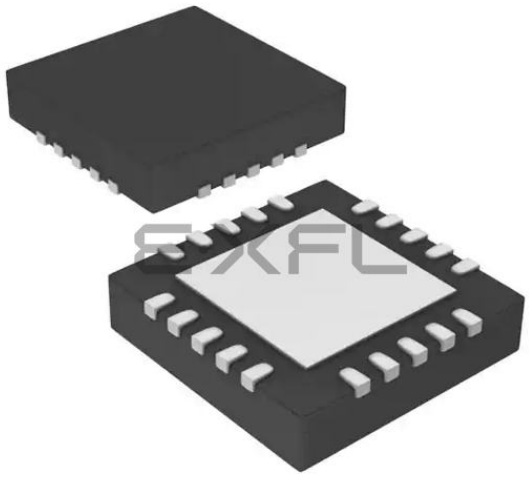


Welcome to E-XFL.COM

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.

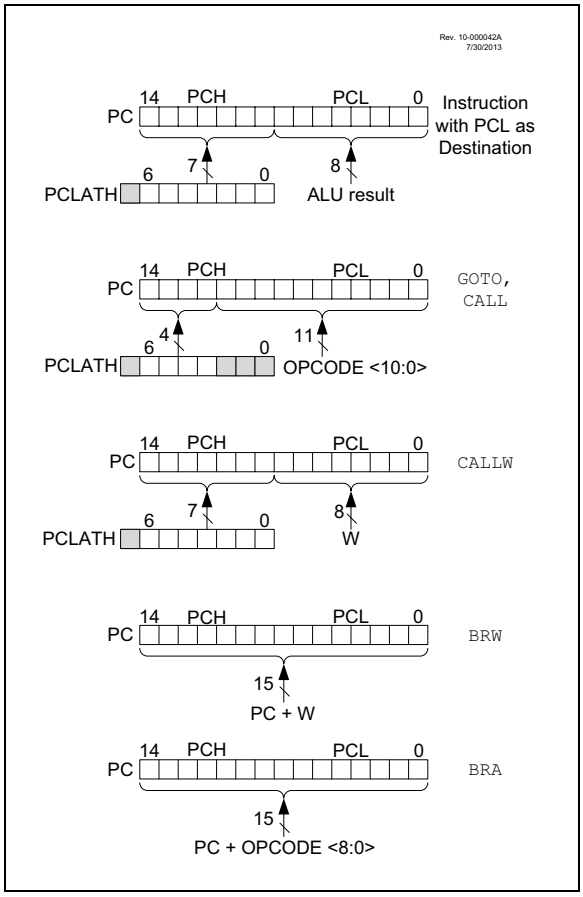
Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	32MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	18
Program Memory Size	7KB (4K x 14)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 8x10b; D/A 1x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	20-VFQFN Exposed Pad
Supplier Device Package	20-QFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f1618-e-ml

3.4 PCL and PCLATH

The Program Counter (PC) is 15 bits wide. The low byte comes from the PCL register, which is a readable and writable register. The high byte (PC<14:8>) is not directly readable or writable and comes from PCLATH. On any Reset, the PC is cleared. Figure 3-3 shows the five situations for the loading of the PC.

FIGURE 3-3: LOADING OF PC IN DIFFERENT SITUATIONS



3.4.1 MODIFYING PCL

Executing any instruction with the PCL register as the destination simultaneously causes the Program Counter PC<14:8> bits (PCH) to be replaced by the contents of the PCLATH register. This allows the entire contents of the program counter to be changed by writing the desired upper seven bits to the PCLATH register. When the lower eight bits are written to the PCL register, all 15 bits of the program counter will change to the values contained in the PCLATH register and those being written to the PCL register.

3.4.2 COMPUTED GOTO

A computed GOTO is accomplished by adding an offset to the program counter (ADDWF PCL). When performing a table read using a computed GOTO method, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block). Refer to Application Note AN556, *"Implementing a Table Read"* (DS00556).

3.4.3 COMPUTED FUNCTION CALLS

A computed function CALL allows programs to maintain tables of functions and provide another way to execute state machines or look-up tables. When performing a table read using a computed function CALL, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block).

If using the CALL instruction, the PCH<2:0> and PCL registers are loaded with the operand of the CALL instruction. PCH<6:3> is loaded with PCLATH<6:3>.

The CALLW instruction enables computed calls by combining PCLATH and W to form the destination address. A computed CALLW is accomplished by loading the W register with the desired address and executing CALLW. The PCL register is loaded with the value of W and PCH is loaded with PCLATH.

3.4.4 BRANCHING

The branching instructions add an offset to the PC. This allows relocatable code and code that crosses page boundaries. There are two forms of branching, BRW and BRA. The PC will have incremented to fetch the next instruction in both cases. When using either branching instruction, a PCL memory boundary may be crossed.

If using BRW, load the W register with the desired unsigned address and execute BRW. The entire PC will be loaded with the address PC + 1 + W.

If using BRA, the entire PC will be loaded with PC + 1 +, the signed value of the operand of the BRA instruction.

EXAMPLE 10-3: WRITING TO FLASH PROGRAM MEMORY (32 WRITE LATCHES)

```
; This write routine assumes the following:
; 1. 64 bytes of data are loaded, starting at the address in DATA_ADDR
; 2. Each word of data to be written is made up of two adjacent bytes in DATA_ADDR,
;    stored in little endian format
; 3. A valid starting address (the Least Significant bits = 00000) is loaded in ADDRH:ADDRL
; 4. ADDRH and ADDRL are located in shared data memory 0x70 - 0x7F (common RAM)
;
        BCF         INTCON,GIE     ; Disable ints so required sequences will execute properly
        BANKSEL     PMADRH         ; Bank 3
        MOVF        ADDRH,W        ; Load initial address
        MOVWF       PMADRH         ;
        MOVF        ADDRL,W        ;
        MOVWF       PMADRL         ;
        MOVLW       LOW DATA_ADDR  ; Load initial data address
        MOVWF       FSR0L          ;
        MOVLW       HIGH DATA_ADDR ; Load initial data address
        MOVWF       FSR0H          ;
        BCF         PMCON1,CFGS    ; Not configuration space
        BSF         PMCON1,WREN    ; Enable writes
        BSF         PMCON1,LWLO    ; Only Load Write Latches
LOOP
        MOVIW       FSR0++         ; Load first data byte into lower
        MOVWF       PMDATL         ;
        MOVIW       FSR0++         ; Load second data byte into upper
        MOVWF       PMDATH         ;

        MOVF        PMADRL,W       ; Check if lower bits of address are '00000'
        XORLW       0x1F           ; Check if we're on the last of 32 addresses
        ANDLW       0x1F           ;
        BTFSC       STATUS,Z       ; Exit if last of 32 words,
        GOTO        START_WRITE    ;

  Required Sequence
        MOVLW       55h            ; Start of required write sequence:
        MOVWF       PMCON2         ; Write 55h
        MOVLW       AAh            ;
        MOVWF       PMCON2         ; Write AAh
        BSF         PMCON1,WR      ; Set WR bit to begin write
        NOP                        ; NOP instructions are forced as processor
                                   ; loads program memory write latches
        NOP                        ;

        INCF        PMADRL,F       ; Still loading latches Increment address
        GOTO        LOOP           ; Write next latches

START_WRITE
        BCF         PMCON1,LWLO    ; No more loading latches - Actually start Flash program
                                   ; memory write

  Required Sequence
        MOVLW       55h            ; Start of required write sequence:
        MOVWF       PMCON2         ; Write 55h
        MOVLW       AAh            ;
        MOVWF       PMCON2         ; Write AAh
        BSF         PMCON1,WR      ; Set WR bit to begin write
        NOP                        ; NOP instructions are forced as processor writes
                                   ; all the program memory write latches simultaneously
        NOP                        ; to program memory.
                                   ; After NOPs, the processor
                                   ; stalls until the self-write process in complete
                                   ; after write processor continues with 3rd instruction
        BCF         PMCON1,WREN    ; Disable writes
        BSF         INTCON,GIE     ; Enable interrupts
```

12.0 I/O PORTS

Each port has six standard registers for its operation. These registers are:

- TRISx registers (data direction)
- PORTx registers (reads the levels on the pins of the device)
- LATx registers (output latch)
- INLVLx (input level control)
- ODCONx registers (open-drain)
- SLRCONx registers (slew rate)

Some ports may have one or more of the following additional registers. These registers are:

- ANSELx (analog select)
- WPUx (weak pull-up)

In general, when a peripheral is enabled on a port pin, that pin cannot be used as a general purpose output. However, the pin can still be read.

TABLE 12-1: PORT AVAILABILITY PER DEVICE

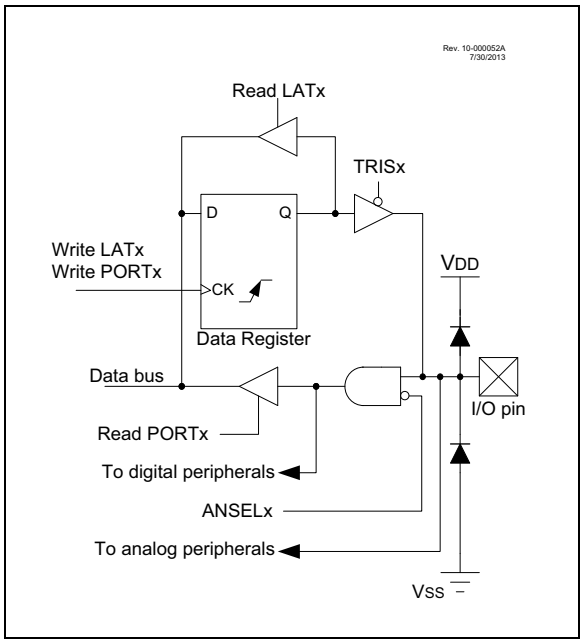
Device	PORTA	PORTB	PORTC
PIC16(L)F1618	•	•	•
PIC16(L)F1614	•		•

The Data Latch (LATx registers) is useful for read-modify-write operations on the value that the I/O pins are driving.

A write operation to the LATx register has the same effect as a write to the corresponding PORTx register. A read of the LATx register reads of the values held in the I/O PORT latches, while a read of the PORTx register reads the actual I/O pin value.

Ports that support analog inputs have an associated ANSELx register. When an ANSEL bit is set, the digital input buffer associated with that bit is disabled. Disabling the input buffer prevents analog signal levels on the pin between a logic high and low from causing excessive current in the logic input circuitry. A simplified model of a generic I/O port, without the interfaces to other peripherals, is shown in Figure 12-1.

FIGURE 12-1: GENERIC I/O PORT OPERATION



REGISTER 12-5: WPUA: WEAK PULL-UP PORTA 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
—	—	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0
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 **WPUA<5:0>**: Weak Pull-up Register bits(3)
1 = Pull-up enabled
0 = Pull-up disabled

Note 1: Global $\overline{\text{WPUEN}}$ bit of the OPTION_REG register must be cleared for individual pull-ups to be enabled.
2: The weak pull-up device is automatically disabled if the pin is configured as an output.
3: For the WPUA3 bit, when MCLRE = 1, weak pull-up is internally enabled, but not reported here.

REGISTER 12-6: ODCONA: PORTA OPEN-DRAIN CONTROL REGISTER

U-0	U-0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	ODA5	ODA4	—	ODA2	ODA1	ODA0
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 5-4 **ODA<5:4>:** PORTA Open-Drain Enable bits
For RA<5:4> pins, respectively
1 = Port pin operates as open-drain drive (sink current only)
0 = Port pin operates as standard push-pull drive (source and sink current)

bit 3 **Unimplemented:** Read as '0'

bit 2-0 **ODA<2:0>:** PORTA Open-Drain Enable bits
For RA<2:0> pins, respectively
1 = Port pin operates as open-drain drive (sink current only)
0 = Port pin operates as standard push-pull drive (source and sink current)

12.6 Register Definitions: PORTC

REGISTER 12-17: PORTC: PORTC REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
RC7(1)	RC6(1)	RC5	RC4	RC3	RC2	RC1	RC0
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 **RC<7:0>**: PORTC I/O Value bits(1, 2)
1 = Port pin is ≥ VIH
0 = Port pin is ≤ VIL

Note 1: RC<7:6> on PIC16(L)F1618 only.
2: Writes to PORTC are actually written to corresponding LATC register. Reads from PORTC register is return of actual I/O pin values.

REGISTER 12-18: TRISC: PORTC TRI-STATE REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TRISC7(1)	TRISC6(1)	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
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 **TRISC<7:0>**: PORTC Tri-State Control bits(1)
1 = PORTC pin configured as an input (tri-stated)
0 = PORTC pin configured as an output

Note 1: TRISC<7:6> on PIC16(L)F1618 only.

14.6 Register Definitions: Interrupt-on-Change Control

REGISTER 14-1: IOCAP: INTERRUPT-ON-CHANGE PORTA POSITIVE EDGE REGISTER

U-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
—	—	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0
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 **IOCAP<5:0>**: Interrupt-on-Change PORTA Positive Edge Enable bits
1 = Interrupt-on-Change enabled on the pin for a positive going edge. IOCAFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

REGISTER 14-2: IOCAN: INTERRUPT-ON-CHANGE PORTA NEGATIVE EDGE REGISTER

U-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
—	—	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0
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 IOCAN<5:0>: Interrupt-on-Change PORTA Negative Edge Enable bits
1 = Interrupt-on-Change enabled on the pin for a negative going edge. IOCAFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

REGISTER 14-3: IOCAF: INTERRUPT-ON-CHANGE PORTA FLAG REGISTER

U-0	U-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
—	—	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0
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 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **IOCAF<5:0>**: Interrupt-on-Change PORTA Flag bits
1 = An enabled change was detected on the associated pin.
Set when IOCAPx = 1 and a rising edge was detected on RAx, or when IOCANx = 1 and a falling edge was detected on RAx.
0 = No change was detected, or the user cleared the detected change.

TABLE 16-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE TEMPERATURE INDICATOR

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
FVRCON	FVREN	FVRRDY	TSEN	TSRNG	CDAFVR<1:0>		ADFVR<1:0>		118

Legend: Shaded cells are unused by the temperature indicator module.

TABLE 19-3: SUMMARY OF REGISTERS ASSOCIATED WITH COMPARATOR MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSELA	—	—	—	ANSA4	—	ANSA2	ANSA1	ANSA0	152
CM1CON0	C1ON	C1OUT	—	C1POL	—	C1SP	C1HYS	C1SYNC	212
CM1CON1	C1INTP	C1INTN	C1PCH<1:0>		—	C1NCH<2:0>			213
CM2CON0	C2ON	C2OUT	—	C2POL	—	C2SP	C2HYS	C2SYNC	212
CM2CON1	C2INTP	C2INTN	C2PCH<1:0>		—	C2NCH<2:0>			213
CMOUT	—	—	—	—	—	—	MC2OUT	MC1OUT	213
FVRCON	FVREN	FVRRDY	TSEN	TSRNG	CDAFVR<1:0>		ADFVR<1:0>		186
DAC1CON0	DAC1EN	—	DAC1OE1	—	DAC1PSS<1:0>		—	—	206
DAC1CON1	DAC1R<7:0>								206
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	97
PIE2	OSFIE	C2IE	C1IE	—	BCL1IE	TMR6IE	TMR4IE	CCP2IE	99
PIR2	OSFIF	C2IF	C1IF	—	BCL1IF	TMR6IF	TMR4IF	CCP2IF	104
TRISA	—	—	TRISA5	TRISA4	—(1)	TRISA2	TRISA1	TRISA0	151
TRISC(2)	TRISC7(2)	TRISC6(2)	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	165

Legend: — = unimplemented location, read as '0'. Shaded cells are unused by the comparator module.

Note 1: Unimplemented, read as '1'.

2: PIC16F1618 only.

22.8 Register Definitions: Timer1 Control

REGISTER 22-1: T1CON: TIMER1 CONTROL REGISTER

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	U-0	R/W-0/u	U-0	R/W-0/u
TMR1CS<1:0>		T1CKPS<1:0>		—	$\overline{\text{T1SYNC}}$	—	TMR1ON
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 **TMR1CS<1:0>:** Timer1 Clock Source Select bits
- 11 =LFINTOSC
- 10 =T1CKI
- 01 =FOSC
- 00 =FOSC/4

bit 5-4 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
- 11 =1:8 Prescale value
- 10 =1:4 Prescale value
- 01 =1:2 Prescale value
- 00 =1:1 Prescale value

bit 3 **Unimplemented:** Read as '0'

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 Synchronization Control bit
- 1 = Do not synchronize asynchronous clock input
- 0 = Synchronize asynchronous clock input with system clock (FOSC)

bit 1 **Unimplemented:** Read as '0'

bit 0 **TMR1ON:** Timer1 On bit
- 1 = Enables Timer1
- 0 = Stops Timer1 and clears Timer1 gate flip-flop

23.6 Timer2 Operation During Sleep

When PSYNC = 1, Timer2 cannot be operated while the processor is in Sleep mode. The contents of the TMR2 and T2PR registers will remain unchanged while processor is in Sleep mode.

When PSYNC = 0, Timer2 will operate in Sleep as long as the clock source selected is also still running. Selecting the LFINTOSC, MFINTOSC, or HFINTOSC oscillator as the timer clock source will keep the selected oscillator running during Sleep.

24.2 SPI Mode Overview

The Serial Peripheral Interface (SPI) bus is a synchronous serial data communication bus that operates in Full-Duplex mode. Devices communicate in a master/slave environment where the master device initiates the communication. A slave device is controlled through a Chip Select known as Slave Select.

The SPI bus specifies four signal connections:

- Serial Clock (SCK)
- Serial Data Out (SDO)
- Serial Data In (SDI)
- Slave Select ($\overline{SS}$)

Figure 24-1 shows the block diagram of the MSSP module when operating in SPI mode.

The SPI bus operates with a single master device and one or more slave devices. When multiple slave devices are used, an independent Slave Select connection is required from the master device to each slave device.

Figure 24-4 shows a typical connection between a master device and multiple slave devices.

The master selects only one slave at a time. Most slave devices have tri-state outputs so their output signal appears disconnected from the bus when they are not selected.

Transmissions involve two shift registers, eight bits in size, one in the master and one in the slave. With either the master or the slave device, data is always shifted out one bit at a time, with the Most Significant bit (MSb) shifted out first. At the same time, a new Least Significant bit (LSb) is shifted into the same register.

Figure 24-5 shows a typical connection between two processors configured as master and slave devices.

Data is shifted out of both shift registers on the programmed clock edge and latched on the opposite edge of the clock.

The master device transmits information out on its SDO output pin which is connected to, and received by, the slave's SDI input pin. The slave device transmits information out on its SDO output pin, which is connected to, and received by, the master's SDI input pin.

To begin communication, the master device first sends out the clock signal. Both the master and the slave devices should be configured for the same clock polarity.

The master device starts a transmission by sending out the MSb from its shift register. The slave device reads this bit from that same line and saves it into the LSb position of its shift register.

During each SPI clock cycle, a full-duplex data transmission occurs. This means that while the master device is sending out the MSb from its shift register (on its SDO pin) and the slave device is reading this bit and saving it as the LSb of its shift register, that the slave device is also sending out the MSb from its shift register (on its SDO pin) and the master device is reading this bit and saving it as the LSb of its shift register.

After eight bits have been shifted out, the master and slave have exchanged register values.

If there is more data to exchange, the shift registers are loaded with new data and the process repeats itself.

Whether the data is meaningful or not (dummy data), depends on the application software. This leads to three scenarios for data transmission:

- Master sends useful data and slave sends dummy data.
- Master sends useful data and slave sends useful data.
- Master sends dummy data and slave sends useful data.

Transmissions may involve any number of clock cycles. When there is no more data to be transmitted, the master stops sending the clock signal and it deselects the slave.

Every slave device connected to the bus that has not been selected through its slave select line must disregard the clock and transmission signals and must not transmit out any data of its own.

REGISTER 24-3: SSP1CON2: SSP CONTROL REGISTER 2(1)

R/W-0/0	R-0/0	R/W-0/0	R/S/HS-0/0	R/S/HS-0/0	R/S/HS-0/0	R/S/HS-0/0	R/W/HS-0/0
GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
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 = Cleared by hardware S = User set

bit 7 **GCEN:** General Call Enable bit (in I^2C Slave mode only)
1 = Enable interrupt when a general call address (0x00 or 00h) is received in the SSPSR
0 = General call address disabled

bit 6 **ACKSTAT:** Acknowledge Status bit (in I^2C mode only)
1 = Acknowledge was not received
0 = Acknowledge was received

bit 5 **ACKDT:** Acknowledge Data bit (in I^2C mode only)
In Receive mode:
Value transmitted when the user initiates an Acknowledge sequence at the end of a receive
1 = Not Acknowledge
0 = Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit (in I^2C Master mode only)
In Master Receive mode:
1 = Initiate Acknowledge sequence on SDA and SCL pins, and transmit ACKDT data bit. Automatically cleared by hardware.
0 = Acknowledge sequence idle

bit 3 **RCEN:** Receive Enable bit (in I^2C Master mode only)
1 = Enables Receive mode for I^2C
0 = Receive idle

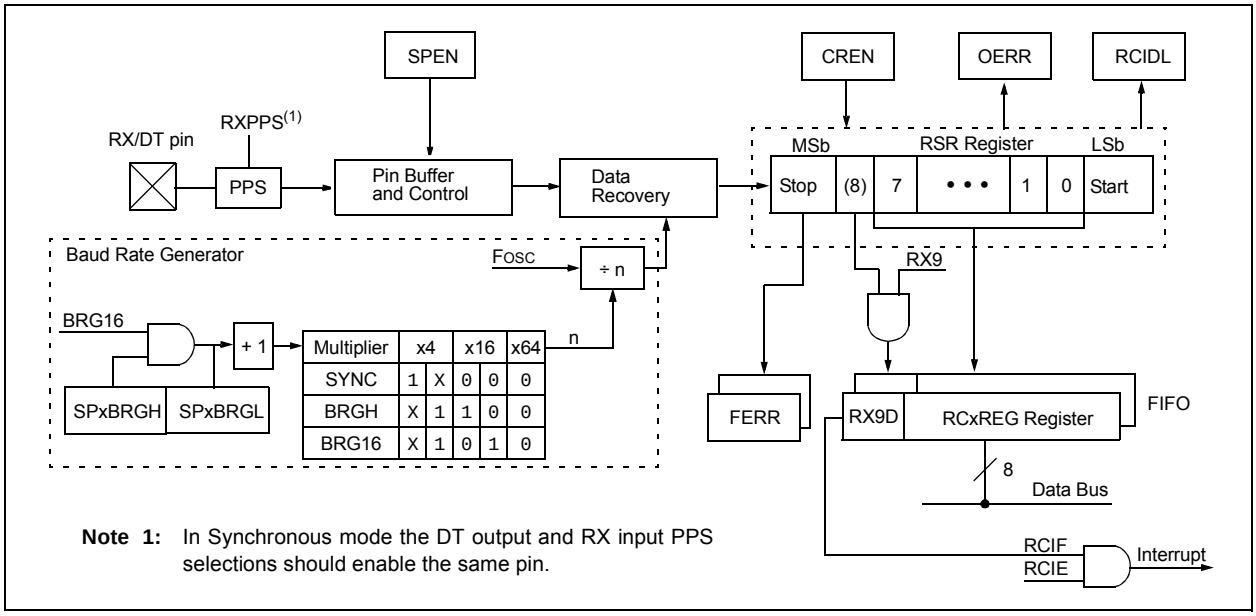
bit 2 **PEN:** Stop Condition Enable bit (in I^2C Master mode only)
SCKMSSP Release Control:
1 = Initiate Stop condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Stop condition Idle

bit 1 **RSEN:** Repeated Start Condition Enable bit (in I^2C Master mode only)
1 = Initiate Repeated Start condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Repeated Start condition Idle

bit 0 **SEN:** Start Condition Enable/Stretch Enable bit
In Master mode:
1 = Initiate Start condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Start condition Idle
In Slave mode:
1 = Clock stretching is enabled for both slave transmit and slave receive (stretch enabled)
0 = Clock stretching is disabled

Note 1: For bits ACKEN, RCEN, PEN, RSEN, SEN: If the I^2C module is not in the Idle mode, this bit may not be set (no spooling) and the SSP1BUF may not be written (or writes to the SSP1BUF are disabled).

FIGURE 25-2: EUSART RECEIVE BLOCK DIAGRAM



Note 1: In Synchronous mode the DT output and RX input PPS selections should enable the same pin.

The operation of the EUSART module is controlled through three registers:

- Transmit Status and Control (TXxSTA)
- Receive Status and Control (RCxSTA)
- Baud Rate Control (BAUDxCON)

These registers are detailed in Register 25-1, Register 25-2 and Register 25-3, respectively.

The RX and CK input pins are selected with the RXPPS and CKPPS registers, respectively. TX, CK, and DT output pins are selected with each pin's RxyPPS register. Since the RX input is coupled with the DT output in Synchronous mode, it is the user's responsibility to select the same pin for both of these functions when operating in Synchronous mode. The EUSART control logic will control the data direction drivers automatically.

FIGURE 30-14: TIME OF FLIGHT MODE REPEAT ACQUISITION TIMING DIAGRAM

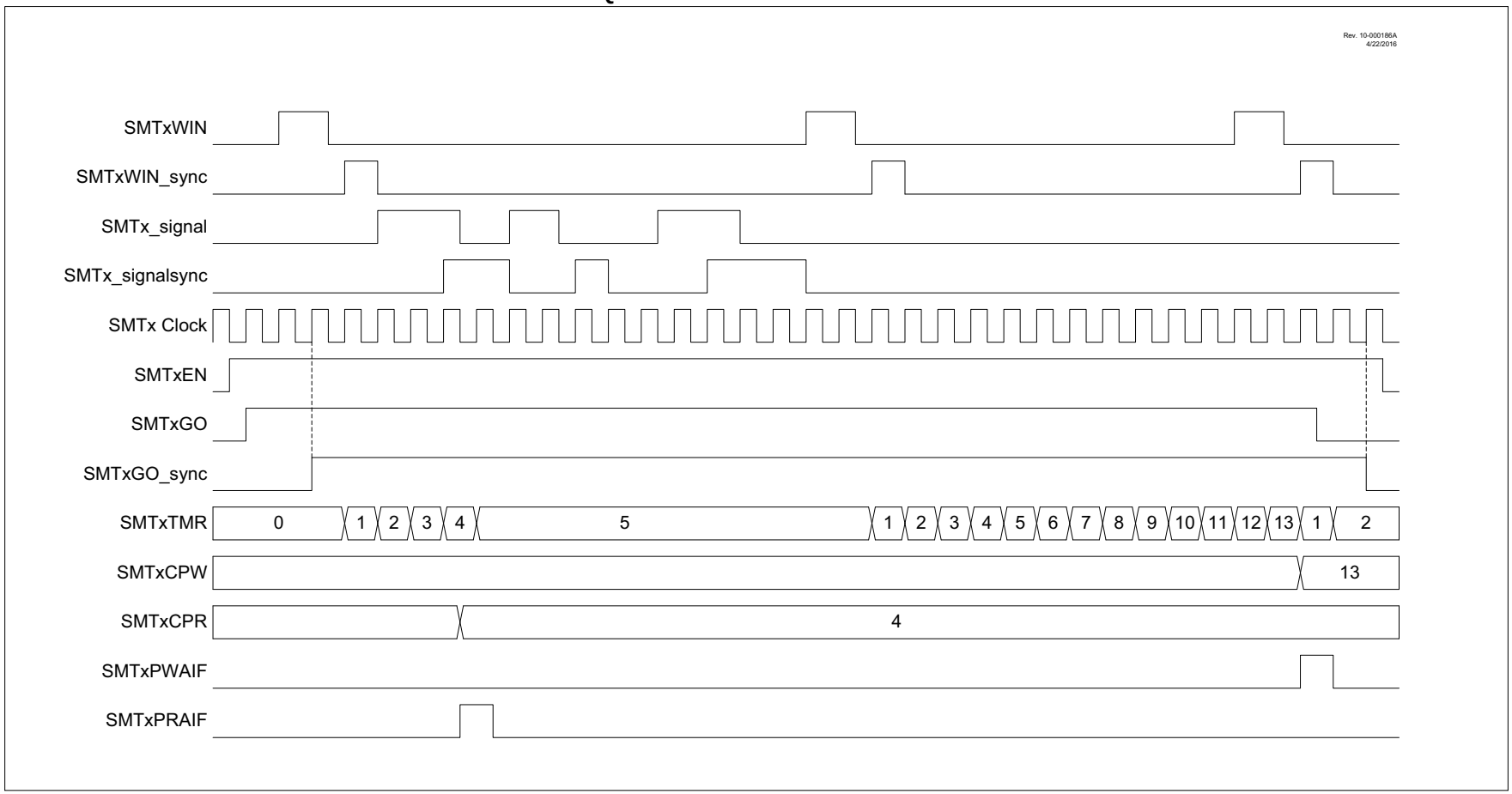
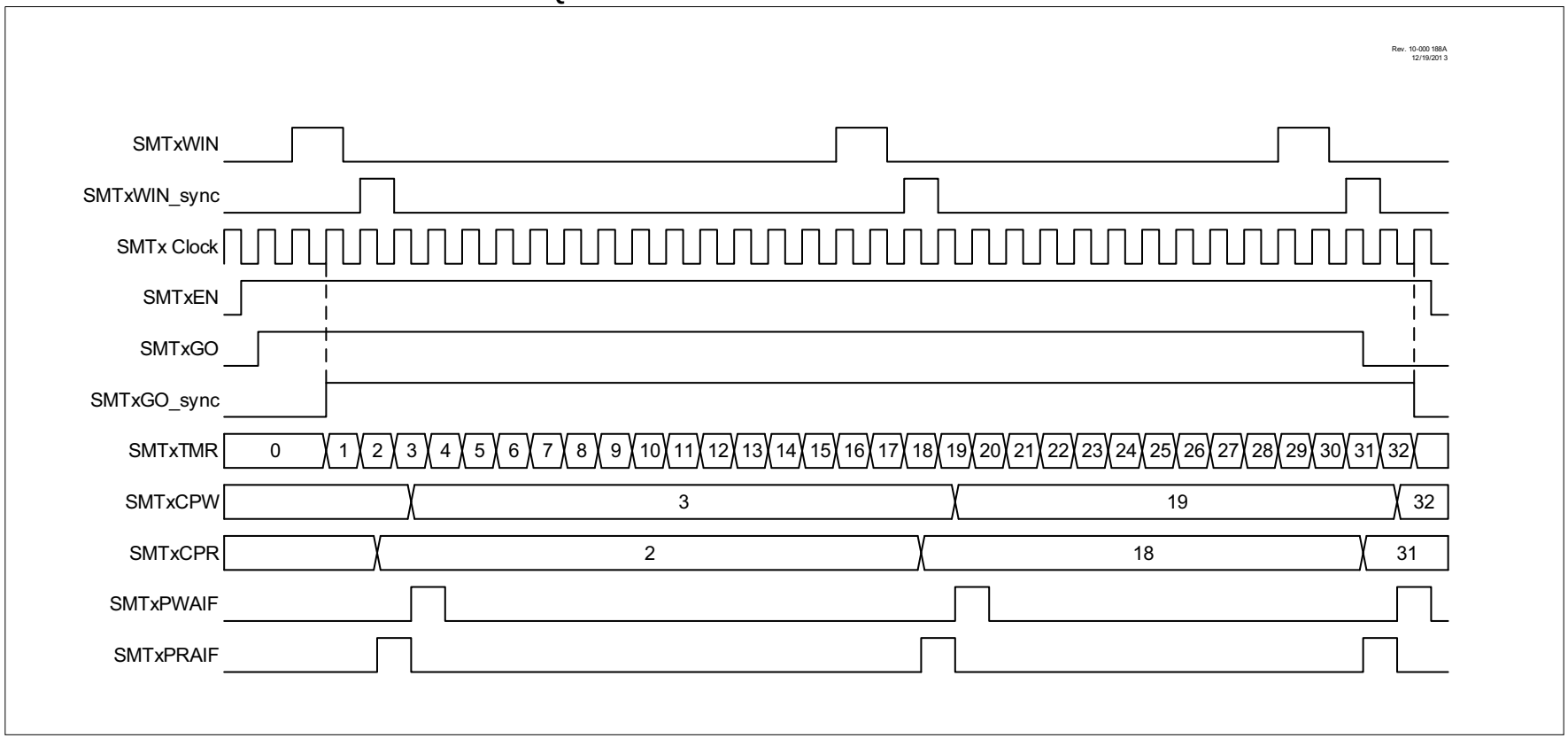


FIGURE 30-16: CAPTURE MODE REPEAT ACQUISITION TIMING DIAGRAM



31.2.1 SINGLE-PULSE MODE

The operation of Single-Pulse mode is illustrated in Figure 31-1. The calculations on the input signal are done in a few distinct steps. First, there is a divider that divides the module clock by the ATxRES register pair and uses the resulting signal to increment a period counter. This operation is expressed by Equation 31-2. This equation differs slightly from that of Equation 31-1 because the counters include the count of zero. To compensate for this, the number written to the resolution register, ATxRES, must be one less than the desired resolution.

EQUATION 31-2:

$$ATxPER = \frac{\frac{F(ATxclk)}{F(ATxsig)}}{(ATxRES + 1)}$$

Variables in Equation 31-2 are as follows:

- ATxPER is the value of the period counter latched by the input signal.
- ATxRES is the user-specified resolution. The phase counter will count up to this value.
- F(ATxclk) is the ATx clock frequency.
- F(ATxsig) is the input signal frequency.

The second step in the angular timer's operation is the creation of the phase clock, which is also illustrated in Figure 31-1. The input clock is divided by the ATxPER value, latched-in during the previous step, and the resulting signal is used to increment the phase counter. This signal also is used as the phase clock output, and for setting the PHSIF interrupt flag bit of the ATxIR0 register. The result is that the phase counter counts from zero to a final value expressed in Equation 31-3, outputting a pulse each time the counter increments. The value of the phase counter can be accessed by software by reading the ATxPHS register pair. However, because of the synchronization required, in order for reads of this register pair to be accurate, the instruction clock (FOSC/4) needs to be at least 3x the ATx_phsclk output frequency.

EQUATION 31-3:

$$ATxPHS(final) = \frac{\left(\frac{F(ATxclk)}{F(ATxsig)}\right)}{(ATxPER + 1)}$$

The variables in Equation 31-3 are as follows:

- ATxPHS(final) is the maximum value that the phase counter will reach before being reset by the input signal. As noted in Equation 31-1, this will equal ATxRES.
- ATxPER is the maximum value of the period counter.
- F(ATxclk) is the ATx clock frequency.
- F(ATxsig) is the input signal frequency.

Notice that the division is ATxPER + 1. Ideally, this would be just ATxPER but the divider includes zero in the count. In most applications, ATxPER is a large number so the error introduced by adding one is negligible.

ATxPHS counting from 0 to ATxRES is useful when the input signal represents a rotation (for example, a motor or A/C mains). In this case, the input signal is understood to provide a period pulse every 360 degrees. Since the phase clock equally divides the signal period into a number of intervals determined by the ATxRES register pair, each pulse on the phase clock output marks a fixed phase angle in that rotation, as expressed by Equation 31-4.

EQUATION 31-4:

$$AngleResolution = \frac{360degrees}{ATxRES + 1}$$

ATxRES can then be used with the instantaneous value of the ATxPHS register pair to get the instantaneous angle of the rotation using Equation 31-5.

EQUATION 31-5:

$$Angle = 360degrees \bullet \frac{ATxPHS}{ATxRES + 1}$$

31.2.2 MULTI-PULSE MODE

The operation of Multi-Pulse mode is illustrated in Figure 31-3. The calculations on the input signal are similar to those in Single-Pulse mode, with the primary difference relating to when the ATxPHS register pair is reset.

The period counter is latched into the ATxPER register pair and reset on every input pulse except the pulse immediately following a missing pulse. The first active pulse following a missing pulse triggers all of the following:

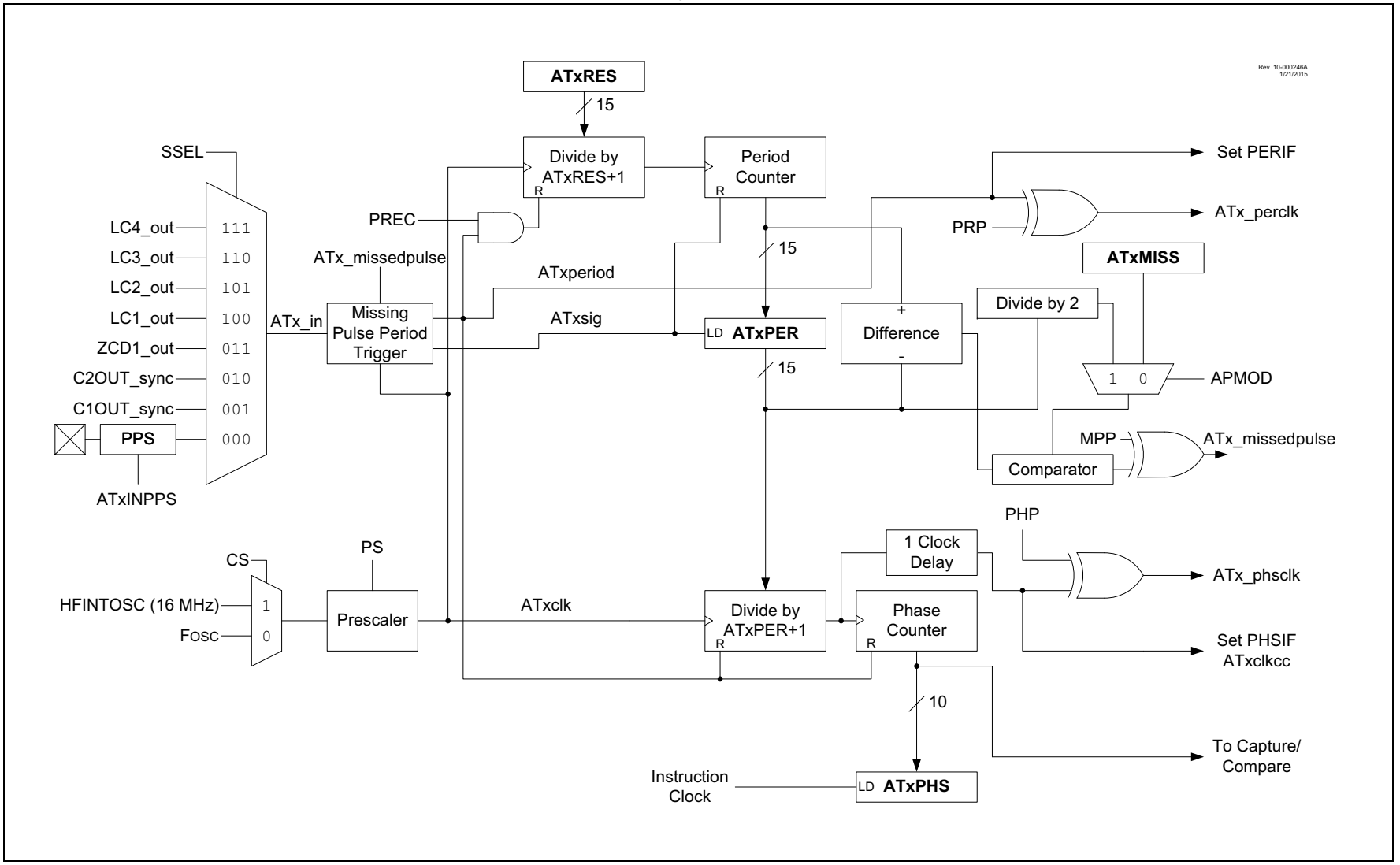
- Period clock output
- PERIF interrupt
- Phase counter reset

The result is a period clock output that has a period length equal to the time between missing pulses (e.g., a missing tooth in a gear). This leads to a significantly different relation between ATxRES and the maximum phase count, ATxPHS, as shown in Equation 31-6.

EQUATION 31-6:

$$ATxPHS(final) = ATxRES\left(\frac{MissP}{PulseP}\right)$$

FIGURE 31-2: ANGULAR TIMER SIMPLIFIED BLOCK DIAGRAM, MULTI-PULSE MODE



32.5 PID Control Registers

Long bit name prefixes for the 16-bit PID peripherals are shown in Table 32-1. Refer to **Section 1.1 "Register and Bit Naming Conventions"** for more information

TABLE 32-1:

Peripheral	Bit Name Prefix
PID1	PID1

REGISTER 32-1: PIDxCON: PID CONFIGURATION REGISTER

R/W-0/0	R/HS/HC-0/0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
EN	BUSY	—	—	—	MODE<2:0>		
bit 7							bit 0

Legend:		
HC = Bit is cleared by hardware		HS = Bit is set by hardware
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-m/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 **EN:** PID Module Enable bit
1 = PID module is enabled
0 = PID module is disabled

bit 6 **BUSY:** PID module is currently calculating

bit 5-3 **Unimplemented:** Read as '0'

bit 2-0 **MODE<2:0>:** PID Mode Control bits
11x = Reserved. Do not use.
101 = PID output is the calculated output (current error plus accumulated previous errors) in 2's complement notation
100 = Reserved. Do not use.
011 = (IN<15:0>+SET<15:0>)*K1<15:0> 2's complement signed inputs, with accumulation
010 = (IN<15:0>+SET<15:0>)*K1<15:0> 2's complement signed inputs, without accumulation
001 = (IN<15:0>+SET<15:0>)*K1<15:0> unsigned inputs, with accumulation
000 = (IN<15:0>+SET<15:0>)*K1<15:0> unsigned inputs, without accumulation

TABLE 34-3: ENHANCED MID-RANGE INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb			LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	f, d	Add W and f	1	00	0111	dfff	ffff	C, DC, Z	2
ADDWFC	f, d	Add with Carry W and f	1	11	1101	dfff	ffff	C, DC, Z	2
ANDWF	f, d	AND W with f	1	00	0101	dfff	ffff	Z	2
ASRF	f, d	Arithmetic Right Shift	1	11	0111	dfff	ffff	C, Z	2
LSLF	f, d	Logical Left Shift	1	11	0101	dfff	ffff	C, Z	2
LSRF	f, d	Logical Right Shift	1	11	0110	dfff	ffff	C, Z	2
CLRF	f	Clear f	1	00	0001	lfff	ffff	Z	2
CLRW	–	Clear W	1	00	0001	0000	00xx	Z	
COMF	f, d	Complement f	1	00	1001	dfff	ffff	Z	2
DECF	f, d	Decrement f	1	00	0011	dfff	ffff	Z	2
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z	2
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	2
MOVF	f, d	Move f	1	00	1000	dfff	ffff	Z	2
MOVWF	f	Move W to f	1	00	0000	1fff	ffff		2
RLF	f, d	Rotate Left f through Carry	1	00	1101	dfff	ffff	C	2
RRF	f, d	Rotate Right f through Carry	1	00	1100	dfff	ffff	C	2
SUBWF	f, d	Subtract W from f	1	00	0010	dfff	ffff	C, DC, Z	2
SUBWFB	f, d	Subtract with Borrow W from f	1	11	1011	dfff	ffff	C, DC, Z	2
SWAPF	f, d	Swap nibbles in f	1	00	1110	dfff	ffff		2
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	2
BYTE ORIENTED SKIP OPERATIONS									
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1, 2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1, 2
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff		2
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff		2
BIT-ORIENTED SKIP OPERATIONS									
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff		1, 2
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff		1, 2
LITERAL OPERATIONS									
ADDLW	k	Add literal and W	1	11	1110	kkkk	kkkk	C, DC, Z	
ANDLW	k	AND literal with W	1	11	1001	kkkk	kkkk	Z	
IORLW	k	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z	
MOVLB	k	Move literal to BSR	1	00	0000	001k	kkkk		
MOVLP	k	Move literal to PCLATH	1	11	0001	1kkk	kkkk		
MOVLW	k	Move literal to W	1	11	0000	kkkk	kkkk		
SUBLW	k	Subtract W from literal	1	11	1100	kkkk	kkkk	C, DC, Z	
XORLW	k	Exclusive OR literal with W	1	11	1010	kkkk	kkkk	Z	

Note 1: If the Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

2: If this instruction addresses an INDF register and the MSb of the corresponding FSR is set, this instruction will require one additional instruction cycle.

APPENDIX A: DATA SHEET REVISION HISTORY

Revision A (12/2014)

Original release.

Revision B (5/2016)

Minor typos corrected.

Added High endurance column to Table 1: PIC12/16(L)F161x Family Types.

Updated High-Endurance Flash data memory information on the cover page.

Updated Registers 19-2, 29-1 and 31-22. Section 19.6, 19.7. Updated Table 3: Pin Allocations Table and Table 5-1. Updated Figure 19-2.

Updated Package Drawings C04-127.