQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc9s08pt32vld

Chapter 12
FlexTimer Module (FTM)

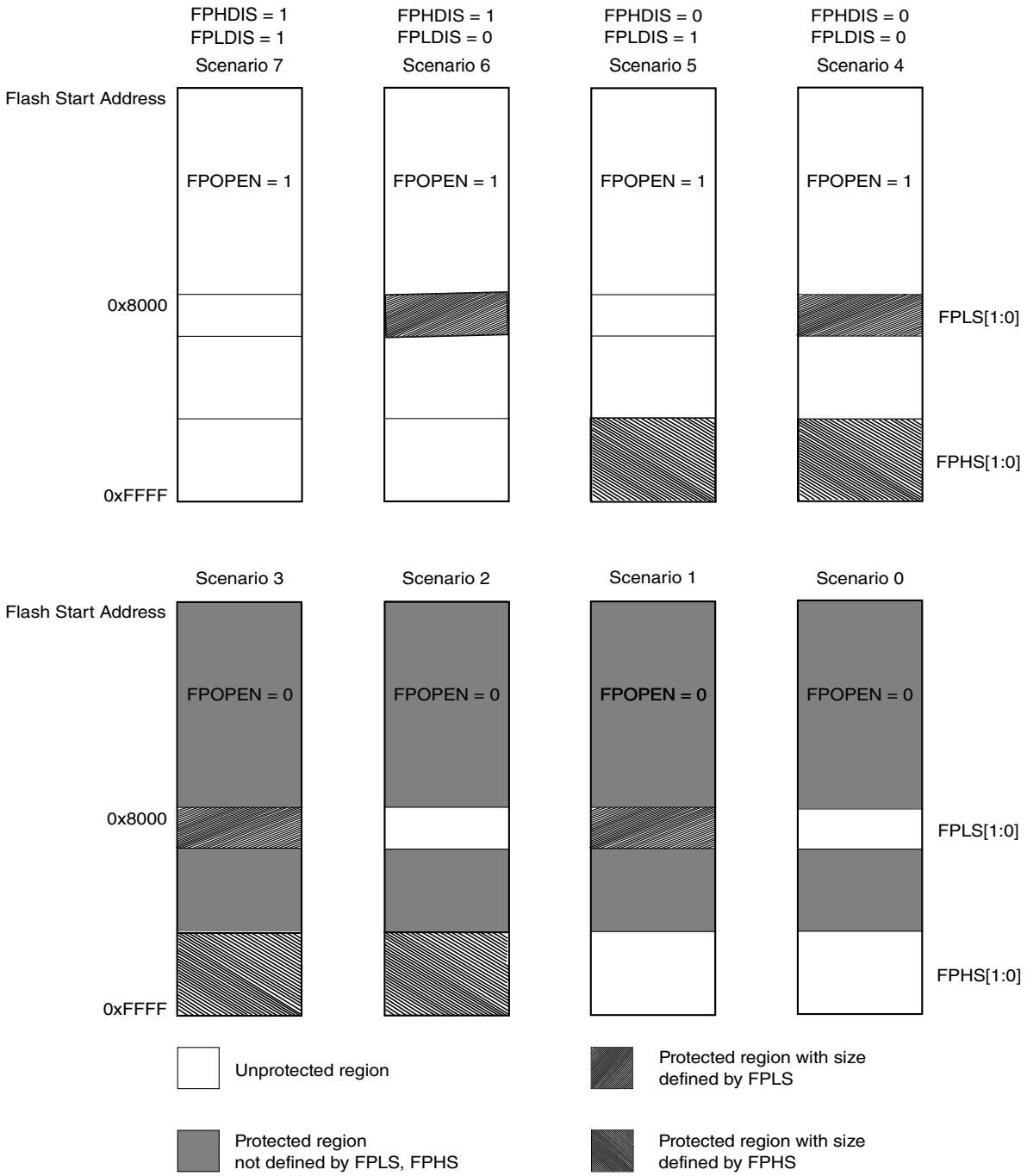


Figure 4-6. Flash protection scenarios

The general guideline is that flash protection can only be added and not removed. The following table specifies all valid transitions between flash protection scenarios. Any attempt to write an invalid scenario to the FPROT register will be ignored. The contents of the FPROT register reflect the active protection scenario. See the FPROT[FPHS] and FPROT[FPLS] bit descriptions for additional restrictions.

5.3.1 Interrupt Pin Request Status and Control Register (IRQ_SC)

This direct page register includes status and control bits, which are used to configure the IRQ function, report status, and acknowledge IRQ events.

Address: 3Bh base + 0h offset = 3Bh

Bit	7	6	5	4	3	2	1	0
Read	0	IRQPDD	IRQEDG	IRQPE	IRQF	0	IRQIE	IRQMOD
Write						IRQACK		
Reset	0	0	0	0	0	0	0	0

IRQ_SC field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 IRQPDD	Interrupt Request (IRQ) Pull Device Disable This read/write control bit is used to disable the internal pullup device when the IRQ pin is enabled (IRQPE = 1) allowing for an external device to be used. 0 IRQ pull device enabled if IRQPE = 1. 1 IRQ pull device disabled if IRQPE = 1.
5 IRQEDG	Interrupt Request (IRQ) Edge Select This read/write control bit is used to select the polarity of edges or levels on the IRQ pin that cause IRQF to be set. The IRQMOD control bit determines whether the IRQ pin is sensitive to both edges and levels or only edges. When the IRQ pin is enabled as the IRQ input and is configured to detect rising edges, the optional pullup resistor is disabled. 0 IRQ is falling edge or falling edge/low-level sensitive. 1 IRQ is rising edge or rising edge/high-level sensitive.
4 IRQPE	IRQ Pin Enable This read/write control bit enables the IRQ pin function. When this bit is set the IRQ pin can be used as an interrupt request. 0 IRQ pin function is disabled. 1 IRQ pin function is enabled.
3 IRQF	IRQ Flag This read-only status bit indicates when an interrupt request event has occurred. 0 No IRQ request. 1 IRQ event detected.
2 IRQACK	IRQ Acknowledge This write-only bit is used to acknowledge interrupt request events (write 1 to clear IRQF). Writing 0 has no meaning or effect. Reads always return 0. If edge-and-level detection is selected (IRQMOD = 1), IRQF cannot be cleared while the IRQ pin remains at its asserted level.

Table continues on the next page...

PORT_PTCD field descriptions (continued)

Field	Description
	Reset forces PTCD to all 0s, but these 0s are not driven out of the corresponding pins because reset also configures all port pins as high-impedance inputs with pullups disabled.

7.7.4 Port D Data Register (PORT_PTDD)

Address: 0h base + 3h offset = 3h

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

PORT_PTDD field descriptions

Field	Description
PTDD	Port D Data Register Bits For port D pins that are configured as inputs, a read returns the logic level on the pin. For port D pins that are configured as outputs, a read returns the last value that was written to this register. For port D pins that are configured as Hi-Z, a read returns uncertainty data. Writes are latched into all bits of this register. For port D pins that are configured as outputs, the logic level is driven out of the corresponding MCU pin. Reset forces PTDD to all 0s, but these 0s are not driven out of the corresponding pins because reset also configures all port pins as high-impedance inputs with pullups disabled.

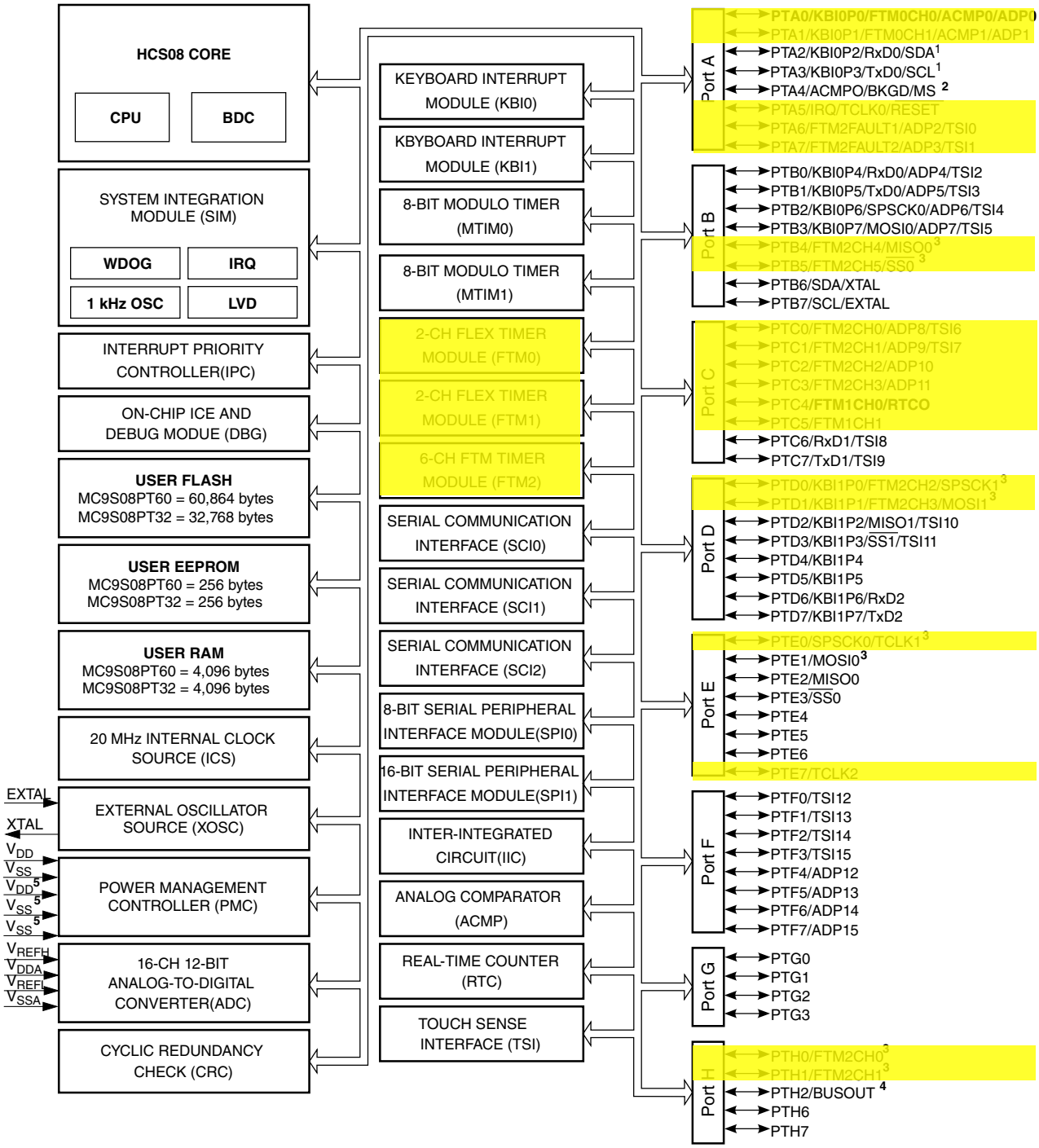
7.7.5 Port E Data Register (PORT_PTED)

Address: 0h base + 4h offset = 4h

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

PORT_PTED field descriptions

Field	Description
PTED	Port E Data Register Bits For port E pins that are configured as inputs, a read returns the logic level on the pin. For port E pins that are configured as outputs, a read returns the last value that was written to this register. For port E pins that are configured as Hi-Z, a read returns uncertainty data. Writes are latched into all bits of this register. For port E pins that are configured as outputs, the logic level is driven out of the corresponding MCU pin.



1. PTA2 and PTA3 operate as true open drain when working as output .
2. PTA4/ACMPO/BKGD/MS is an output-only pin when used as port pin.
3. PTB4, PTB5, PTD0, PTD1, PTE0, PTE1, PTH0, and PTH1 can provide ultra-high source/sink current up to 20 mA.
4. The frequency of the clock from BUSOUT must be equal or less than 10 MHz with 25 pF loading at PAD.
5. The secondary power pair of V_{DD} and V_{SS} (pin 41 and pin 40 in 64-pin packages) and the third V_{SS} (pin 13 in 64-pin packages) are not bonded in 32-pin packages.

Figure 9-3. Device block diagram highlighting FTM modules and pins

- When a BGND instruction is executed.
- When encountering a BDC breakpoint.

Background commands are of two types:

- Non-intrusive commands, defined as commands that can be issued while the user program is running. Non-intrusive commands can be issued through the BKGD pin while the MCU is in run mode; non-intrusive commands can also be executed when the MCU is in the active background mode. Non-intrusive commands include:
  - Memory access commands
  - Memory-access-with-status commands
  - BDC register access commands
  - The BACKGROUND command
- Active background commands, which can be executed only while the MCU is in active background mode. Active background commands include commands to:
  - Read or write CPU registers
  - Trace one user program instruction at a time
  - Leave active background mode to return to the user's application program (GO)

The active background mode is used to program a bootloader or user application program into the flash program memory before the MCU is operated in run mode for the first time. The active background mode can also be used to erase and reprogram the flash memory after it has been previously programmed.

10.4.4 Security mode

Usually HCS08 V6 MCUs are implemented with a secure operating mode. When in secure mode, external access to internal memory is restricted, so that only instructions fetched from secure memory can access secure memory.

The method by which the MCU is put into secure mode is not defined by the HCS08 V6 Core. The core receives an external input signal that, when asserted, informs to the core that the MCU is in secure mode.

While in secure mode, the core controls the following set of conditions:

1. Store the contents of PCL, PCH, X, A, and CCR on the stack, in that order.
2. Set the I bit in the CCR.
3. Fetch the high-order half of the interrupt vector.
4. Fetch the low-order half of the interrupt vector.
5. Delay for one free bus cycle.

Fetch three bytes of program information starting at the address indicated by the interrupt vector to fill the instruction queue in preparation for execution of the first instruction in the interrupt service routine.

After the CCR contents are pushed onto the stack, the I bit in the CCR is set to prevent other interrupts while in the interrupt service routine. Although it is possible to clear the I bit with an instruction in the interrupt service routine, this would allow nesting of interrupts (which is not recommended because it leads to programs that are difficult to debug and maintain).

For compatibility with the earlier M68HC05 MCUs, the high-order half of the H:X index register pair (H) is not saved on the stack as part of the interrupt sequence. The user must use a PSHH instruction at the beginning of the service routine to save H and then use a PULH instruction just before the RTI that ends the interrupt service routine. It is not necessary to save H if you are certain that the interrupt service routine does not use any instructions or auto-increment addressing modes that might change the value of H.

The software interrupt (SWI) instruction is like a hardware interrupt except that it is not masked by the global I bit in the CCR and it is associated with an instruction opcode within the program so it is not asynchronous to program execution.

10.7 Instruction Set Summary

Table 10-3. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
ADC #opr8i	Add with Carry	A ← (A) + (M) + (C)	↕	↕	–	↕	↕	↕	IMM	A9	ii	2
ADC opr8a			↕	↕	–	↕	↕	↕	DIR	B9	dd	3
ADC opr16a			↕	↕	–	↕	↕	↕	EXT	C9	hh ll	4
ADC oprx16,X			↕	↕	–	↕	↕	↕	IX2	D9	ee ff	4
ADC oprx8,X			↕	↕	–	↕	↕	↕	IX1	E9	ff	3
ADC ,X			↕	↕	–	↕	↕	↕	IX	F9		3
ADC oprx16,SP			↕	↕	–	↕	↕	↕	SP2	9ED9	ee ff	5
ADC oprx8,SP			↕	↕	–	↕	↕	↕	SP1	9EE9	ff	4

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
ADD #opr8i	Add without Carry	A ← (A) + (M)	↕	↕	–	↕	↕	↕	IMM	AB	ii	2
ADD opr8a			↕	↕	–	↕	↕	↕	DIR	BB	dd	3
ADD opr16a			↕	↕	–	↕	↕	↕	EXT	CB	hh ll	4
ADD oprx16,X			↕	↕	–	↕	↕	↕	IX2	DB	ee ff	4
ADD oprx8,X			↕	↕	–	↕	↕	↕	IX1	EB	ff	3
ADD ,X			↕	↕	–	↕	↕	↕	IX	FB		3
ADD oprx16,SP			↕	↕	–	↕	↕	↕	SP2	9EDB	ee ff	5
ADD oprx8,SP			↕	↕	–	↕	↕	↕	SP1	9EEB	ff	4
AIS #opr8i	Add Immediate Value (Signed) to Stack Pointer	SP ← (SP) + (M) where M is sign extended to a 16-bit value	–	–	–	–	–	–	IMM	A7	ii	2
AIX #opr8i	Add Immediate Value (Signed) to Index Register (H:X)	H:X ← (H:X) + (M) where M is sign extended to a 16-bit value	–	–	–	–	–	–	IMM	AF	ii	2
AND #opr8i	Logical AND	A ← (A) & (M)	0	–	–	↕	↕	–	IMM	A4	ii	2
AND opr8a			0	–	–	↕	↕	–	DIR	B4	dd	3
AND opr16a			0	–	–	↕	↕	–	EXT	C4	hh ll	4
AND oprx16,X			0	–	–	↕	↕	–	IX2	D4	ee ff	4
AND oprx8,X			0	–	–	↕	↕	–	IX1	E4	ff	3
AND ,X			0	–	–	↕	↕	–	IX	F4		3
AND oprx16,SP			0	–	–	↕	↕	–	SP2	9ED4	ee ff	5
AND oprx8,SP			0	–	–	↕	↕	–	SP1	9EE4	ff	4
ASL opr8a	Arithmetic Shift Left (same as LSL)	C ← MSB, LSB ← 0	↕	–	–	↕	↕	↕	DIR	38	dd	5
ASLA			↕	–	–	↕	↕	↕	INH	48		1
ASLX			↕	–	–	↕	↕	↕	INH	58		1
ASL oprx8,X			↕	–	–	↕	↕	↕	IX1	68	ff	5
ASL ,X			↕	–	–	↕	↕	↕	IX	78		4
ASL oprx8,SP			↕	–	–	↕	↕	↕	SP1	9E68	ff	6
ASR opr8a	Arithmetic Shift Right	MSB → MSB, LSB → C	↕	–	–	↕	↕	↕	DIR	37	dd	5
ASRA			↕	–	–	↕	↕	↕	INH	47		1
ASRX			↕	–	–	↕	↕	↕	INH	57		1
ASR oprx8,X			↕	–	–	↕	↕	↕	IX1	67	ff	5
ASR ,X			↕	–	–	↕	↕	↕	IX	77		4
ASR oprx8,SP			↕	–	–	↕	↕	↕	SP1	9E67	ff	6
BCC rel	Branch if Carry Bit Clear	Branch if (C) = 0	–	–	–	–	–	–	REL	24	rr	3
			–	–	–	–	–	–	DIR (b0)	11	dd	5

Table continues on the next page...

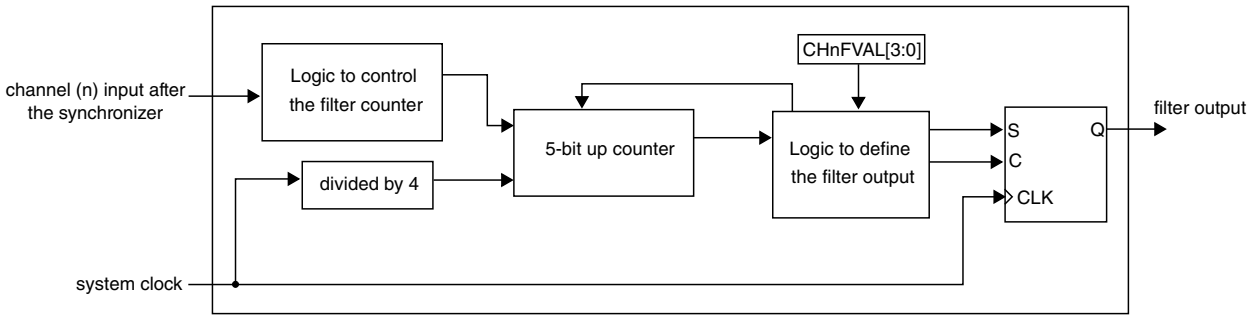


Figure 12-194. Channel input filter

If the opposite edge appears on the input signal before validation, the counter is reset. At the next input transition, the counter starts counting again. Any pulse shorter than the minimum valid width (CHnFVAL[3:0] bits × 4 system clocks) is regarded as a glitch and is not passed on to the edge detector. A timing diagram of the input filter is shown in the following figure.

The filter function is disabled when CHnFVAL[3:0] bits are zero. In this case, the input signal is delayed three rising edges of the system clock. If (CHnFVAL[3:0] ≠ 0000), then the input signal is delayed by the minimum pulse width (CHnFVAL[3:0] × 4 system clocks) plus a further four rising edges of the system clock (two rising edges to the synchronizer, one rising edge to the filter output plus one more to the edge detector). In other words, CHnF is set (4 + 4 × CHnFVAL[3:0]) system clock periods after a valid edge occurs on the channel input.

The clock for the 5-bit counter in the channel input filter is the system clock divided by 4.

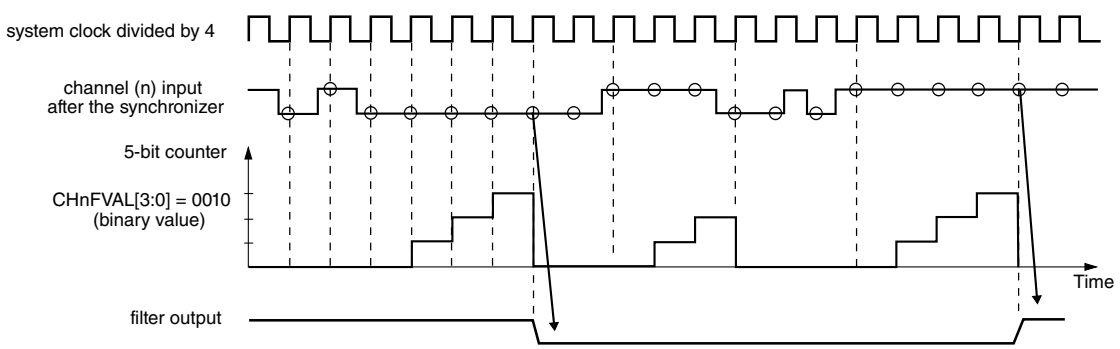


Figure 12-195. Channel input filter example

12.4.5 Output compare mode

The output compare mode is selected when (DECAPEN = 0), (COMBINE = 0), (CPWMS = 0) and (MSnB:MSnA = 0:1).

It is possible to use the output compare mode with (ELSnB:ELSnA = 0:0). In this case, when the counter reaches the value in the CnVH:CnVL registers, the CHnF bit is set and the channel (n) interrupt is generated, if CHnIE = 1. However, the channel (n) output is not modified and controlled by FTM.

Note

- Output compare mode is available only with (CNTINH:CNTINL = 0x0000).
- Output compare mode with (CNTINH:CNTINL ≠ 0x0000) is not recommended and its results are not guaranteed.

12.4.6 Edge-aligned PWM (EPWM) mode

The edge-aligned mode is selected when all of the following apply:

- (DECAPEN = 0)
- (COMBINE = 0)
- (CPWMS = 0)
- (MSnB = 1)

The EPWM period is determined by (MODH:L – CNTINH:L + 0x0001) and the pulse width (duty cycle) is determined by (CnVH:L – CNTINH:L).

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM counter = CnVH:L), that is, at the end of the pulse width.

This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an FTM.

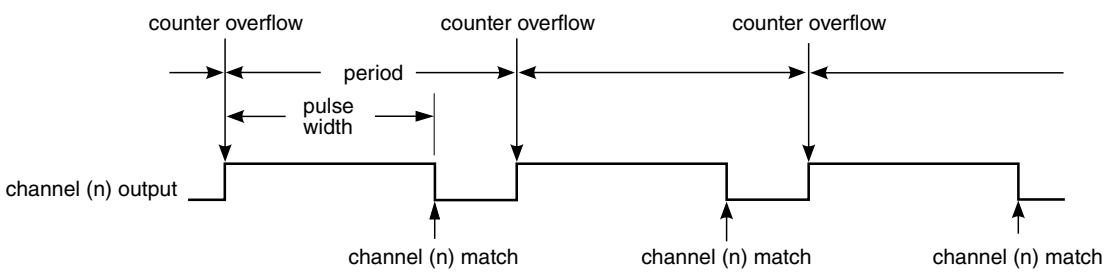


Figure 12-199. EPWM period and pulse width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:0) when the counter reaches the value in the CnVH:L registers, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1), however, the channel (n) output is not controlled by FTM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the counter overflow, when the value of CNTINH:L is loaded into the FTM counter. Additionally, it is forced low at the channel (n) match, when the FTM counter = CnVH:L. See the following figure.

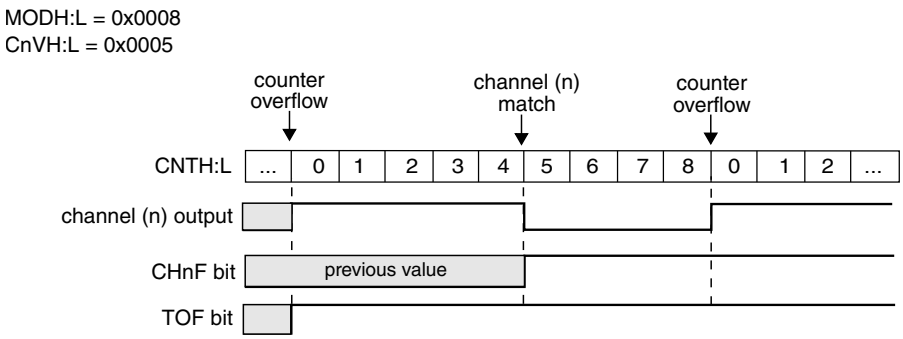


Figure 12-200. EPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the counter overflow, when the value of CNTINH:L is loaded into the FTM counter. Additionally, it is forced high at the channel (n) match, when the FTM counter = CnVH:L. See the following figure.

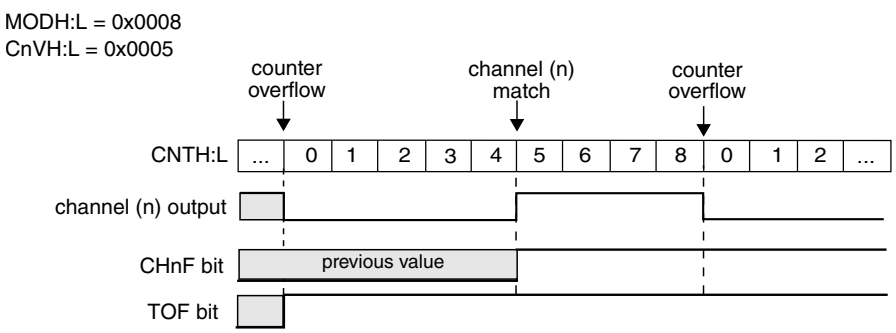


Figure 12-201. EPWM signal with ELSnB:ELSnA = X:1

If (CnVH:L = 0x0000), then the channel (n) output is a 0% duty cycle EPWM signal and CHnF bit is not set, even when there is the channel (n) match. If (CnVH:L > MODH:L), then the channel (n) output is a 100% duty cycle EPWM signal and CHnF bit is not set, even when there is the channel (n) match. Therefore, MODH:MODL must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

Note

- EPWM mode is available only with (CNTINH:L = 0x0000).
- EPWM mode with (CNTINH:L ≠ 0x0000) is not recommended and its results are not guaranteed.

13.6.2 MTIM Clock Configuration Register (MTIMx_CLK)

MTIM_CLK contains the clock select bits (CLKS) and the prescaler select bits (PS).

Address: Base address + 1h offset

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

MTIMx_CLK field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 CLKS	Clock Source Select These two read/write bits select one of four different clock sources as the input to the MTIM prescaler. Changing the clock source while the counter is active does not clear the counter. The count continues with the new clock source. Reset clears CLKS to 000b. 00 Encoding 0. Bus clock (BUSCLK). 01 Encoding 1. Fixed-frequency clock (XCLK). 10 Encoding 2. External source (TCLK pin), falling edge. 11 Encoding 3. External source (TCLK pin), rising edge.
PS	Clock Source Prescaler These four read/write bits select one of nine outputs from the 8-bit prescaler. Changing the prescaler value while the counter is active does not clear the counter. The count continues with the new prescaler value. Reset clears PS to 0000b. 0000 Encoding 0. MTIM clock source. 0001 Encoding 1. MTIM clock source/2. 0010 Encoding 2. MTIM clock source/4. 0011 Encoding 3. MTIM clock source/8. 0100 Encoding 4. MTIM clock source/16. 0101 Encoding 5. MTIM clock source/32. 0110 Encoding 6. MTIM clock source/64. 0111 Encoding 7. MTIM clock source/128. 1000 Encoding 8. MTIM clock source/256. Others Default to MTIM clock source/256.

- Loop mode
- Single-wire mode

15.1.3 Block diagram

The following figure shows the transmitter portion of the SCI.

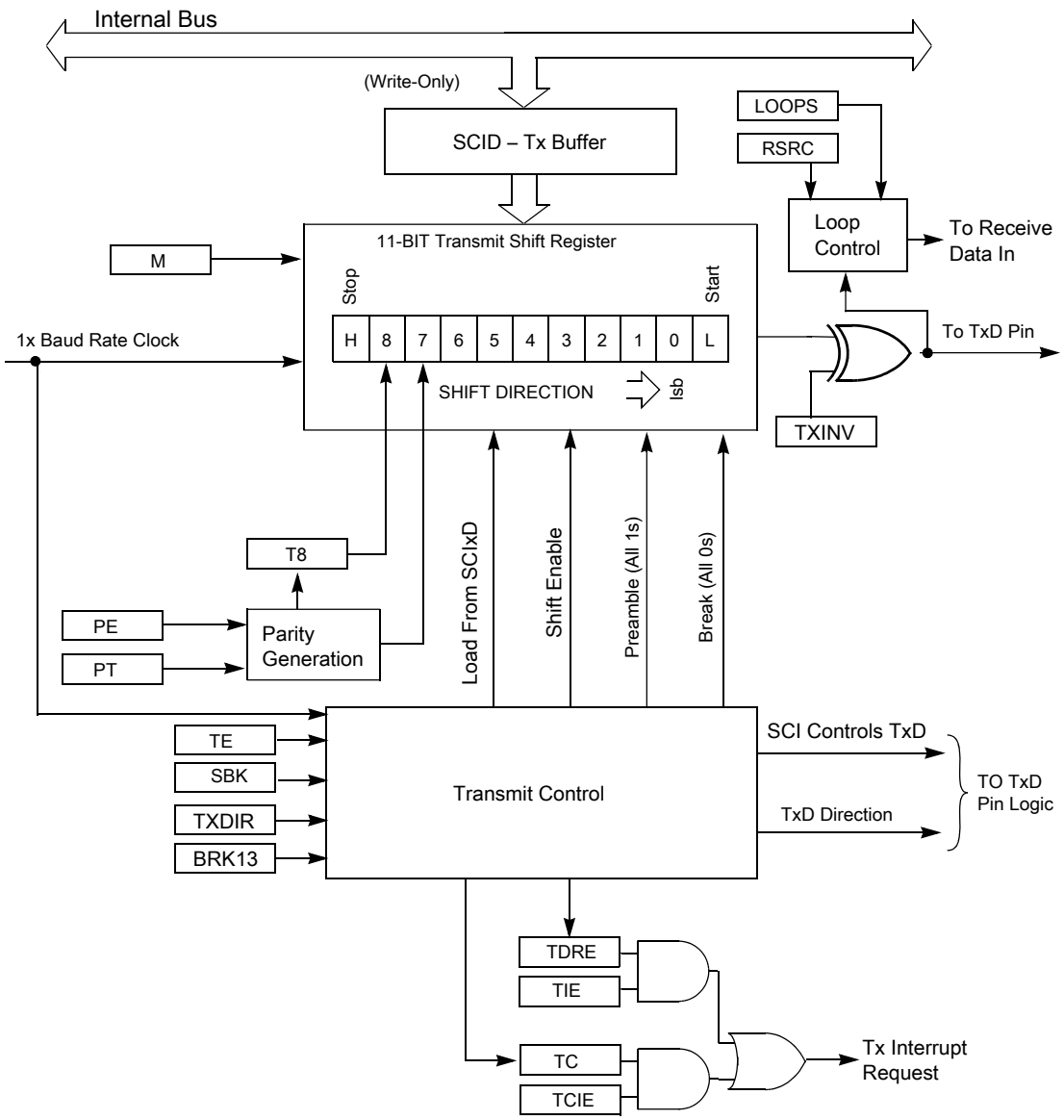


Figure 15-1. SCI transmitter block diagram

The following figure shows the receiver portion of the SCI.

The SS pin is the slave select input. Before a data transmission occurs, the SS pin of the slave SPI must be low. SS must remain low until the transmission is complete . If SS goes high, the SPI is forced into idle state.

The SS input also controls the serial data output pin, if SS is high (not selected), the serial data output pin is high impedance, and, if SS is low the first bit in the SPI Data Register is driven out of the serial data output pin . Also, if the slave is not selected (SS is high), then the SPSCK input is ignored and no internal shifting of the SPI shift register takes place.

Although the SPI is capable of duplex operation, some SPI peripherals are capable of only receiving SPI data in a slave mode. For these simpler devices, there is no serial data out pin.

Note

When peripherals with duplex capability are used, take care not to simultaneously enable two receivers whose serial outputs drive the same system slave's serial data output line.

As long as no more than one slave device drives the system slave's serial data output line, it is possible for several slaves to receive the same transmission from a master, although the master would not receive return information from all of the receiving slaves.

If the CPHA bit in SPI Control Register 1 is clear, odd numbered edges on the SPSCK input cause the data at the serial data input pin to be latched. Even numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on the LSBFE bit .

If the CPHA bit is set, even numbered edges on the SPSCK input cause the data at the serial data input pin to be latched. Odd numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on the LSBFE bit.

When CPHA is set, the first edge is used to get the first data bit onto the serial data output pin. When CPHA is clear and the SS input is low (slave selected), the first bit of the SPI data is driven out of the serial data output pin. After the eighth shift, the transfer is considered complete and the received data is transferred into the SPI data registers. To indicate transfer is complete, the SPRF flag in the SPI Status Register is set.

Note

A change of the bits BIDIROE with SPC0 set, CPOL, CPHA, SSOE, LSBFE, MODFEN, and SPC0 in slave mode will corrupt a transmission in progress and must be avoided.

16.5.2 Pseudo-Code Example

In this example, the SPI module will be set up for master mode with only hardware match interrupts enabled. The SPI will run at a maximum baud rate of bus clock divided by 2. Clock phase and polarity will be set for an active-high SPI clock where the first edge on SPSCK occurs at the start of the first cycle of a data transfer.

SPIx_C1=0x54(%01010100)				
Bit 7	SPIE	=	0	Disables receive and mode fault interrupts
Bit 6	SPE	=	1	Enables the SPI system
Bit 5	SPTIE	=	0	Disables SPI transmit interrupts
Bit 4	MSTR	=	1	Sets the SPI module as a master SPI device
Bit 3	CPOL	=	0	Configures SPI clock as active-high
Bit 2	CPHA	=	1	First edge on SPSCK at start of first data transfer cycle
Bit 1	SSOE	=	0	Determines SS pin function when mode fault enabled
Bit 0	LSBFE	=	0	SPI serial data transfers start with most significant bit

SPIx_C2 = 0x80(%10000000)				
Bit 7	SPMIE	=	1	SPI hardware match interrupt enabled
Bit 6		=	0	Unimplemented
Bit 5		=	0	Reserved
Bit 4	MODFEN	=	0	Disables mode fault function
Bit 3	BIDIROE	=	0	SPI data I/O pin acts as input
Bit 2		=	0	Reserved
Bit 1	SPISWAI	=	0	SPI clocks operate in wait mode
Bit 0	SPC0	=	0	uses separate pins for data input and output

SPIx_BR = 0x00(%00000000)				
Bit 7		=	0	Reserved
Bit 6:4		=	000	Sets prescale divisor to 1
Bit 3:0		=	0000	Sets baud rate divisor to 2

SPIx_S = 0x00(%00000000)				
Bit 7	SPRF	=	0	Flag is set when receive data buffer is full
Bit 6	SPMF	=	0	Flag is set when SPI_M = receive data buffer
Bit 5	SPTEF	=	0	Flag is set when transmit data buffer is empty
Bit 4	MODF	=	0	Mode fault flag for master mode
Bit 3:0		=	0	Reserved

SPI1_C2 field descriptions (continued)

Field	Description
	When the SPI is configured for slave mode, this bit has no meaning or effect. (The $\overline{SS}$ pin is the slave select input.) In master mode, this bit determines how the $\overline{SS}$ pin is used. For details, refer to the description of the SSOE bit in the C1 register. 0 Mode fault function disabled, master $\overline{SS}$ pin reverts to general-purpose I/O not controlled by SPI 1 Mode fault function enabled, master $\overline{SS}$ pin acts as the mode fault input or the slave select output
3 BIDIROE	Bidirectional Mode Output Enable When bidirectional mode is enabled because SPI pin control 0 (SPC0) is set to 1, BIDIROE determines whether the SPI data output driver is enabled to the single bidirectional SPI I/O pin. Depending on whether the SPI is configured as a master or a slave, it uses the MOSI (MOMI) or MISO (SISO) pin, respectively, as the single SPI data I/O pin. When SPC0 is 0, BIDIROE has no meaning or effect. 0 Output driver disabled so SPI data I/O pin acts as an input 1 SPI I/O pin enabled as an output
2 Reserved	This field is reserved. Do not write to this reserved bit.
1 SPISWAI	SPI Stop in Wait Mode This bit is used for power conservation while the device is in Wait mode. 0 SPI clocks continue to operate in Wait mode. 1 SPI clocks stop when the MCU enters Wait mode.
0 SPC0	SPI Pin Control 0 Enables bidirectional pin configurations. 0 SPI uses separate pins for data input and data output (pin mode is normal). In master mode of operation: MISO is master in and MOSI is master out. In slave mode of operation: MISO is slave out and MOSI is slave in. 1 SPI configured for single-wire bidirectional operation (pin mode is bidirectional). In master mode of operation: MISO is not used by SPI; MOSI is master in when BIDIROE is 0 or master I/O when BIDIROE is 1. In slave mode of operation: MISO is slave in when BIDIROE is 0 or slave I/O when BIDIROE is 1; MOSI is not used by SPI.

17.3.3 SPI Baud Rate Register (SPIx_BR)

Use this register to set the prescaler and bit rate divisor for an SPI master. This register may be read or written at any time.

Address: 30A0h base + 2h offset = 30A2h

Bit	7	6	5	4	3	2	1	0
Read	0	SPPR[2:0]			SPR[3:0]			
Write								
Reset	0	0	0	0	0	0	0	0

$\overline{SS}$ is the input or output for the master mode, and it is always the input for the slave mode.

The bidirectional mode does not affect SPSCK and $\overline{SS}$ functions.

Note

In bidirectional master mode, with the mode fault feature enabled, both data pins MISO and MOSI can be occupied by the SPI, though MOSI is normally used for transmissions in bidirectional mode and MISO is not used by the SPI. If a mode fault occurs, the SPI is automatically switched to slave mode. In this case, MISO becomes occupied by the SPI and MOSI is not used. Consider this scenario if the MISO pin is used for another purpose.

17.4.9 Error conditions

The SPI module has one error condition: the mode fault error.

17.4.9.1 Mode fault error

If the $\overline{SS}$ input becomes low while the SPI is configured as a master, it indicates a system error where more than one master may be trying to drive the MOSI and SPSCK lines simultaneously. This condition is not permitted in normal operation, and it sets the MODF bit in the SPI status register automatically provided that C2[MODFEN] is set.

In the special case where the SPI is in master mode and C2[MODFEN] is cleared, the $\overline{SS}$ pin is not used by the SPI. In this special case, the mode fault error function is inhibited and MODF remains cleared. If the SPI system is configured as a slave, the $\overline{SS}$ pin is a dedicated input pin. A mode fault error does not occur in slave mode.

If a mode fault error occurs, the SPI is switched to slave mode, with the exception that the slave output buffer is disabled. So the SPSCK, MISO and MOSI pins are forced to be high impedance inputs to avoid any possibility of conflict with another output driver. A transmission in progress is aborted and the SPI is forced into idle state.

If the mode fault error occurs in the bidirectional mode for an SPI system configured in master mode, the output enable of MOMI (MOSI in bidirectional mode) is cleared if it was set. No mode fault error occurs in the bidirectional mode for the SPI system configured in slave mode.

Note

Care must be taken when expecting data from a master while the slave is in a Wait mode or a Stop mode where the peripheral bus clock is stopped but internal logic states are retained. Even though the shift register continues to operate, the rest of the SPI is shut down (that is, an SPRF interrupt is not generated until an exit from Stop or Wait mode). Also, the data from the shift register is not copied into the SPIx_DH:SPIx_DL registers until after the slave SPI has exited Wait or Stop mode. An SPRF flag and SPIx_DH:SPIx_DL copy is only generated if Wait mode is entered or exited during a transmission. If the slave enters Wait mode in idle mode and exits Wait mode in idle mode, neither an SPRF nor a SPIx_DH:SPIx_DL copy occurs.

17.4.10.3 SPI in Stop mode

Operation in a Stop mode where the peripheral bus clock is stopped but internal logic states are retained depends on the SPI system. The Stop mode does not depend on C2[SPISWAI]. Upon entry to this type of stop mode, the SPI module clock is disabled (held high or low).

- If the SPI is in master mode and exchanging data when the CPU enters the Stop mode, the transmission is frozen until the CPU exits stop mode. After the exit from stop mode, data to and from the external SPI is exchanged correctly.
- In slave mode, the SPI remains synchronized with the master.

The SPI is completely disabled in a stop mode where the peripheral bus clock is stopped and internal logic states are not retained. After an exit from this type of stop mode, all registers are reset to their default values, and the SPI module must be reinitialized.

17.4.11 Reset

The reset values of registers and signals are described in the Memory Map and Register Descriptions content, which details the registers and their bitfields.

- If a data transmission occurs in slave mode after a reset without a write to SPIx_DH:SPIx_DL, the transmission consists of "garbage" or the data last received from the master before the reset.
- Reading from SPIx_DH:SPIx_DL after reset always returns zeros.

19.4.7 MCU wait mode operation

Wait mode is a low-power consumption standby mode from which recovery is fast because the clock sources remain active. If a conversion is in progress when the MCU enters wait mode, it continues until completion. Conversions can be initiated while the MCU is in wait mode by means of the hardware trigger or if continuous conversions are enabled.

The bus clock, bus clock divided by two, ALTCLK and ADACK are available as conversion clock sources while in wait mode.

ADC_SC1[COCO] is set by a conversion complete event that generates an ADC interrupt to wake the MCU from wait mode if the ADC interrupt is enabled (ADC_SC1[AIEN] = 1).

19.4.8 MCU Stop3 mode operation

Stop3 mode is a low-power consumption standby mode during which most or all clock sources on the MCU are disabled.

19.4.8.1 Stop3 mode with ADACK disabled

If the asynchronous clock, ADACK, is not selected as the conversion clock, executing a STOP instruction aborts the current conversion and places the ADC in its idle state. The contents of ADC_RH and ADC_RL are unaffected by Stop3 mode. After exiting from Stop3 mode, a software or hardware trigger is required to resume conversions.

19.4.8.2 Stop3 mode with ADACK enabled

If ADACK is selected as the conversion clock, the ADC continues operation during Stop3 mode. For guaranteed ADC operation, the MCU's voltage regulator must remain active during Stop3 mode. See the module introduction for configuration information for this MCU.

If a conversion is in progress when the MCU enters Stop3 mode, it continues until completion. Conversions can be initiated while the MCU is in Stop3 mode by means of the hardware trigger or if continuous conversions are enabled.

23.2 Memory map and register definition

WDOG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3030	Watchdog Control and Status Register 1 (WDOG_CS1)	8	R/W	80h	23.2.1/615
3031	Watchdog Control and Status Register 2 (WDOG_CS2)	8	R/W	01h	23.2.2/617
3032	Watchdog Counter Register: High (WDOG_CNTH)	8	R	00h	23.2.3/618
3033	Watchdog Counter Register: Low (WDOG_CNTL)	8	R	00h	23.2.4/618
3034	Watchdog Timeout Value Register: High (WDOG_TOVALH)	8	R/W	00h	23.2.5/619
3035	Watchdog Timeout Value Register: Low (WDOG_TOVALL)	8	R/W	04h	23.2.6/619
3036	Watchdog Window Register: High (WDOG_WINH)	8	R/W	00h	23.2.7/620
3037	Watchdog Window Register: Low (WDOG_WINL)	8	R/W	00h	23.2.8/620

23.2.1 Watchdog Control and Status Register 1 (WDOG_CS1)

This section describes the function of Watchdog Control and Status Register 1.

NOTE

TST is cleared (0:0) on POR only. Any other reset does not affect the value of this field.

Address: 3030h base + 0h offset = 3030h

Bit	7	6	5	4	3	2	1	0
Read Write	EN	INT	UPDATE	TST		DBG	WAIT	STOP
Reset	1	0	0	0	0	0	0	0

WDOG_CS1 field descriptions

Field	Description
7 EN	Watchdog Enable This write-once bit enables the watchdog counter to start counting. 0 Watchdog disabled. 1 Watchdog enabled.
6 INT	Watchdog Interrupt This write-once bit configures the watchdog to generate an interrupt request upon a reset-triggering event (timeout or illegal write to the watchdog), prior to forcing a reset. After the interrupt vector fetch, the reset occurs after a delay of 128 bus clocks.

Table continues on the next page...