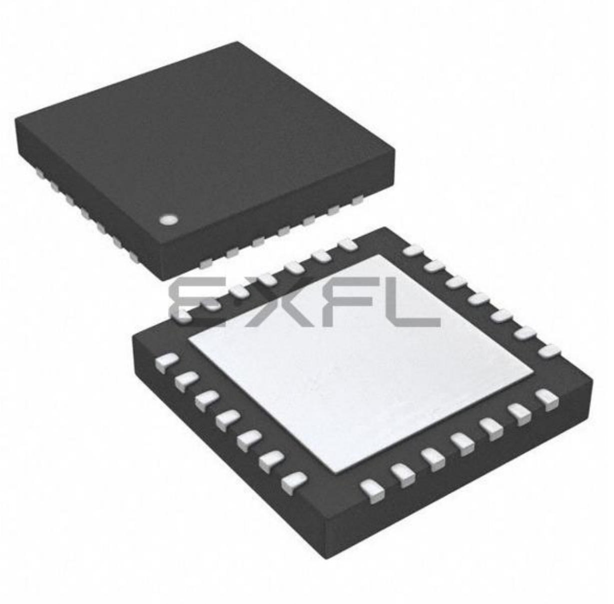


Welcome to E-XFL.COM

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	128KB (64K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	8K 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 ~ 85°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/pic18f27k42t-i-ml

3.0 PIC18 CPU

This family of devices contains a PIC18 8-bit CPU core based on the modified Harvard architecture. The PIC18 CPU supports:

- System Arbitration, which decides memory access allocation depending on user priorities
- Vectored Interrupt capability with automatic two level deep context saving
- 31-level deep hardware stack with overflow and underflow reset capabilities
- Support Direct, Indirect, and Relative Addressing modes
- 8x8 Hardware Multiplier

3.1.1 PRIORITY LOCK

The System arbiter grants memory access to the peripheral selections (DMAx, Scanner) when the PRLOCKED bit (PRLOCK Register) is set.

Priority selections are locked by setting the PRLOCKED bit of the PRLOCK register. Setting and clearing this bit requires a special sequence as an extra precaution against inadvertent changes. Examples of setting and clearing the PRLOCKED bit are shown in Example 3-1 and Example 3-2.

EXAMPLE 3-1: PRIORITY LOCK SEQUENCE

```
; Disable interrupts
BCF INTCON0,GIE

; Bank to PRLOCK register
BANKSEL PRLOCK
MOVLW 55h

; Required sequence, next 4
instructions
MOVWF PRLOCK
MOVLW AAh
MOVWF PRLOCK
; Set PRLOCKED bit to grant memory
access to peripherals
BSF PRLOCK,0

; Enable Interrupts
BSF INTCON0,GIE
```

EXAMPLE 3-2: PRIORITY UNLOCK SEQUENCE

```
; Disable interrupts
BCF INTCON0,GIE

; Bank to PRLOCK register
BANKSEL PRLOCK
MOVLW 55h

; Required sequence, next 4
instructions
MOVWF PRLOCK
MOVLW AAh
MOVWF PRLOCK
; Clear PRLOCKED bit to allow changing
priority settings
BCF PRLOCK,0

; Enable Interrupts
BSF INTCON0,GIE
```

3.2 Memory Access Scheme

The user can assign priorities to both system level and peripheral selections based on which the system arbiter grants memory access. Let us consider the following priority scenarios between ISR, MAIN, and Peripherals.

Note:	It is always required that the ISR priority be higher than Main priority.

3.2.1 ISR PRIORITY > MAIN PRIORITY > PERIPHERAL PRIORITY

When the Peripheral Priority (DMAx, Scanner) is lower than ISR and MAIN Priority, and the peripheral requires:

1. Access to the Program Flash Memory, then the peripheral waits for an instruction cycle in which the CPU does not need to access the PFM (such as a branch instruction) and uses that cycle to do its own Program Flash Memory access, unless a PFM Read/Write operation is in progress.
2. Access to the SFR/GPR, then the peripheral waits for an instruction cycle in which the CPU does not need to access the SFR/GPR (such as `MOVLW`, `CALL`, `NOP`) and uses that cycle to do its own SFR/GPR access.
3. Access to the Data EEPROM, then the peripheral has access to Data EEPROM unless a Data EEPROM Read/Write operation is being performed.

This results in the lowest throughput for the peripheral to access the memory, and does so without any impact on execution times.

3.2.2 PERIPHERAL PRIORITY > ISR PRIORITY > MAIN PRIORITY

When the Peripheral Priority (DMAx, Scanner) is higher than ISR and MAIN Priority, the CPU operation is stalled when the peripheral requests memory.

The CPU is held in its current state until the peripheral completes its operation. Since the peripheral requests access to the bus, the peripheral cannot be disabled until it completes its operation.

This results in the highest throughput for the peripheral to access the memory, but has the cost of stalling other execution while it occurs.

TABLE 4-9: SPECIAL FUNCTION REGISTER MAP FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES BANK 58

3AFFh	—	3ADFh	SPI1SDIPPS	3ABFh	PPSLOCK	3A9Fh	—	3A7Fh	—	3A5Fh	—	3A3Fh	—	3A1Fh	RD7PPS[2]
3AFEh	—	3ADEh	SPI1SCKPPS	3ABEh	—[4]	3A9Eh	—	3A7Eh	—	3A5Eh	—	3A3Eh	—	3A1Eh	RD6PPS[2]
3AFDh	—	3ADDh	ADACTPPS	3ABDh	—	3A9Dh	—	3A7Dh	—	3A5Dh	—	3A3Dh	—	3A1Dh	RD5PPS[2]
3AFCh	—	3ADCh	CLCIN3PPS	3ABCh	—	3A9Ch	—	3A7Ch	—	3A5Ch	—	3A3Ch	—	3A1Ch	RD4PPS[2]
3AFBh	—	3ADBh	CLCIN2PPS	3ABBh	—	3A9Bh	—	3A7Bh	RD1I2C[2]	3A5Bh	RB2I2C	3A3Bh	—	3A1Bh	RD3PPS[2]
3AFAh	—	3ADAh	CLCIN1PPS	3ABAh	—	3A9Ah	—	3A7Ah	RD0I2C[2]	3A5Ah	RB1I2C	3A3Ah	—	3A1Ah	RD2PPS[2]
3AF9h	—	3AD9h	CLCIN0PPS	3AB9h	—	3A99h	—[4]	3A79h	—[4]	3A59h	—[4]	3A39h	—	3A19h	RD1PPS[2]
3AF8h	—	3AD8h	MD1SRCPPS	3AB8h	—	3A98h	—[4]	3A78h	—[4]	3A58h	—[4]	3A38h	—	3A18h	RD0PPS[2]
3AF7h	—	3AD7h	MD1CARHPPS	3AB7h	—	3A97h	—	3A77h	—	3A57h	IOCBF	3A37h	—	3A17h	RC7PPS
3AF6h	—	3AD6h	MD1CARLPPS	3AB6h	—	3A96h	—	3A76h	—	3A56h	IOCBN	3A36h	—	3A16h	RC6PPS
3AF5h	—	3AD5h	CWG3INPPS	3AB5h	—	3A95h	—	3A75h	—	3A55h	IOCBP	3A35h	—	3A15h	RC5PPS
3AF4h	—	3AD4h	CWG2INPPS	3AB4h	—	3A94h	INLVLF[3]	3A74h	INLVLD[2]	3A54h	INLVLB	3A34h	—	3A14h	RC4PPS
3AF3h	—	3AD3h	CWG1INPPS	3AB3h	—	3A93h	SLRCONF[3]	3A73h	SLRCOND[2]	3A53h	SLRCONB	3A33h	—	3A13h	RC3PPS
3AF2h	—	3AD2h	SMT1SIGPPS	3AB2h	—	3A92h	ODCONF[3]	3A72h	ODCOND[2]	3A52h	ODCONB	3A32h	—	3A12h	RC2PPS
3AF1h	—	3AD1h	SMT1WINPPS	3AB1h	—	3A91h	WPUF[3]	3A71h	WPUD[2]	3A51h	WPUB	3A31h	—	3A11h	RC1PPS
3AF0h	—	3AD0h	CCP4PPS	3AB0h	—	3A90h	ANSELF[3]	3A70h	ANSELD[2]	3A50h	ANSELB	3A30h	—	3A10h	RC0PPS
3AEFh	—	3ACFh	CCP3PPS	3AAFh	—	3A8Fh	—	3A6Fh	—	3A4Fh	—	3A2Fh	RF7PPS[3]	3A0Fh	RB7PPS
3AEEh	—	3ACEh	CCP2PPS	3AAEh	—	3A8Eh	—	3A6Eh	—	3A4Eh	—	3A2Eh	RF6PPS[3]	3A0Eh	RB6PPS
3AEDh	—	3ACDh	CCP1PPS	3AADh	—	3A8Dh	—	3A6Dh	—	3A4Dh	—	3A2Dh	RF5PPS[3]	3A0Dh	RB5PPS
3AECh	—	3ACCh	T6INPPS	3AACh	—	3A8Ch	—	3A6Ch	—	3A4Ch	—	3A2Ch	RF4PPS[3]	3A0Ch	RB4PPS
3AEBh	—	3ACBh	T4INPPS	3AABh	—	3A8Bh	—	3A6Bh	RC4I2C	3A4Bh	—	3A2Bh	RF3PPS[3]	3A0Bh	RB3PPS
3AEAh	—	3ACAh	T2INPPS	3AAAh	—	3A8Ah	—	3A6Ah	RC3I2C	3A4Ah	—	3A2Ah	RF2PPS[3]	3A0Ah	RB2PPS
3AE9h	U2CTSPPS	3AC9h	T5GPPS	3AA9h	—	3A89h	—[4]	3A69h	—[4]	3A49h	—[4]	3A29h	RF1PPS[3]	3A09h	RB1PPS
3AE8h	U2RXPPS	3AC8h	T5CKIPPS	3AA8h	—	3A88h	—[4]	3A68h	—[4]	3A48h	—[4]	3A28h	RF0PPS[3]	3A08h	RB0PPS
3AE7h	—	3AC7h	T3GPPS	3AA7h	—	3A87h	IOCEF	3A67h	IOCCF	3A47h	IOCAF	3A27h	—	3A07h	RA7PPS
3AE6h	U1CTSPPS	3AC6h	T3CKIPPS	3AA6h	—	3A86h	IOCEN	3A66h	IOCCN	3A46h	IOCAN	3A26h	—	3A06h	RA6PPS
3AE5h	U1RXPPS	3AC5h	T1GPPS	3AA5h	—	3A85h	IOCEP	3A65h	IOCCP	3A45h	IOCAP	3A25h	—	3A05h	RA5PPS
3AE4h	I2C2SDAPPS	3AC4h	T1CKIPPS	3AA4h	—	3A84h	INLVLE	3A64h	INLVLC	3A44h	INLVLA	3A24h	—	3A04h	RA4PPS
3AE3h	I2C2SCLPPS	3AC3h	T0CKIPPS	3AA3h	—	3A83h	SLRCONE[2]	3A63h	SLRCONC	3A43h	SLRCONA	3A23h	—	3A03h	RA3PPS
3AE2h	I2C1SDAPPS	3AC2h	INT2PPS	3AA2h	—	3A82h	ODCONE[2]	3A62h	ODCONC	3A42h	ODCONA	3A22h	RE2PPS[2]	3A02h	RA2PPS
3AE1h	I2C1SCLPPS	3AC1h	INT1PPS	3AA1h	—	3A81h	WPUE	3A61h	WPUC	3A41h	WPUA	3A21h	RE1PPS[2]	3A01h	RA1PPS
3AE0h	SPI1SSPPS	3AC0h	INT0PPS	3AA0h	—	3A80h	ANSELE[2]	3A60h	ANSELC	3A40h	ANSELA	3A20h	RE0PPS[2]	3A00h	RA0PPS

Legend: Unimplemented data memory locations and registers, read as '0'.

Note 1: Unimplemented in LF devices.
2: Unimplemented in PIC18(L)F26/27K42.
3: Unimplemented in PIC18(L)F26/27/45/46/47K42.
4: Reserved, maintain as '0'.

9.12 Register Definitions: Interrupt Control

REGISTER 9-1: INTCON0: INTERRUPT CONTROL REGISTER 0

R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-1/1	R/W-1/1	R/W-1/1
GIE/GIEH	GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **GIE/GIEH:** Global Interrupt Enable bits

If IPEN = 0:

GIE:

1 = Enables all unmasked interrupts
0 = Disables all interrupts

If IPEN = 1:

GIEH:

1 = Enables all unmasked high priority interrupts: bit also needs to be set for enabling low priority interrupts
0 = Disables all interrupts

bit 6 **GIEL:** Global Low Priority Interrupt Enable bit

If IPEN = 0:

Reserved, read as ‘0’

If IPEN = 1:

GIEL:

1 = Enables all unmasked low priority interrupts, GIEH also needs to be set for low priority interrupts
0 = Disables all low priority

bit 5 **IPEN:** Interrupt Priority Enable bit

1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts; all interrupts are treated as high priority interrupts

bit 4-3 **Unimplemented**: Read as ‘0’

bit 2 **INT2EDG:** External Interrupt 2 Edge Select bit

1 = Interrupt on rising edge of INT2 pin
0 = Interrupt on falling edge of INT2 pin

bit 1 **INT1EDG:** External Interrupt 1 Edge Select bit

1 = Interrupt on rising edge of INT1 pin
0 = Interrupt on falling edge of INT1 pin

bit 0 **INT0EDG:** External Interrupt 0 Edge Select bit

1 = Interrupt on rising edge of INT0 pin
0 = Interrupt on falling edge of INT0 pin

REGISTER 9-19: PIE5: PERIPHERAL INTERRUPT ENABLE REGISTER 5

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
I2C2TXIE	I2C2RXIE	DMA2AIE	DMA2ORIE	DMA2DCNTIE	DMA2SCNTIE	C2IE	INT1IE
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 **I2C2TXIE:** I^{2}C2 Transmit Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6 **I2C2RXIE:** I^{2}C2 Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 5 **DMA2AIE:** DMA2 Abort Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **DMA2ORIE:** DMA2 Overrun Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **DMA2DCNTIE:** DMA2 Destination Count Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **DMA2SCNTIE:** DMA2 Source Count Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **C2IE:** C2 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **INT1IE:** External Interrupt 1 Enable bit
1 = Enabled
0 = Disabled

REGISTER 10-2: CPUDOZE: DOZE AND IDLE REGISTER

R/W-0/0	R/W/HC/HS-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
IDLEN	DOZEN	ROI	DOE	—	DOZE<2:0>		
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	HC = Bit is cleared by hardware; HS = Bit is set by hardware

bit 7 **IDLEN:** Idle Enable bit
1 = A `SLEEP` instruction places the device into Idle mode
0 = A `SLEEP` instruction places the device into Sleep mode

bit 6 **DOZEN:** Doze Enable bit(1)
1 = Places the device into Doze mode
0 = Places the device into Normal mode

bit 5 **ROI:** Recover-On-Interrupt bit(1)
1 = Entering the Interrupt Service Routine (ISR) makes DOZEN = 0
0 = Entering the Interrupt Service Routine (ISR) does not change DOZEN

bit 4 **DOE:** Doze-On-Exit bit(1)
1 = Exiting the Interrupt Service Routine (ISR) makes DOZEN = 1
0 = Exiting the Interrupt Service Routine (ISR) does not change DOZEN

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **DOZE<2:0>:** Ratio of CPU Instruction Cycles to Peripheral Instruction Cycles
111 =1:256
110 =1:128
101 =1:64
100 =1:32
011 =1:16
010 =1:8
001 =1:4
000 =1:2

Note 1: Refer Table 10-1 for more details.

TABLE 10-2: SUMMARY OF REGISTERS ASSOCIATED WITH POWER-DOWN MODE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
VREGCON(1)	—	—	—	—	—	—	VREGPM	Reserved	176
CPUDOZE	IDLEN	DOZEN	ROI	DOE	—	DOZE<2:0>			177

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used in Power-Down mode.
Note 1: Not present in LF parts.

11.1 Independent Clock Source

The WWDT can derive its time base from either the 31 kHz LFINTOSC or 31.25 kHz MFINTOSC internal oscillators, depending on the value of WDTE<1:0> Configuration bits.

If WDTE = `0b1x`, then the clock source will be enabled depending on the WDTCCS<2:0> Configuration bits.

If WDTE = `0b01`, the SEN bit should be set by software to enable WWDT, and the clock source is enabled by the CS bits in the WDTCON1 register.

Time intervals in this chapter are based on a minimum nominal interval of 1 ms. See **Section 44.0 "Electrical Specifications"** for LFINTOSC and MFINTOSC tolerances.

11.2 WWDT Operating Modes

The Windowed Watchdog Timer module has four operating modes controlled by the WDTE<1:0> bits in Configuration Words. See Table 11-1.

11.2.1 WWDT IS ALWAYS ON

When the WDTE bits of Configuration Words are set to '`11`', the WWDT is always on.

WWDT protection is active during Sleep.

11.2.2 WWDT IS OFF IN SLEEP

When the WDTE bits of Configuration Words are set to '`10`', the WWDT is on, except in Sleep.

WWDT protection is not active during Sleep.

11.2.3 WWDT CONTROLLED BY SOFTWARE

When the WDTE bits of Configuration Words are set to '`01`', the WWDT is controlled by the SEN bit of the WDTCON0 register.

WWDT protection is unchanged by Sleep. See Table 11-1 for more details.

TABLE 11-1: WWDT OPERATING MODES

WDTE<1:0>	SEN	Device Mode	WWDT Mode
11	x	X	Active
10	x	Awake	Active
		Sleep	Disabled
01	1	X	Active
	0	X	Disabled
00	x	X	Disabled

11.3 Time-out Period

If the WDTCPS<4:0> Configuration bits default to `0b11111`, then the PS bits of the WDTCON0 register set the time-out period from 1 ms to 256 seconds (nominal). If any value other than the default value is assigned to WDTCPS<4:0> Configuration bits, then the timer period will be based on the WDTCPS<4:0> bits in the CONFIG3L register. After a Reset, the default time-out period is 2s.

11.4 Watchdog Window

The Windowed Watchdog Timer has an optional Windowed mode that is controlled by the WDTCWS<2:0> Configuration bits and WINDOW<2:0> bits of the WDTCON1 register. In the Windowed mode, the `CLRWDT` instruction must occur within the allowed window of the WDT period. Any `CLRWDT` instruction that occurs outside of this window will trigger a window violation and will cause a WWDT Reset, similar to a WWDT time out. See Figure 11-2 for an example.

The window size is controlled by the WINDOW<2:0> Configuration bits, or the WINDOW<2:0> bits of WDTCON1, if WDTCWS<2:0> = `111`.

The five Most Significant bits of the WDTTMR register are used to determine whether the window is open, as defined by the WINDOW<2:0> bits of the WDTCON1 register.

In the event of a window violation, a Reset will be generated and the $\overline{\text{WDTWV}}$ bit of the PCON0 register will be cleared. This bit is set by a POR or can be set in firmware.

11.5 Clearing the WWDT

The WWDT is cleared when any of the following conditions occur:

- Any Reset
- Valid `CLRWDT` instruction is executed
- Device enters Sleep
- Exit Sleep by Interrupt
- WWDT is disabled
- Oscillator Start-up Timer (OST) is running
- Any write to the WDTCON0 or WDTCON1 registers

11.5.1 `CLRWDT` CONSIDERATIONS (WINDOWED MODE)

When in Windowed mode, the WWDT must be armed before a `CLRWDT` instruction will clear the timer. This is performed by reading the WDTCON0 register. Executing a `CLRWDT` instruction without performing such an arming action will trigger a window violation regardless of whether the window is open or not.

See Table 11-2 for more information.

REGISTER 11-3: WDTPSL: WWDT PRESCALE SELECT LOW BYTE REGISTER (READ-ONLY)

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
PSCNT<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 **PSCNT<7:0>:** Prescale Select Low Byte bits(1)

Note 1: The 18-bit WDT prescale value, PSCNT<17:0> includes the WDTPSL, WDTPSH and the lower bits of the WDTTMR registers. PSCNT<17:0> is intended for debug operations and should not be read during normal operation.

REGISTER 11-4: WDTPSH: WWDT PRESCALE SELECT HIGH BYTE REGISTER (READ-ONLY)

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
PSCNT<15:8>							
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 7-0 **PSCNT<15:8>:** Prescale Select High Byte bits(1)

Note 1: The 18-bit WDT prescale value, PSCNT<17:0> includes the WDTPSL, WDTPSH and the lower bits of the WDTTMR registers. PSCNT<17:0> is intended for debug operations and should not be read during normal operation.

REGISTER 14-14: SCANLADRL: SCAN LOW ADDRESS LOW BYTE 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
LADR<7:0>(1, 2)							
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 **LADR<7:0>**: Scan Start/Current Address bits(1, 2)
Least Significant bits of the current address to be fetched from, value increments on each fetch of memory

Note 1: Registers SCANLADRU/H/L form a 22-bit value, but are not guarded for atomic or asynchronous access; registers should only be read or written while SGO = `0` (SCANCON0 register).
2: While SGO = `1` (SCANCON0 register), writing to this register is ignored.

REGISTER 14-15: SCANHADRU: SCAN HIGH ADDRESS UPPER BYTE REGISTER

U-0	U-0	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
—	—	HADR<21:16>					
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 5-0 **HADR<21:16>:** Scan End Address bits(1, 2)
Upper bits of the address at the end of the designated scan

Note 1: Registers SCANHADRU/H/L form a 22-bit value but are not guarded for atomic or asynchronous access; registers should only be read or written while SGO = `0` (SCANCON0 register).
2: While SGO = `1` (SCANCON0 register), writing to this register is ignored.

REGISTER 15-9: DMAxSPTRU: DMAx SOURCE POINTER UPPER REGISTER

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0
—	—	SPTR<21:16>					
bit 7							bit 0

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

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

bit 5-0 **SPTR<21:16>:** Current Source Address Pointer

REGISTER 15-10: DMAxSSZL: DMAx SOURCE SIZE 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
SSZ<7:0>							
bit 7							bit 0

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

bit 7-0 **SSZ<7:0>:** Source Message Size bits

REGISTER 15-11: DMAxSSZH: DMAx SOURCE SIZE HIGH REGISTER

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	SSZ<11:8>			
bit 7							bit 0

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

bit 7-4 **Unimplemented:** Read as ‘0’

bit 3-0 **SSZ<11:8>:** Source Message Size bits

18.6 Register Definitions: Interrupt-on-Change Control

REGISTER 18-1: IOCxP: INTERRUPT-ON-CHANGE POSITIVE EDGE REGISTER EXAMPLE

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
IOCxP7	IOCxP6	IOCxP5	IOCxP4	IOCxP3	IOCxP2	IOCxP1	IOCxP0
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 **IOCxP<7:0>:** Interrupt-on-Change Positive Edge Enable bits
1 = Interrupt-on-Change enabled on the IOCx pin for a positive-going edge. Associated Status bit and interrupt flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

REGISTER 18-2: IOCxN: INTERRUPT-ON-CHANGE NEGATIVE EDGE REGISTER EXAMPLE

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
IOCxN7	IOCxN6	IOCxN5	IOCxN4	IOCxN3	IOCxN2	IOCxN1	IOCxN0
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 **IOCxN<7:0>:** Interrupt-on-Change Negative Edge Enable bits
1 = Interrupt-on-Change enabled on the IOCx pin for a negative-going edge. Associated Status bit and interrupt flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin

REGISTER 18-3: IOCxF: INTERRUPT-ON-CHANGE FLAG REGISTER EXAMPLE

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
IOCxF7	IOCxF6	IOCxF5	IOCxF4	IOCxF3	IOCxF2	IOCxF1	IOCxF0
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 7-0 **IOCxF<7:0>:** Interrupt-on-Change Flag bits
1 = A enabled change was detected on the associated pin. Set when IOCP[n] = 1 and a positive edge was detected on the IOCn pin, or when IOCN[n] = 1 and a negative edge was detected on the IOCn pin
0 = No change was detected, or the user cleared the detected change

20.1 Timer0 Operation

Timer0 can operate as either an 8-bit timer/counter or a 16-bit timer/counter. The mode is selected with the MD16 bit of the T0CON register.

20.1.1 16-BIT MODE

The register pair TMR0H:TMR0L increments on the rising edge of the clock source. A 15-bit prescaler on the clock input gives several prescale options (see prescaler control bits, CKPS<3:0> in the T0CON1 register).

20.1.1.1 Timer0 Reads and Writes in 16-Bit Mode

In 16-bit mode, in order to avoid rollover between reading high and low registers, the TMR0H register is a buffered copy of the actual high byte of Timer0, which is neither directly readable, nor writable (see Figure 20-1). TMR0H is updated with the contents of the high byte of Timer0 during a read of TMR0L. This provides the ability to read all 16 bits of Timer0 without having to verify that the read of the high and low byte was valid, due to a rollover between successive reads of the high and low byte.

Similarly, a write to the high byte of Timer0 must also take place through the TMR0H Buffer register. The high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16 bits of Timer0 to be updated at once.

20.1.2 8-BIT MODE

In 8-bit mode, the value of TMR0L is compared to that of the Period buffer, a copy of TMR0H, on each clock cycle. When the two values match, the following events happen:

- TMR0_out goes high for one prescaled clock period
- TMR0L is reset
- The contents of TMR0H are copied to the period buffer

In 8-bit mode, the TMR0L and TMR0H registers are both directly readable and writable. The TMR0L register is cleared on any device Reset, while the TMR0H register initializes at FFh.

Both the prescaler and postscaler counters are cleared on the following events:

- A write to the TMR0L register
- A write to either the T0CON0 or T0CON1 registers
- Any device Reset – Power-on Reset (POR), $\overline{\text{MCLR}}$ Reset, Watchdog Timer Reset (WDTR) or
- Brown-out Reset (BOR)

20.1.3 COUNTER MODE

In Counter mode, the prescaler is normally disabled by setting the CKPS bits of the T0CON1 register to ‘0000’. Each rising edge of the clock input (or the output of the prescaler if the prescaler is used) increments the counter by ‘1’.

20.1.4 TIMER MODE

In Timer mode, the Timer0 module will increment every instruction cycle as long as there is a valid clock signal and the CKPS bits of the T0CON1 register (Register 20-2) are set to ‘0000’. When a prescaler is added, the timer will increment at the rate based on the prescaler value.

20.1.5 ASYNCHRONOUS MODE

When the ASYNC bit of the T0CON1 register is set (ASYNC = ‘1’), the counter increments with each rising edge of the input source (or output of the prescaler, if used). Asynchronous mode allows the counter to continue operation during Sleep mode provided that the clock also continues to operate during Sleep.

20.1.6 SYNCHRONOUS MODE

When the ASYNC bit of the T0CON1 register is clear (ASYNC = ‘0’), the counter clock is synchronized to the system clock (FOSC/4). When operating in Synchronous mode, the counter clock frequency cannot exceed FOSC/4.

20.2 Clock Source Selection

The CS<2:0> bits of the T0CON1 register are used to select the clock source for Timer0. Register 20-2 displays the clock source selections.

20.2.1 INTERNAL CLOCK SOURCE

When the internal clock source is selected, Timer0 operates as a timer and will increment on multiples of the clock source, as determined by the Timer0 prescaler.

20.2.2 EXTERNAL CLOCK SOURCE

When an external clock source is selected, Timer0 can operate as either a timer or a counter. Timer0 will increment on multiples of the rising edge of the external clock source, as determined by the Timer0 prescaler.

24.1.9 SETUP FOR PWM OPERATION USING PWMx PINS

The following steps should be taken when configuring the module for PWM operation using the PWMx pins:

1. Disable the PWMx pin output driver(s) by setting the associated TRIS bit(s).
2. Clear the PWMxCON register.
3. Load the T2PR register with the PWM period value.
4. Load the PWMxDCH register and bits <7:6> of the PWMxDCL register with the PWM duty cycle value.
5. Configure and start Timer2:
   - Clear the TMR2IF interrupt flag bit of the respective PIR register. See Note 1 below.
   - Select the timer clock source to be as FOSC/4 using the TxCLK register. This is required for correct operation of the PWM module.
   - Configure the CKPS bits of the T2CON register with the Timer2 prescale value.
   - Enable Timer2 by setting the ON bit of the T2CON register.
6. Enable PWM output pin and wait until Timer2 overflows, TMR2IF bit of the respective PIR register is set. See note below.
7. Enable the PWMx pin output driver(s) by clearing the associated TRIS bit(s) and setting the desired pin PPS control bits.
8. Configure the PWM module by loading the PWMxCON register with the appropriate values.

Note 1: In order to send a complete duty cycle and period on the first PWM output, the above steps must be followed in the order given. If it is not critical to start with a complete PWM signal, then move Step 8 to replace Step 4.

2: For operation with other peripherals only, disable PWMx pin outputs.

24.1.10 SETUP FOR PWM OPERATION TO OTHER DEVICE PERIPHERALS

The following steps should be taken when configuring the module for PWM operation to be used by other device peripherals:

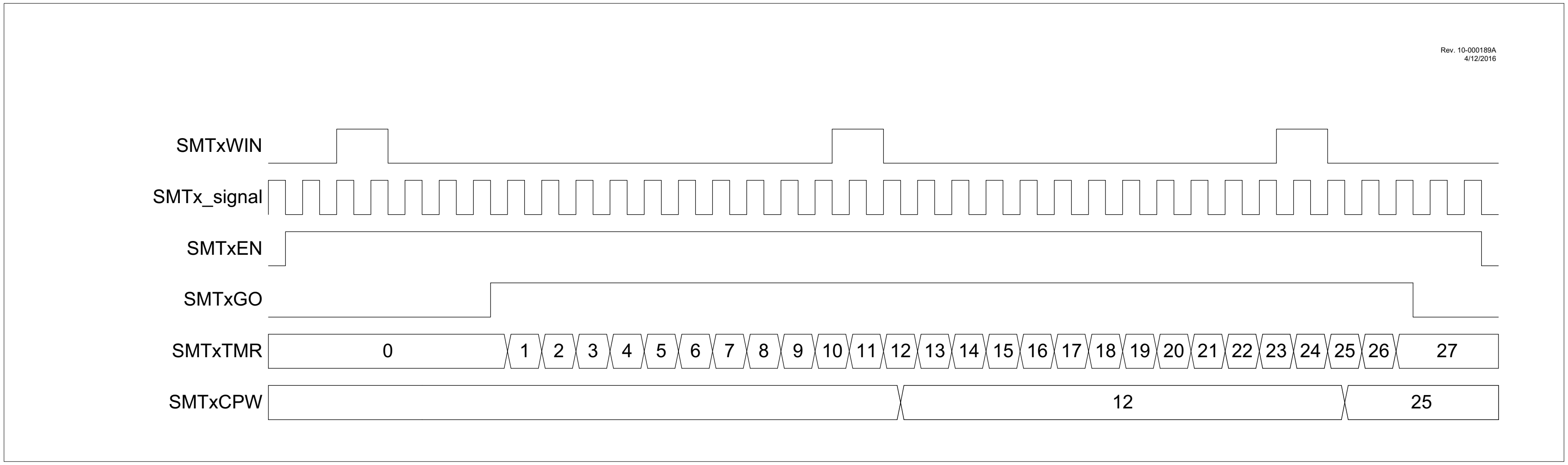
1. Disable the PWMx pin output driver(s) by setting the associated TRIS bit(s).
2. Clear the PWMxCON register.
3. Load the T2PR register with the PWM period value.
4. Load the PWMxDCH register and bits <7:6> of the PWMxDCL register with the PWM duty cycle value.
5. Configure and start Timer2:
   - Clear the TMR2IF interrupt flag bit of the respective PIR register. See Note 1 below.
   - Select the timer clock source to be as FOSC/4 using the TxCLK register. This is required for correct operation of the PWM module.
   - Configure the CKPS bits of the T2CON register with the Timer2 prescale value.
   - Enable Timer2 by setting the ON bit of the T2CON register.
6. Enable PWM output pin:
   - Wait until Timer2 overflows, TMR2IF bit of the respective PIR register is set. See Note 1 below.
7. Configure the PWM module by loading the PWMxCON register with the appropriate values.

Note 1: In order to send a complete duty cycle and period on the first PWM output, the above steps must be included in the setup sequence. If it is not critical to start with a complete PWM signal on the first output, then step 6 may be ignored.

25.6.9 COUNTER MODE

This mode increments the timer on each pulse of the SMT1_signal input. This mode is asynchronous to the SMT clock and uses the SMT1_signal as a time source. The SMT1CPW register will be updated with the current SMT1TMR value on the rising edge of the SMT1WIN input. See Figure 25-18.

FIGURE 25-18: COUNTER MODE TIMING DIAGRAM



Preliminary

25.8 Register Definitions: SMT Control

Long bit name prefixes for the Signal Measurement Timer peripherals are shown in **Section 1.3 "Register and Bit naming conventions"**.

TABLE 25-2: LONG BIT NAMES PREFIXES FOR SMT PERIPHERALS

Peripheral	Bit Name Prefix
SMT1	SMT1

REGISTER 25-1: SMT1CON0: SMT CONTROL REGISTER 0

R/W-0/0	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
EN[(1)]	—	STP	WPOL	SPOL	CPOL	PS<1:0>	
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 **EN:** SMT Enable bit[(1)]
1 = SMT is enabled
0 = SMT is disabled; internal states are reset, clock requests are disabled

bit 6 **Unimplemented**: Read as '0'

bit 5 **STP:** SMT Counter Halt Enable bit
When SMT1TMR = SMT1PR:
1 = Counter remains SMT1PR; period match interrupt occurs when clocked
0 = Counter resets to 24'h000000; period match interrupt occurs when clocked

bit 4 **WPOL:** SMT1WIN Input Polarity Control bit
1 = SMT1WIN signal is active-low/falling edge enabled
0 = SMT1WIN signal is active-high/rising edge enabled

bit 3 **SPOL:** SMT1SIG Input Polarity Control bit
1 = SMT1_signal is active-low/falling edge enabled
0 = SMT1_signal is active-high/rising edge enabled

bit 2 **CPOL:** SMT Clock Input Polarity Control bit
1 = SMT1TMR increments on the falling edge of the selected clock signal
0 = SMT1TMR increments on the rising edge of the selected clock signal

bit 1-0 **PS<1:0>:** SMT Prescale Select bits
11 = Prescaler = 1:8
10 = Prescaler = 1:4
01 = Prescaler = 1:2
00 = Prescaler = 1:1

Note 1: Setting EN to '0' does not affect the register contents.

REGISTER 25-6: SMT1SIG: SMT1 SIGNAL INPUT SELECT REGISTER

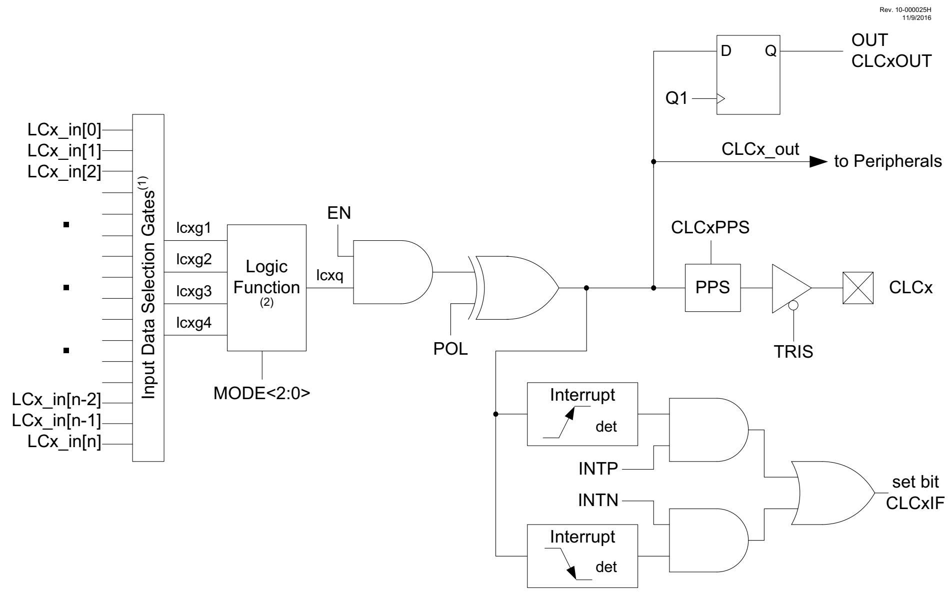
U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	SSEL<4:0>				
bit 7							bit 0

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	q = Value depends on condition

bit 7-5 **Unimplemented**: Read as ‘0’

bit 4-0 **SSEL<4:0>:** SMT1 Signal Selection bits
11111 = Reserved
•
•
•
11010 = Reserved
11001 = CLC4_out
11000 = CLC3_out
10111 = CLC2_out
10110 = CLC1_out
10101 = ZCD1_out
10100 = CMP2_out
10011 = CMP1_out
10010 = NCO1_out
10001 = Reserved
10000 = Reserved
01111 = PWM8_out
01110 = PWM7_out
01101 = PWM6_out
01100 = PWM5_out
01011 = CCP4_out
01010 = CCP3_out
01001 = CCP2_out
01000 = CCP1_out
00111 = TMR6_postscaled
00110 = TMR5_postscaled
00101 = TMR4_postscaled
00100 = TMR3_postscaled
00011 = TMR2_postscaled
00010 = TMR1_postscaled
00001 = TMR0_overflow
00000 = SMTxSIGPPS

FIGURE 27-1: CLCx SIMPLIFIED BLOCK DIAGRAM



Note 1: See Figure 27-2: Input Data Selection and Gating

2: See Figure 27-3: Programmable Logic Functions.

27.1 CLCx Setup

Programming the CLCx module is performed by configuring the four stages in the logic signal flow. The four stages are:

- Data selection
- Data gating
- Logic function selection
- Output polarity

Each stage is setup at run time by writing to the corresponding CLCx Special Function Registers. This has the added advantage of permitting logic reconfiguration on-the-fly during program execution.

27.1.1 DATA SELECTION

There are 32 signals available as inputs to the configurable logic. Four 32-input multiplexers are used to select the inputs to pass on to the next stage.

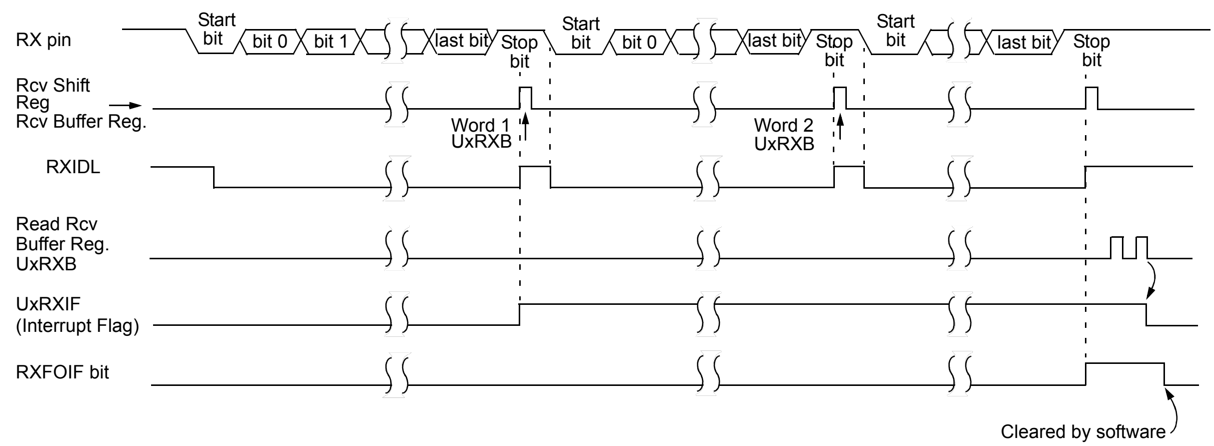
Data selection is through four multiplexers as indicated on the left side of Figure 27-2. Data inputs in the figure are identified by a generic numbered input name.

Table 27-1 correlates the generic input name to the actual signal for each CLC module. The column labeled 'DyS<4:0> Value' indicates the MUX selection code for the selected data input. DyS is an abbreviation for the MUX select input codes: D1S<4:0> through D4S<4:0>.

Data inputs are selected with CLCxSEL0 through CLCxSEL3 registers (Register 27-3 through Register 27-6).

Note:	Data selections are undefined at power-up.

FIGURE 31-5: ASYNCHRONOUS RECEPTION



Note: This timing diagram shows three words appearing on the RX input. The UxRXB (receive buffer) is not read before the third word is received, causing the RXFOIF (FIFO overrun) bit to be set. STPMD = 0, STP<1:0> = 00.

CALLW Subroutine Call Using WREG

Syntax:	CALLW
Operands:	None
Operation:	(PC + 2) $\rightarrow$ TOS, (W) $\rightarrow$ PCL, (PCLATH) $\rightarrow$ PCH, (PCLATU) $\rightarrow$ PCU
Status Affected:	None
Encoding:	`0000` `0000` `0001` `0100`
Description	First, the return address (PC + 2) is pushed onto the return stack. Next, the contents of W are written to PCL; the existing value is discarded. Then, the contents of PCLATH and PCLATU are latched into PCH and PCU, respectively. The second cycle is executed as a `NOP` instruction while the new next instruction is fetched. Unlike `CALL`, there is no option to update W, Status or BSR.
Words:	1
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read WREG	PUSH PC to stack	No operation
No operation	No operation	No operation	No operation

Example: `HERE CALLW`

Before Instruction
PC = address `(HERE)`
PCLATH = 10h
PCLATU = 00h
W = 06h
After Instruction
PC = 001006h
TOS = address `(HERE + 2)`
PCLATH = 10h
PCLATU = 00h
W = 06h

CLRF Clear f

Syntax:	CLRF f {,a}
Operands:	$0 \leq f \leq 255$ $a \in [0,1]$
Operation:	000h $\rightarrow$ f 1 $\rightarrow$ Z
Status Affected:	Z
Encoding:	`0110` `101a` `ffff` `ffff`
Description:	Clears the contents of the specified register. If 'a' is '`0`', the Access Bank is selected. If 'a' is '`1`', the BSR is used to select the GPR bank. If 'a' is '`0`' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 41.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `CLRF FLAG_REG, 1`

Before Instruction
FLAG_REG = 5Ah
After Instruction
FLAG_REG = 00h

SUBULNK **Subtract Literal from FSR2 and Return**

Syntax:	SUBULNK k
Operands:	$0 \le k \le 63$
Operation:	FSR2 – k → FSR2 (TOS) → PC
Status Affected:	None
Encoding:	`1110` `1001` `11kk` `kkkk`
Description:	The 6-bit literal 'k' is subtracted from the contents of the FSR2. A `RETURN` is then executed by loading the PC with the TOS. The instruction takes two cycles to execute; a `NOP` is performed during the second cycle. This may be thought of as a special case of the `SUBFSR` instruction, where f = 3 (binary '`11`'); it operates only on FSR2.
Words:	1
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination
No Operation	No Operation	No Operation	No Operation

Example: `SUBULNK 23h`

Before Instruction
FSR2 = 03FFh
PC = 0100h
After Instruction
FSR2 = 03DCh
PC = (TOS)