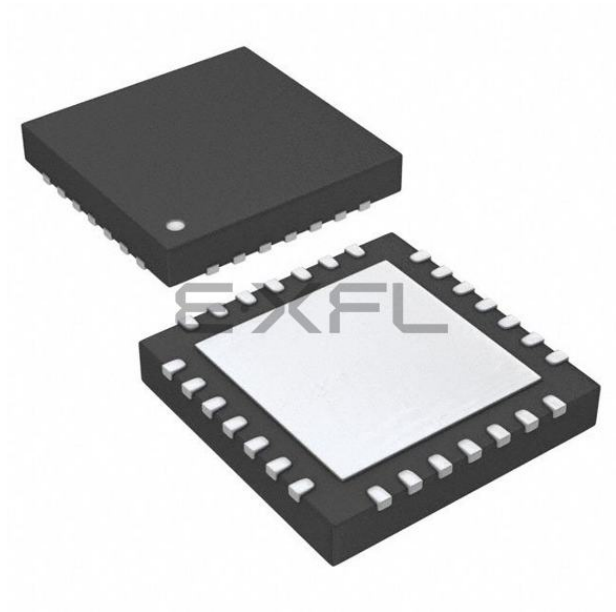


Welcome to E-XFL.COM

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	PIC
Core Size	8-Bit
Speed	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, HLVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 24x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VQFN Exposed Pad
Supplier Device Package	28-QFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f24k42-e-ml

TABLE 4-11: REGISTER FILE SUMMARY FOR PIC18(L)F24/25K42 DEVICES (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR
3C72h	CLC2GLS2	G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N	xxxxxxxx
3C71h	CLC2GLS1	G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N	xxxxxxxx
3C70h	CLC2GLS0	G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N	xxxxxxxx
3C6Fh	CLC2SEL3	D4S								xxxxxxxx
3C6Eh	CLC2SEL2	D3S								xxxxxxxx
3C6Dh	CLC2SEL1	D2S								xxxxxxxx
3C6Ch	CLC2SEL0	D1S								xxxxxxxx
3C6Bh	CLC2POL	POL	—	—	—	G4POL	G3POL	G2POL	G1POL	0---xxxx
3C6Ah	CLC2CON	EN	OE	OUT	INTP	INTN	MODE			00000000
3C69h	CLC3GLS3	G4D4T	G4D4N	G4D3T	G4D3N	G4D2T	G4D2N	G4D1T	G4D1N	xxxxxxxx
3C68h	CLC3GLS2	G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N	xxxxxxxx
3C67h	CLC3GLS1	G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N	xxxxxxxx
3C66h	CLC3GLS0	G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N	xxxxxxxx
3C65h	CLC3SEL3	D4S								xxxxxxxx
3C64h	CLC3SEL2	D3S								xxxxxxxx
3C63h	CLC3SEL1	D2S								xxxxxxxx
3C62h	CLC3SEL0	D1S								xxxxxxxx
3C61h	CLC3POL	POL	—	—	—	G4POL	G3POL	G2POL	G1POL	0---xxxx
3C60h	CLC3CON	EN	OE	OUT	INTP	INTN	MODE			00000000
3C5Fh	CLC4GLS3	G4D4T	G4D4N	G4D3T	G4D3N	G4D2T	G4D2N	G4D1T	G4D1N	xxxxxxxx
3C5Eh	CLC4GLS2	G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N	xxxxxxxx
3C5Dh	CLC4GLS1	G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N	xxxxxxxx
3C5Ch	CLC4GLS0	G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N	xxxxxxxx
3C5Bh	CLC4SEL3	D4S								xxxxxxxx
3C5Ah	CLC4SEL2	D3S								xxxxxxxx
3C59h	CLC4SEL1	D2S								xxxxxxxx
3C58h	CLC4SEL0	D1S								xxxxxxxx
3C57h	CLC4POL	POL	—	—	—	G4POL	G3POL	G2POL	G1POL	0---xxxx
3C56h	CLC4CON	EN	OE	OUT	INTP	INTN	MODE			00000000
3C55h - 3C00h	—	Unimplemented								—
3BFFh	DMA1SIRQ	SIRQ								00000000
3BFEh	DMA1AIRQ	AIRQ								00000000
3BFDh	DMA1CON1	EN	SIRQEN	DGO	—	—	AIRQEN	—	XIP	000--0-0
3BFCh	DMA1CON0	DMODE		DSTP	SMR		SMODE		SSTP	00000000
3BFBh	DMA1SSAU	—	—	SSA						--000000
3BFAh	DMA1SSAH	SSA								00000000
3BF9h	DMA1SSAL	SSA								00000000
3BF8h	DMA1SSZH	—	—	—	—	SSZ				----0000
3BF7h	DMA1SSZL	SSZ								00000000
3BF6h	DMA1SPTRU	—	—	SPTR						--000000
3BF5h	DMA1SPTRH	SPTR								00000000
3BF4h	DMA1SPTRL	SPTR								00000000
3BF3h	DMA1SCNTH	—	—	—	—	SCNT				----0000
3BF2h	DMA1SCNTL	SCNT								00000000
3BF1h	DMA1DSAH	DSA								00000000
3BF0h	DMA1DSAL	SSA								00000000
3BEFh	DMA1DSZH	—	—	—	—	DSZ				----0000
3BEEh	DMA1DSZL	DSZ								xxxxxxxx

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition
Note 1: Not present in LF devices.

REGISTER 11-4: PIR1: PERIPHERAL INTERRUPT Request Register 1

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
SMT1PWAIF	SMT1PRAIF	SMT1IF	C1IF	ADTIF	ADIF	ZCDIF	INT0IF
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	HS = Bit is set in hardware

bit 7 **SMT1PWAIF:** SMT1 Pulse Width Acquisition Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 6 **SMT1PRAIF:** SMT1 Period Acquisition Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 5 **SMT1IF:** SMT1 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 4 **C1IF:** CMP1 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 3 **ADTIF:** ADC Threshold Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 2 **ADIF:** ADC Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 1 **ZCDIF:** ZCD Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 0 **INT0IF:** External Interrupt 0 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

Note: Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit, or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

15.4 Register Definitions: Nonvolatile Memory

REGISTER 15-1: NVMCON1: NONVOLATILE MEMORY CONTROL 1 REGISTER

R/W-0/0	R/W-0/0	U-0	R/S/HC-0/0	R/W/HS-x/q	R/W-0/0	R/S/HC-0/0	R/S/HC-0/0
REG<1:0>		—	FREE	WRERR	WREN	WR	RD
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	HC = Bit is cleared by hardware
x = Bit is unknown	-n = Value at POR	S = Bit can be set by software, but not cleared
'0' = Bit is cleared	'1' = Bit is set	U = Unimplemented bit, read as '0'

bit 7-6 **REG<1:0>:** NVM Region Selection bit
10 =Access PFM Locations
x1 = Access User IDs, Configuration Bits, DIA, DCI, Rev ID and Device ID
00 = Access Data EEPROM Memory Locations

bit 5 **Unimplemented:** Read as '0'

bit 4 **FREE:** Program Flash Memory Erase Enable bit(1)
1 = Performs an erase operation on the next WR command
0 = The next WR command performs a write operation

bit 3 **WRERR:** Write-Reset Error Flag bit(2,3,4)
1 = A write operation was interrupted by a Reset (hardware set),
or WR was written to 1'b1 when an invalid address is accessed (Table 9-1, Table 15-1)
or WR was written to 1'b1 when REG<1:0> and address do not point to the same region
or WR was written to 1'b1 when a write-protected address is accessed (Table 9-2).
0 = All write operations have completed normally

bit 2 **WREN:** Program/Erase Enable bit
1 = Allows program/erase and refresh cycles
0 = Inhibits programming/erasing and user refresh of NVM

bit 1 **WR:** Write Control bit(5,6,7)
When REG points to a Data EEPROM Memory location:
1 = Initiates an erase/program cycle at the corresponding Data EEPROM Memory location
When REG points to a PFM location:
1 = Initiates the PFM write operation with data from the holding registers
0 = NVM program/erase operation is complete and inactive

bit 0 **RD:** Read Control bit(8)
1 = Initiates a read at address pointed by REG and NVMADR, and loads data into NVMDAT
0 = NVM read operation is complete and inactive

Note 1: This can only be used with PFM.
2: This bit is set when WR = 1 and clears when the internal programming timer expires or the write is completed successfully.
3: Bit must be cleared by the user; hardware will not clear this bit.
4: Bit may be written to '1' by the user in order to implement test sequences.
5: This bit can only be set by following the unlock sequence of **Section 15.1.4 "NVM Unlock Sequence"**.
6: Operations are self-timed and the WR bit is cleared by hardware when complete.
7: Once a write operation is initiated, setting this bit to zero will have no effect.
8: The bit can only be set in software. The bit is cleared by hardware when the operation is complete.

Depending on the priority of the DMA with respect to CPU execution (Refer to **Section 3.2 "Memory Access Scheme"** for more information), the DMA Controller can move data through two methods:

- Stalling the CPU execution until it has completed its transfers (DMA has higher priority over the CPU in this mode of operation)
- Utilizing unused CPU cycles for DMA transfers (CPU has higher priority over the DMA in this mode of operation). Unused CPU cycles are referred to as bubbles which are instruction cycles available for use by the DMA to perform read and write operations. In this way, the effective bandwidth for handling data is increased; at the same time, DMA operations can proceed without causing a processor stall.

17.4 DMA Interface

The DMA module transfers data from the source to the destination one byte at a time, this smallest data movement is called a DMA data transaction. A DMA Message refers to one or more DMA data transactions.

Each DMA data transaction consists of two separate actions:

- Reading the Source Address Memory and storing the value in the DMA Buffer register
- Writing the contents of the DMA Buffer register to the Destination Address Memory

Note:	DMA data movement is a two-cycle operation.

The XIP bit (DMAxCON0 register) is a status bit to indicate whether or not the data in the DMAxBUF register has been written to the destination address. If the bit is set then data is waiting to be written to the destination. If clear it means that either data has been written to the destination or that no source read has occurred.

The DMA has read access to PFM, Data EEPROM, and SFR/GPR space, and write access to SFR/GPR space. Based on these memory access capabilities, the DMA can support the following memory transactions:

TABLE 17-1: DMA MEMORY ACCESS

Read Source	Write Destination
Program Flash Memory	GPR
Program Flash Memory	SFR
Data EE	GPR
Data EE	SFR
GPR	GPR
SFR	GPR
GPR	SFR
SFR	SFR

Even though the DMA module has access to all memory and peripherals that are also available to the CPU, it is recommended that the DMA does not access any register that is part of the System arbitration. The DMA, as a system arbitration client should not be read or written by itself or by another DMA instantiation.

The following sections discuss the various control interfaces required for DMA data transfers.

17.4.1 DMA ADDRESSING

The start addresses for the source read and destination write operations are set using the DMAxSSA <21:0> and DMAxDSA <15:0> registers, respectively.

When the DMA Message transfers are in progress, the DMAxSPTR <21:0> and DMAxDPTR <15:0> registers contain the current address pointers for each source read and destination write operation, these registers are modified after each transaction based on the Address mode selection bits.

The SMODE and DMODE bits in the DMAxCON1 control register determine the address modes of operation by controlling how the DMAxSPTR <21:0> and DMAxDPTR <15:0> bits are updated after every DMA data transaction combination (Figure 17-2).

Each address can be separately configured to:

- Remain unchanged
- Increment by 1
- Decrement by 1

TABLE 20-1: IOC REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IOCAP	IOCAP7	IOCAP6	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0
IOCAN	IOCAN7	IOCAN6	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0
IOCAF	IOCAF7	IOCAF6	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0
IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0
IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0
IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0
IOCCP	IOCCP7	IOCCP6	IOCCP5	IOCCP4	IOCCP3	IOCCP2	IOCCP1	IOCCP0
IOCCN	IOCCN7	IOCCN6	IOCCN5	IOCCN4	IOCCN3	IOCCN2	IOCCN1	IOCCN0
IOCCF	IOCCF7	IOCCF6	IOCCF5	IOCCF4	IOCCF3	IOCCF2	IOCCF1	IOCCF0
IOCEP	—	—	—	—	IOCEP3(1)	—	—	—
IOCEN	—	—	—	—	IOCEN3(1)	—	—	—
IOCEF	—	—	—	—	IOCEF3(1)	—	—	—

Note 1: If MCLRE = 1 or LVP = 1, RE3 port functionality is disabled and IOC on RE3 is not available.

TABLE 20-2: SUMMARY OF REGISTERS ASSOCIATED WITH INTERRUPT-ON-CHANGE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
IOCxF	IOCxF7	IOCxF6	IOCxF5	IOCxF4	IOCxF3	IOCxF2	IOCxF1	IOCxF0	290
IOCxN	IOCxN7	IOCxN6	IOCxN5	IOCxN4	IOCxN3	IOCxN2	IOCxN1	IOCxN0	290
IOCxP	IOCxP7	IOCxP6	IOCxP5	IOCxP4	IOCxP3	IOCxP2	IOCxP1	IOCxP0	290

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by interrupt-on-change.

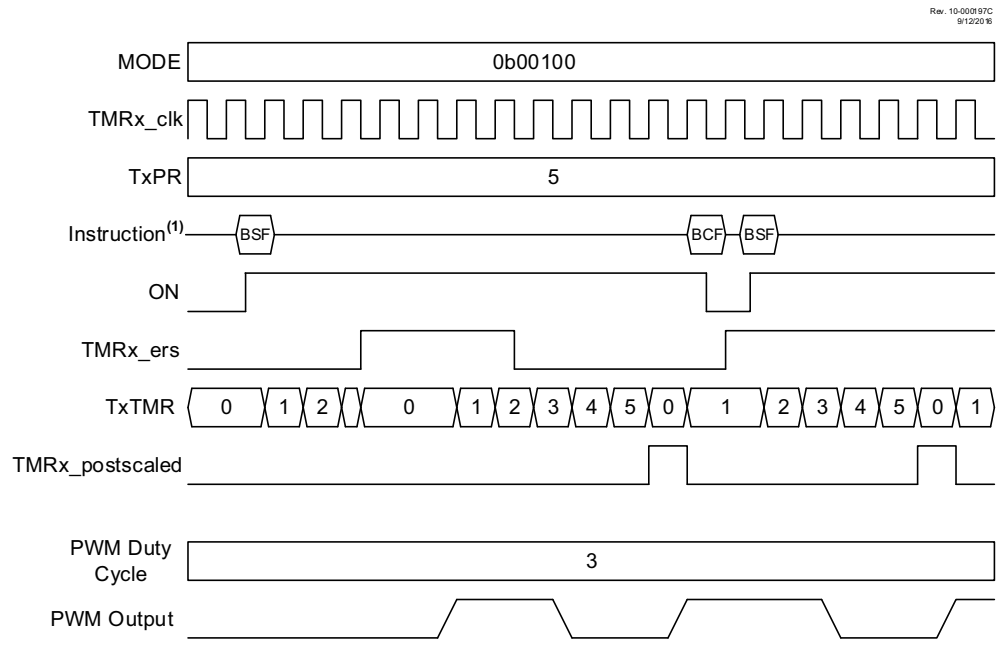
24.5.3 EDGE-TRIGGERED HARDWARE LIMIT MODE

In Hardware Limit mode the timer can be reset by the TMRx_ers external signal before the timer reaches the period count. Three types of Resets are possible:

- Reset on rising or falling edge (MODE<4:0> = 00011)
- Reset on rising edge (MODE<4:0> = 0010)
- Reset on falling edge (MODE<4:0> = 00101)

When the timer is used in conjunction with the CCP in PWM mode then an early Reset shortens the period and restarts the PWM pulse after a two clock delay. Refer to Figure 24-6.

FIGURE 24-6: EDGE TRIGGERED HARDWARE LIMIT MODE TIMING DIAGRAM (MODE=00100)



Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

REGISTER 24-6: TxHLT: TIMERx HARDWARE LIMIT CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
PSYNC	CKPOL	CKSYNC	MODE<4:0>				
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7 **PSYNC:** Timerx Prescaler Synchronization Enable bit(1, 2)
1 = TxTMR Prescaler Output is synchronized to Fosc/4
0 = TxTMR Prescaler Output is not synchronized to Fosc/4

bit 6 **CKPOL:** Timerx Clock Polarity Selection bit(3)
1 = Falling edge of input clock clocks timer/prescaler
0 = Rising edge of input clock clocks timer/prescaler

bit 5 **CKSYNC:** Timerx Clock Synchronization Enable bit(4, 5)
1 = ON register bit is synchronized to T2TMR_clk input
0 = ON register bit is not synchronized to T2TMR_clk input

bit 4-0 **MODE<4:0>:** Timerx Control Mode Selection bits(6, 7)
See Table 24-1 for all operating modes.

Note 1: Setting this bit ensures that reading TxTMR will return a valid data value.
2: When this bit is '1', Timer2 cannot operate in Sleep mode.
3: CKPOL should not be changed while ON = 1.
4: Setting this bit ensures glitch-free operation when the ON is enabled or disabled.
5: When this bit is set then the timer operation will be delayed by two TxTMR input clocks after the ON bit is set.
6: Unless otherwise indicated, all modes start upon ON = 1 and stop upon ON = 0 (stops occur without affecting the value of TxTMR).
7: When TxTMR = TxPR, the next clock clears TxTMR, regardless of the operating mode.

REGISTER 28-7: CWGxAS1: CWG AUTO-SHUTDOWN CONTROL REGISTER 1

U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	AS6E	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	q = Value depends on condition

bit 7 **Unimplemented** Read as '0'

bit 6 **AS6E:** CWG Auto-shutdown Source 6 Enable bit

1 = Auto-shutdown for Source 6 is enabled

CWG Module	CWG1	CWG2	CWG3
Auto-shutdown Source 6	CLC2 OUT	CLC3 OUT	CLC4 OUT

0 = Auto-shutdown for Source 6 is disabled

bit 5 **AS5E:** CWG Auto-shutdown Source 5 (CMP2 OUT) Enable bit

1 = Auto-shutdown for CMP2 OUT is enabled

0 = Auto-shutdown for CMP2 OUT is disabled

bit 4 **AS4E:** CWG Auto-shutdown Source 4 (CMP1 OUT) Enable bit

1 = Auto-shutdown for CMP1 OUT is enabled

0 = Auto-shutdown for CMP1 OUT is disabled

bit 3 **AS3E:** CWG Auto-shutdown Source 3 (TMR6_Postscaled) Enable bit

1 = Auto-shutdown for TMR6_Postscaled is enabled

0 = Auto-shutdown for TMR6_Postscaled is disabled

bit 2 **AS2E:** CWG Auto-shutdown Source 2 (TMR4_Postscaled) Enable bit

1 = Auto-shutdown for TMR4_Postscaled is enabled

0 = Auto-shutdown for TMR4_Postscaled is disabled

bit 1 **AS1E:** CWG Auto-shutdown Source 1 (TMR2_Postscaled) Enable bit

1 = Auto-shutdown for TMR2_Postscaled is enabled

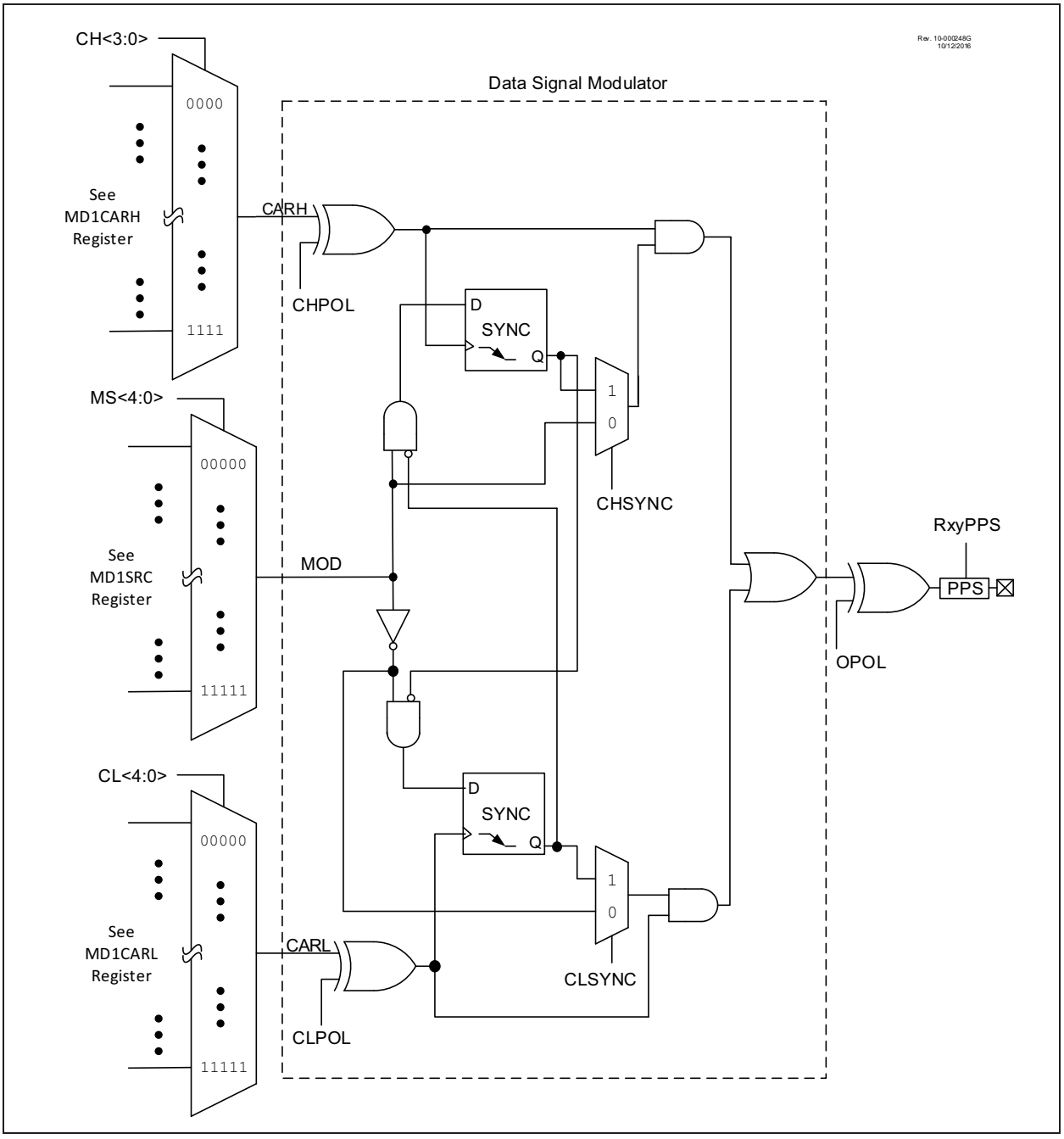
0 = Auto-shutdown for TMR2_Postscaled is disabled

bit 0 **AS0E:** CWG Auto-shutdown Source 0 (Pin selected by CWGxPPS) Enable bit

1 = Auto-shutdown for CWGxPPS Pin is enabled

0 = Auto-shutdown for CWGxPPS Pin is disabled

FIGURE 32-1: SIMPLIFIED BLOCK DIAGRAM OF THE DATA SIGNAL MODULATOR



REGISTER 33-14: UxP2H: UART PARAMETER 2 HIGH REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	P2<8>
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-6 **Unimplemented:** Read as ‘0’

bit 0 **P2<8>:** Most Significant Bit of Parameter 2
DMX mode:
Most Significant bit of first address of receive block
DALI mode:
Most Significant bit of number of half-bit periods of idle time in Forward Frame detection threshold
Other modes:
Not used

REGISTER 33-15: UxP2L: UART PARAMETER 2 LOW REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
P2<7:0>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

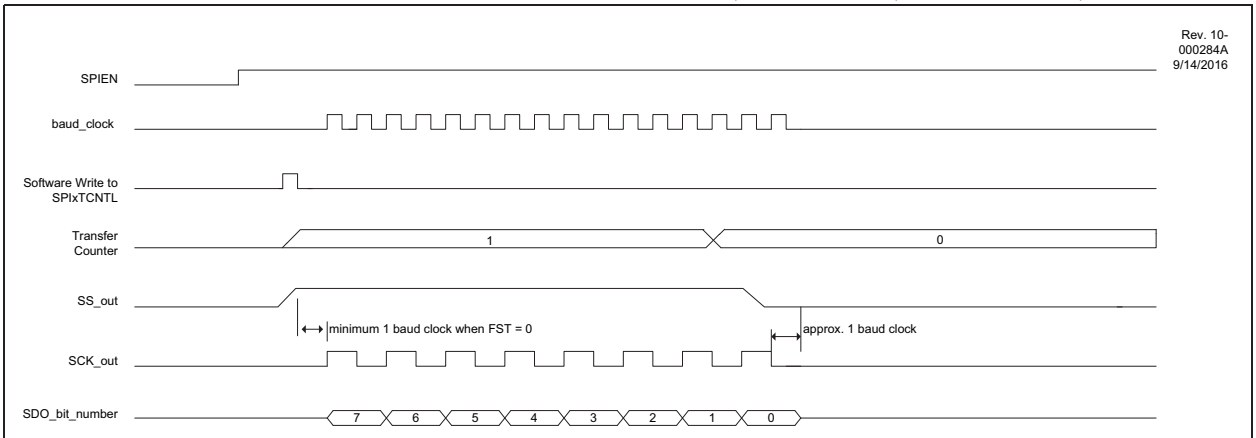
bit 7-0 **P2<7:0>:** Least Significant Bits of Parameter 2
DMX mode:
Least Significant Byte of first address of receive block
LIN Slave mode:
Number of data bytes to transmit
DALI mode:
Least Significant Byte of number of half-bit periods of idle time in Forward Frame detection threshold
Asynchronous Address mode:
Receiver address
Other modes:
Not used

34.5.5 MASTER MODE SLAVE SELECT CONTROL

34.5.5.1 Hardware Slave Select Control

This SPI module allows for direct hardware control of a Slave Select output. The Slave Select output SS(out) is controlled both directly, through the SSET bit of SPIxCON2, as well indirectly by the hardware while the transfer counter is non-zero (see **Section 34.4 "Transfer Counter"**). SS(out) is steered by the PPS registers to pins (see **Section 19.2 "PPS Outputs"**) and its polarity is controlled by the SSP bit of SPIxCON1. Setting the SSET bit will also assert SS(out). Clearing the SSET bit will leave SS(out) to be controlled by the Transfer Counter. When the Transfer Counter is loaded, the SPI module will automatically assert the SS. When the Transfer Counter decrements to zero, the SPI module will deassert SS either one baud period after the final SCK pulse of the final transfer (if CKE/SMP = 0/1) or one half baud period otherwise (see Figure 34-6).

FIGURE 34-6: SPI MASTER SS OPERATION- CKE = 0, BMODE = 1, TCWIDTH = 0, SSP = 0



Note: 1. *SDO bit number* illustrates the transmitted bit number, and is not intended to imply SDO (out) tristate operation.
2. Assumes SPIxTXB holds data when SPIxTCNTL is written.

34.5.5.2 Software Slave Select Control

Slave Select can also be controlled through software via a general purpose I/O pin. In this case, ensure that the pin in question is configured as a GPIO through PPS (see **Section 19.2 "PPS Outputs"**), and ensure that the pin is set as an output (clear the appropriate bit in the appropriate TRIS register). In this case, SSET will not affect the slave select, the Transfer Counter will not automatically control the slave select output, and all setting and clearing of the slave select output line must be directly controlled by software.

FIGURE 34-12: TRADITIONAL SPI DAISY-CHAIN CONNECTION

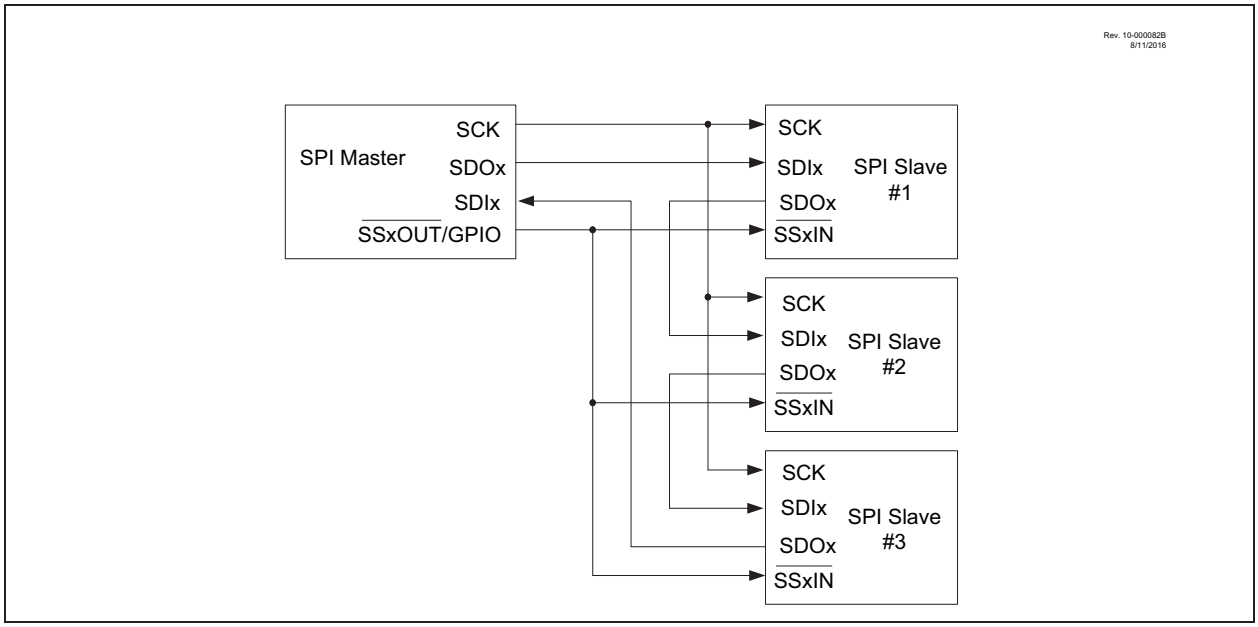
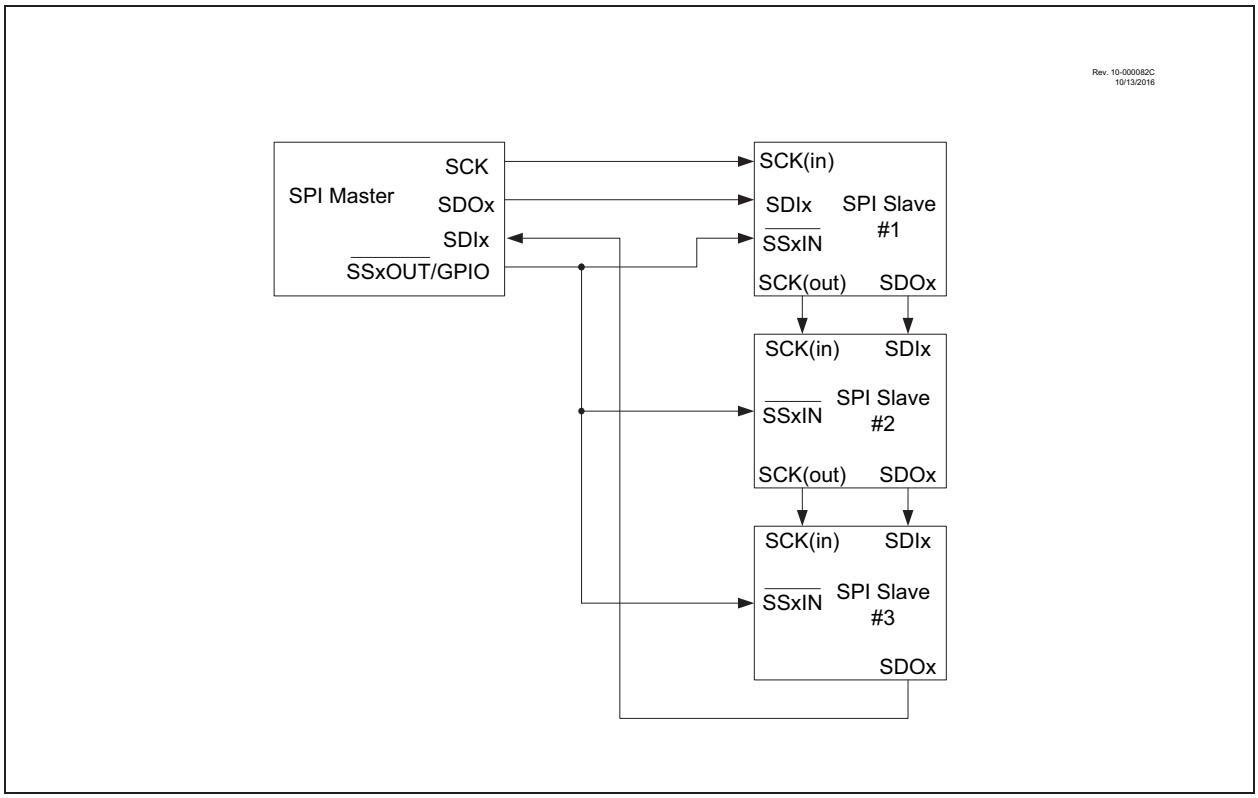


FIGURE 34-13: SPI DAISY-CHAIN CONNECTION WITH CHAINED SCK



35.5.10 MASTER RECEPTION IN 7-BIT ADDRESSING MODE

This section describes the sequence of events for the I^2C module configured as an I^2C master in 7-bit Addressing mode and is receiving data. Figure 35-20 is used as a visual reference for this description.

1. Master software loads slave address in I2CxADB1 with R/W bit = d and number of bytes to be received in one sequence in I2CxCNT register.
2. Master hardware waits for BFRE bit to be set; then shifts out start and address with R/W = 1.
3. Master sends out the 9th SCL pulse for ACK, master hardware clocks in ACK from Slave
4. If ABD = 0; i.e. Address buffers are enabled

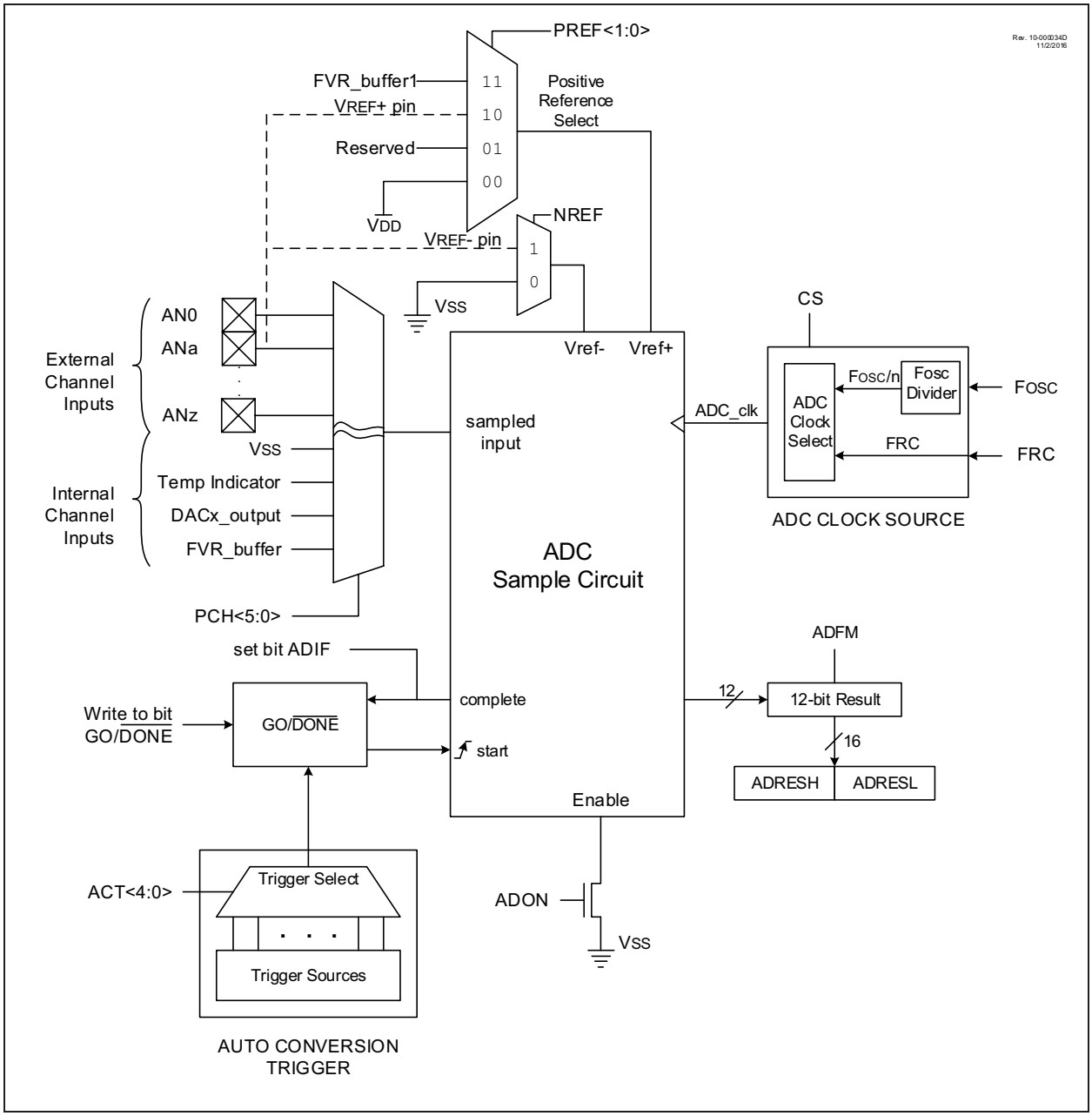
If NACK, master hardware sends Stop or sets MDR (if RSEN = 1) and waits for user software to write to S bit for restart.

If ABD = 1; i.e. Address buffers are disabled

If NACK, master hardware sends Stop or sets MDR (if RSEN = 1) and waits for user software to load the new address into I2CxTXB. Software writes to the S bit are ignored in this case.

5. If ACK, master hardware receives 7-bits of data into the shift register.
6. If the receive buffer is full (i.e. RXBF = 1), clock is stretched on 7th falling SCL edge.
7. Master software must read previous data out of I2CxRXB to clear RXBF.
8. Master hardware receives 8th bit of data into the shift register and loads it into I2CxRXB, sets I2CxRXIF and RXBF bits. I2CxCNT is decremented.
9. If I2CxCNT! = 0, master hardware clocks out ACKDT as ACK value to slave. If I2CxCNT = 0, master hardware clocks out ACKCNT as ACK value to slave. It is up to the user to set the values of ACKDT and ACKCNT correctly. If the user does not set ACKCNT to '1', the master hardware will never send a NACK when I2CxCNT becomes zero. Since a NACK was not seen on the bus, the master hardware will also not assert a Stop condition.
10. Go to step 4.

FIGURE 38-1: ADC^2 BLOCK DIAGRAM



Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ADLTHH	LTH<15:8>								637
ADLTHL	LTH<7:0>								638
ADUTHH	UTH<15:8>								638
ADUTHL	UTH<7:0>								638
ADERRL	ERR<15:8>								637
ADACT	—	—	—	—	ADACT<4:0>				623
ADCP	CPON	—	—	—	—	—	—	CPRDY	640

Legend: – = unimplemented read as '0'. Shaded cells are not used for the ADC module.

40.0 COMPARATOR MODULE

Note:	The PIC18LF24/25K42 devices have two comparators. Therefore, all information in this section refers to both C1 and C2.

Comparators are used to interface analog circuits to a digital circuit by comparing two analog voltages and providing a digital indication of their relative magnitudes. Comparators are very useful mixed signal building blocks because they provide analog functionality independent of program execution.

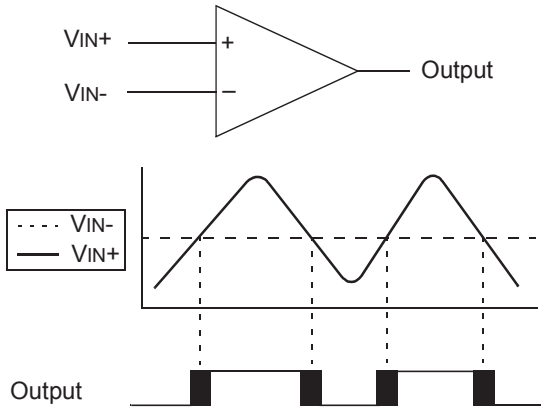
The analog comparator module includes the following features:

- Programmable input selection
- Programmable output polarity
- Rising/falling output edge interrupts

40.1 Comparator Overview

A single comparator is shown in Figure 40-1 along with the relationship between the analog input levels and the digital output. When the analog voltage at VIN+ is less than the analog voltage at VIN-, the output of the comparator is a digital low level. When the analog voltage at VIN+ is greater than the analog voltage at VIN-, the output of the comparator is a digital high level.

FIGURE 40-1: SINGLE COMPARATOR



Note: The black areas of the output of the comparator represents the uncertainty due to input offsets and response time.

BZ **Branch if Zero**

Syntax: BZ n

Operands: -128 ≤ n ≤ 127

Operation: if ZERO bit is '1'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0000	nnnn	nnnn

Description: If the ZERO bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BZ Jump`

Before Instruction
PC = address `(HERE)`
After Instruction
If ZERO = 1;
PC = address `(Jump)`
If ZERO = 0;
PC = address `(HERE + 2)`

CALL **Subroutine Call**

Syntax: CALL k {,s}

Operands: 0 ≤ k ≤ 1048575
s ∈ [0,1]

Operation: (PC) + 4 → TOS,
k → PC<20:1>,
if s = 1
(W) → WS,
(Status) → STATUSS,
(BSR) → BSRS

Status Affected: None

Encoding:

1st word (k<7:0>)	1110	110s	$k_7$kkk	kkkk$_0$
2nd word(k<19:8>)	1111	k_{19}kkk	kkkk	kkkk$_8$

Description: Subroutine call of entire 2-Mbyte memory range. First, return address (PC + 4) is pushed onto the return stack. If 's' = 1, the W, Status and BSR registers are also pushed into their respective shadow registers, WS, STATUSS and BSRS. If 's' = 0, no update occurs (default). Then, the 20-bit value 'k' is loaded into PC<20:1>. CALL is a 2-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	PUSH PC to stack	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: `HERE CALL THERE, 1`

Before Instruction
PC = address `(HERE)`
After Instruction
PC = address `(THERE)`
TOS = address `(HERE + 4)`
WS = W
BSRS = BSR
STATUSS = Status

CPFSLT **Compare f with W, skip if f < W**

Syntax: CPFSLT f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) < (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	000a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.
If the contents of 'f' are less than the contents of W, then the fetched instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction.
If 'a' is '0', the Access Bank is selected.
If 'a' is '1', the BSR is used to select the GPR bank.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    CPFSLT REG, 1
NLESS   :
LESS    :
```

Before Instruction
PC = Address (HERE)
W = ?
After Instruction
If REG < W;
PC = Address (LESS)
If REG ≥ W;
PC = Address (NLESS)

DAW **Decimal Adjust W Register**

Syntax: DAW

Operands: None

Operation: If [W<3:0> > 9] or [DC = 1] then
(W<3:0>) + 6 → W<3:0>;
else
(W<3:0>) → W<3:0>;

If [W<7:4> + DC > 9] or [C = 1] then
(W<7:4>) + 6 + DC → W<7:4> ;
else
(W<7:4>) + DC → W<7:4>

Status Affected: C

Encoding:

0000	0000	0000	0111

Description: DAW adjusts the 8-bit value in W, resulting from the earlier addition of two variables (each in packed BCD format) and produces a correct packed BCD result.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register W	Process Data	Write W

Example1:
```
DAW
```
Before Instruction
W = A5h
C = 0
DC = 0
After Instruction
W = 05h
C = 1
DC = 0

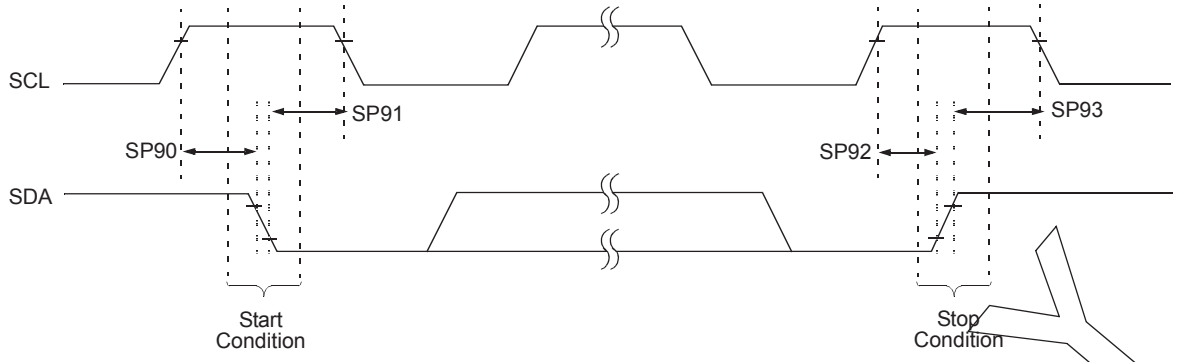
Example 2:
Before Instruction
W = CEh
C = 0
DC = 0
After Instruction
W = 34h
C = 1
DC = 0

TABLE 44-1: REGISTER FILE SUMMARY FOR PIC18(L)F24/25K42 DEVICES (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
3D59h	I2C2ADR0	ADR								591
3D58h	I2C2ADB1	ADB								596
3D57h	I2C2ADB0	ADB								595
3D56h	I2C2CNT	CNT								588
3D55h	I2C2TXB	TXB								
3D54h	I2C2RXB	RXB								
3D53h - 3D1Dh	—	Unimplemented								
3D1Ch	SPI1CLK	CLKSEL								544
3D1Bh	SPI1INTE	SRMTIE	TCZIE	SOSIE	EOSIE	—	RXOIE	TXUIE	—	538
3D1Ah	SPI1INTF	SRMTIF	TCZIF	SOSIF	EOSIF	—	RXOIF	TXUIF	—	537
3D19h	SPI1BAUD	BAUD								540
3D18h	SPI1TWIDTH	—	—	—	—	—	TWIDTH			539
3D17h	SPI1STATUS	TXWE	—	TXBE	—	RXRE	CLRBF	—	RXBF	543
3D16h	SPI1CON2	BUSY	SSFLT	—	—	—	SSET	TXR	RXR	542
3D15h	SPI1CON1	SMP	CKE	CKP	FST	—	SSP	SDIP	SDOP	541
3D14h	SPI1CON0	EN	—	—	—	—	LSBF	MST	BMODE	540
3D13h	SPI1TCNTH	—	—	—	—	—	TCNTH			539
3D12h	SPI1TCNTL	TCNTL								538
3D11h	SPI1TXB	TXB								544
3D10h	SPI1RXB	RXB								543
3D0Fh - 3CFFh	—	Unimplemented								
3CFEh	MD1CARH	—	—	—	CH					474
3CFDh	MD1CARL	—	—	—	CL					474
3CFCh	MD1SRC	—	—	—	MS					475
3CFBh	MD1CON1	—	—	CHPOL	CHSYNC	—	—	CLPOL	CLSYNC	473
3CFAh	MD1CON0	EN	—	OUT	OPOL	—	—	—	BIT	472
3CF9h - 3CE7h	—	Unimplemented								
3CE6h	CLKRCON	EN	—	—	DC		DIV			118
3CE5h	CLKRCLK	—	—	—	—	CLK				119
3CE4h - 3C7Fh	—	Unimplemented								
3C7Eh	CLCDATA0	—	—	—	—	CLC4OUT	CLC3OUT	CLC2OUT	CLC1OUT	450
3C7Dh	CLC1GLS3	G4D4T	G4D4N	G4D3T	G4D3N	G4D2T	G4D2N	G4D1T	G4D1N	449
3C7Ch	CLC1GLS2	G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N	448
3C7Bh	CLC1GLS1	G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N	447
3C7Ah	CLC1GLS0	G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N	446
3C79h	CLC1SEL3	D4S								445
3C78h	CLC1SEL2	D3S								445
3C77h	CLC1SEL1	D2S								445
3C76h	CLC1SEL0	D1S								445
3C75h	CLC1POL	POL	—	—	—	G4POL	G3POL	G2POL	G1POL	444
3C74h	CLC1CON	EN	OE	OUT	INTP	INTN	MODE			443
3C73h	CLC2GLS3	G4D4T	G4D4N	G4D3T	G4D3N	G4D2T	G4D2N	G4D1T	G4D1N	449
3C72h	CLC2GLS2	G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N	448
3C71h	CLC2GLS1	G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N	447
3C70h	CLC2GLS0	G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N	446
3C6Fh	CLC2SEL3	D4S								445

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition
Note 1: Not present in LF devices.

FIGURE 46-18: I²C BUS START/STOP BITS TIMING



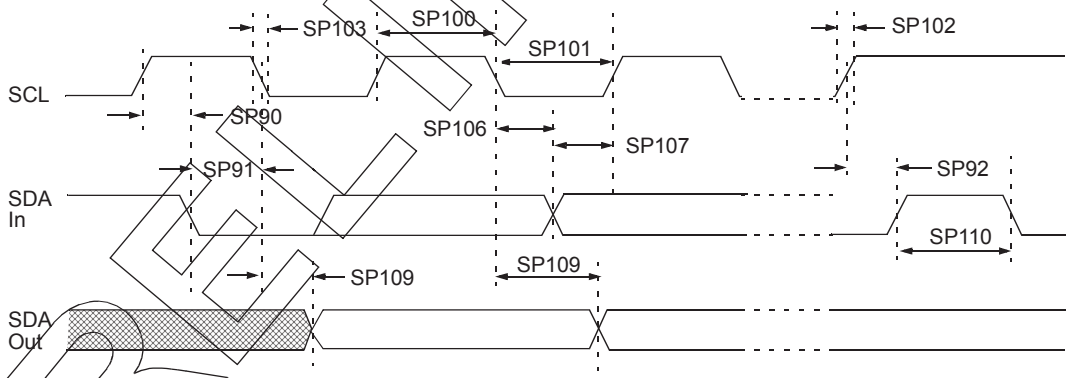
Note: Refer to Figure 46-4 for load conditions.

TABLE 46-22: I²C BUS START/STOP BITS REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)								
Param No.	**Symbol**	**Characteristic**		**Min.**	**Typ**	**Max.**	**Units**	**Conditions**
SP90*	TSU:STA	Start condition Setup time	100 kHz mode	4700	—	—	ns	Only relevant for Repeated Start condition
			400 kHz mode	600	—	—		
SP91*	THD:STA	Start condition Hold time	100 kHz mode	4000	—	—	ns	After this period, the first clock pulse is generated
			400 kHz mode	600	—	—		
SP92*	TSU:STO	Stop condition Setup time	100 kHz mode	4700	—	—	ns	
			400 kHz mode	600	—	—		
SP93	THD:STO	Stop condition Hold time	100 kHz mode	4000	—	—	ns	
			400 kHz mode	600	—	—		

* These parameters are characterized but not tested.

FIGURE 46-19: I²C BUS DATA TIMING



Note: Refer to Figure 46-4 for load conditions.