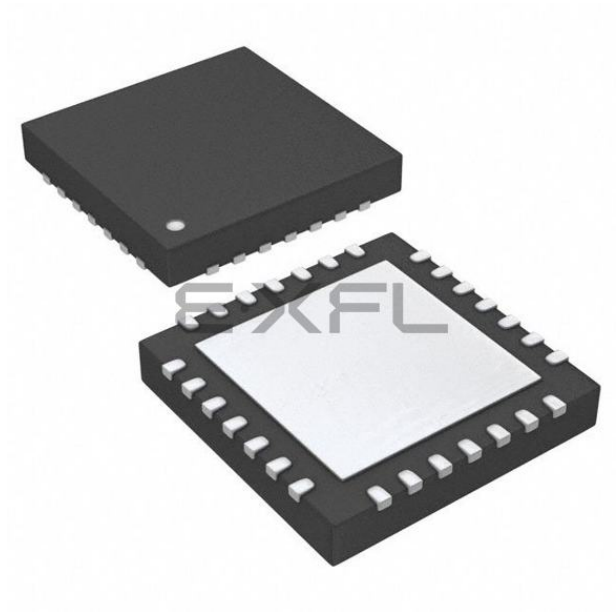


Welcome to E-XFL.COM

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	25
Program Memory Size	7KB (4K x 14)
Program Memory Type	FLASH
EEPROM Size	224 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 24x10b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-UFQFN Exposed Pad
Supplier Device Package	28-UQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf15354-i-mv

TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
Bank 7											
CPU CORE REGISTERS; see Table 4-3 for specifics											
38Ch	PWM6DCL	DC<1:0>		—	—	—	—	—	—	xx-- ----	uu-- ----
38Dh	PWM6DCH	DC<9:0>								xxxx xxxx	uuuu uuuu
38Eh	PWM6CON	EN	—	OUT	POL	—	—	—	—	0-00 ----	0-00 ----
38Fh — 39Fh	—	Unimplemented								—	—

Legend: x = unknown, u = unchanged, q = depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations unimplemented, read as '0'.

TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
Bank 20											
CPU CORE REGISTERS; see Table 4-3 for specifics											
A0Ch — A18h	—	Unimplemented								—	—
A19h	RC2REG	RC2REG<7:0>								0000 0000	0000 0000
A1Ah	TX2REG	TX2REG<7:0>								0000 0000	0000 0000
A1Bh	SP2BRGL	SP2BRGL<7:0>								0000 0000	0000 0000
A1Ch	SP2BRGH	SP2BRGH<7:0>								0000 0000	0000 0000
A1Dh	RC2STA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 0000	0000 0000
A1Eh	TX2STA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
A1Fh	BAUD2CON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-0 0-00	01-0 0-00

Legend: x = unknown, u = unchanged, q = depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations unimplemented, read as '0'.

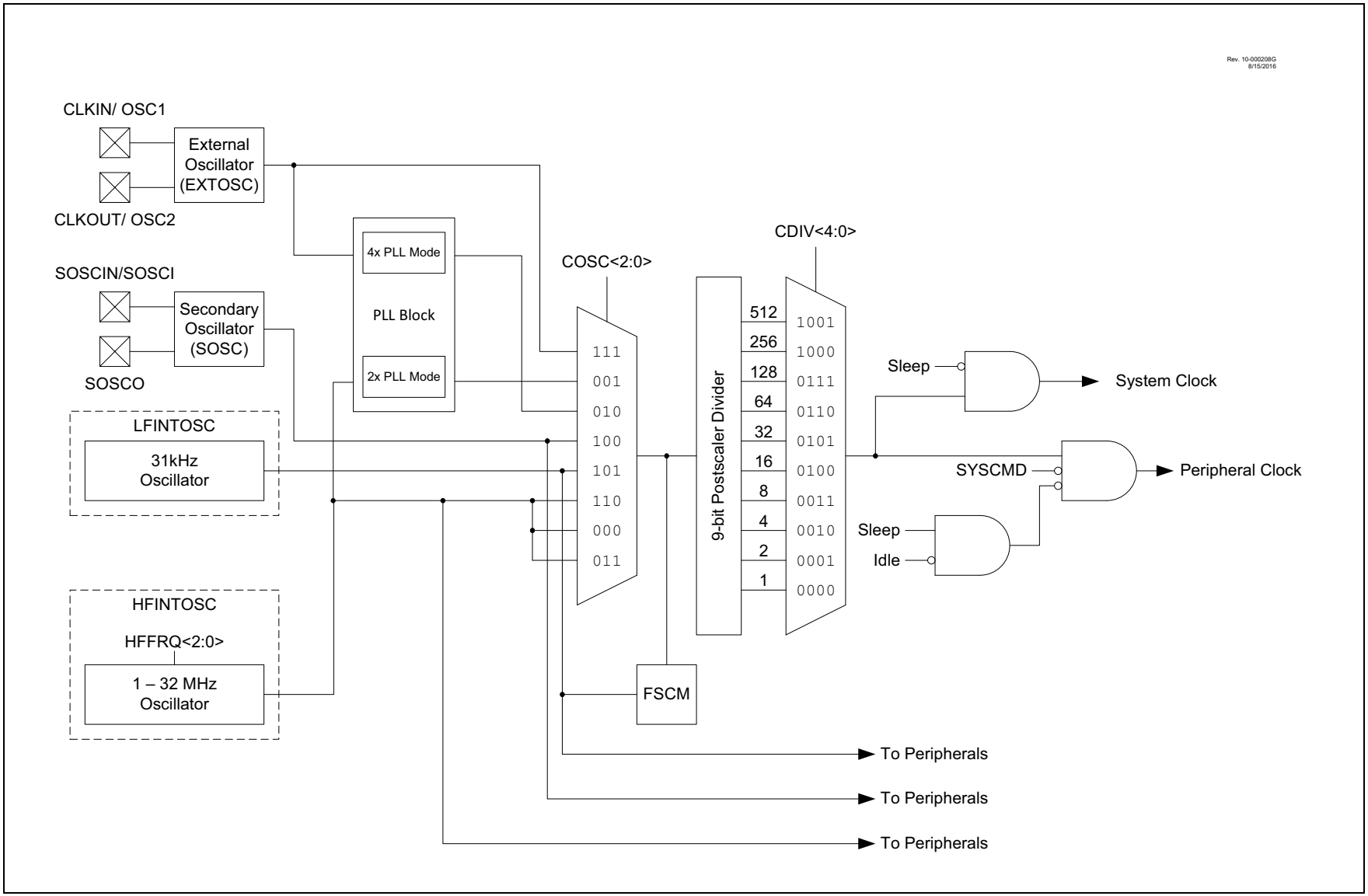
8.12 Memory Execution Violation

A Memory Execution Violation Reset occurs if executing an instruction being fetched from outside the valid execution area. The different valid execution areas are defined as follows:

- Flash Memory: Table 4-1 shows the addresses available on the PIC16(L)F15354/55 devices based on user flash size. Execution outside this region generates a memory execution violation.
- Storage Area Flash (SAF): If Storage Area Flash (SAF) is enabled (**Section 4.2.3 “Storage Area Flash”**), the SAF area (Table 4-2) is not a valid execution area.

Prefetched instructions that are not executed do not cause memory execution violations. For example, a GOTO instruction in the last memory location will prefetch from an invalid location; this is not an error. If an instruction from an invalid location tries to execute, the memory violation is generated immediately, and any concurrent interrupt requests are ignored. When a memory execution violation is generated, the device is reset and flag $\overline{\text{MEMV}}$ is cleared in PCON1 (Register 8-3) to signal the cause. The flag needs to be set in code after a memory execution violation

FIGURE 9-1: SIMPLIFIED PIC® MCU CLOCK SOURCE BLOCK DIAGRAM



10.0 INTERRUPTS

The interrupt feature allows certain events to preempt normal program flow. Firmware is used to determine the source of the interrupt and act accordingly. Some interrupts can be configured to wake the MCU from Sleep mode.

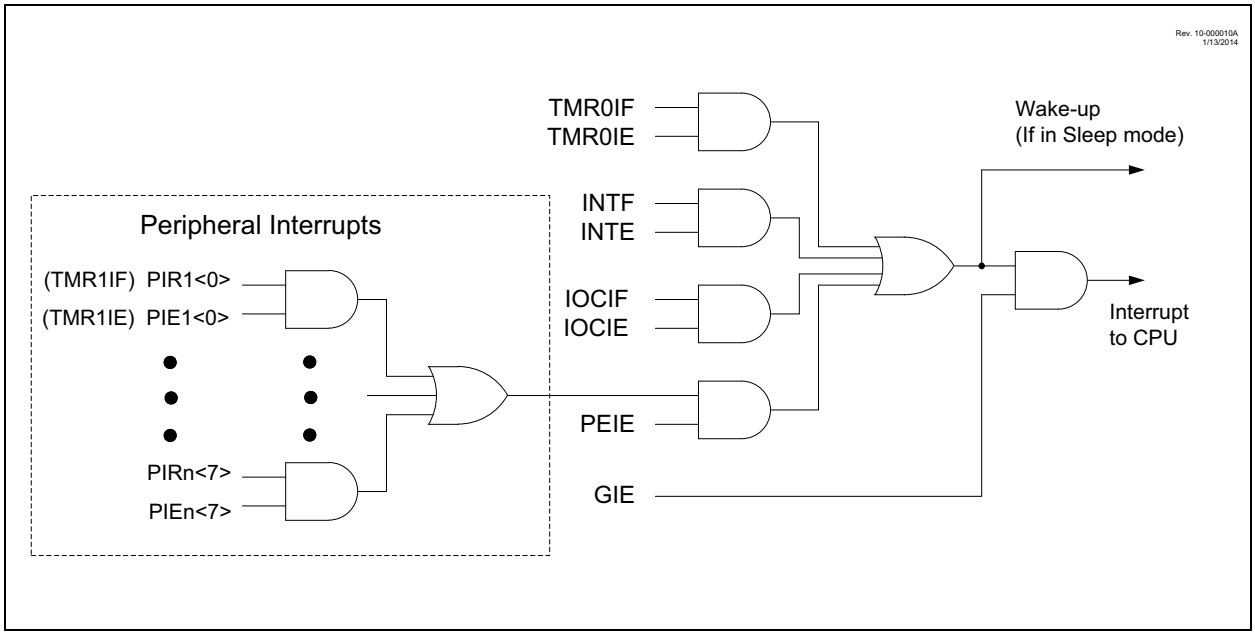
This chapter contains the following information for Interrupts:

- Operation
- Interrupt Latency
- Interrupts During Sleep
- INT Pin
- Automatic Context Saving

Many peripherals produce interrupts. Refer to the corresponding chapters for details.

A block diagram of the interrupt logic is shown in Figure 10-1.

FIGURE 10-1: INTERRUPT LOGIC



REGISTER 10-15: PIR5: PERIPHERAL INTERRUPT REQUEST REGISTER 5

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	U-0	U-0	U-0	R/W/HS-0/0
CLC4IF	CLC3IF	CLC2IF	CLC1IF	—	—	—	TMR1GIF
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 = Hardware set

bit 7 **CLC4IF:** CLC4 Interrupt Flag bit
1 = A CLC4OUT interrupt condition has occurred (must be cleared in software)
0 = No CLC4 interrupt event has occurred

bit 6 **CLC3IF:** CLC3 Interrupt Flag bit
1 = A CLC3OUT interrupt condition has occurred (must be cleared in software)
0 = No CLC3 interrupt event has occurred

bit 5 **CLC2IF:** CLC2 Interrupt Flag bit
1 = A CLC2OUT interrupt condition has occurred (must be cleared in software)
0 = No CLC2 interrupt event has occurred

bit 4 **CLC1IF:** CLC1 Interrupt Flag bit
1 = A CLC1OUT interrupt condition has occurred (must be cleared in software)
0 = No CLC1 interrupt event has occurred

bit 3-1 **Unimplemented:** Read as ‘0’

bit 0 **TMR1GIF:** Timer1 Gate Interrupt Flag bit
1 = The Timer1 Gate has gone inactive (the acquisition is complete)
0 = The Timer1 Gate has not gone inactive

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

REGISTER 14-15: SLRCONB: PORTB SLEW RATE CONTROL 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
SLRB7	SLRB6	SLRB5	SLRB4	SLRB3	SLRB2	SLRB1	SLRB0
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 **SLRB<7:0>:** PORTB Slew Rate Enable bits
For RB<7:0> pins, respectively
1 = Port pin slew rate is limited
0 = Port pin slews at maximum rate

REGISTER 14-16: INLVLB: PORTB INPUT LEVEL CONTROL 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
INLVLB7	INLVLB6	INLVLB5	INLVLB4	INLVLB3	INLVLB2	INLVLB1	INLVLB0
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 **INLVLB<7:0>:** PORTB Input Level Select bits
For RB<7:0> pins, respectively
1 = ST input used for PORT reads and interrupt-on-change
0 = TTL input used for PORT reads and interrupt-on-change

TABLE 14-3: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	178
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	178
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	179
ANSELB	ANSB7	ANSB6	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	179
WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	180
ODCONB	ODCB7	ODCB6	ODCB5	ODCB4	ODCB3	ODCB2	ODCB1	ODCB0	180
SLRCONB	SLRB7	SLRB6	SLRB5	SLRB4	SLRB3	SLRB2	SLRB1	SLRB0	181
INLVLB	INLVLB7	INLVLB6	INLVLB5	INLVLB4	INLVLB3	INLVLB2	INLVLB1	INLVLB0	181

Legend: x = unknown, u = unchanged, – = unimplemented locations read as '0'. Shaded cells are not used by PORTB.

Note 1: Unimplemented, read as '1'.

14.8 PORTE Registers

14.8.1 DATA REGISTER

PORTE is a single bit wide port. The corresponding data direction register is TRISE (Register 14-25). Setting a TRISE bit (= 1) will make the corresponding PORTE pin an input (i.e., disable the output driver). Clearing a TRISE bit (= 0) will make the corresponding PORTE pin an output (i.e., enables output driver and puts the contents of the output latch on the selected pin). Figure 14-1 shows how to initialize PORTE.

Reading the PORTE register (Register 14-25) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATE).

14.8.2 DIRECTION CONTROL

The TRISE register (Register 14-26) controls the PORTE pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISE register are maintained set when using them as analog inputs. I/O pins configured as analog inputs always read '0'.

Note:	The TRISE3 bit is a read-only bit and it always reads a '1'.

14.8.3 INPUT THRESHOLD CONTROL

The INLVLE register (Register 14-28) controls the input voltage threshold for each of the available PORTE input pins. A selection between the Schmitt Trigger CMOS or the TTL Compatible thresholds is available. The input threshold is important in determining the value of a read of the PORTE register and also the level at which an interrupt-on-change occurs, if that feature is enabled. See Table 37-4 for more information on threshold levels.

Note:	Changing the input threshold selection should be performed while all peripheral modules are disabled. Changing the threshold level during the time a module is active may inadvertently generate a transition associated with an input pin, regardless of the actual voltage level on that pin.

14.8.4 WEAK PULL-UP CONTROL

The WPUE register (Register 14-27) controls the individual weak pull-ups for each port pin.

14.8.5 PORTE FUNCTIONS AND OUTPUT PRIORITIES

Each pin defaults to the PORT latch data after Reset. Other output functions are selected with the peripheral pin select logic. See **Section 15.0 "Peripheral Pin Select (PPS) Module"** for more information.

Analog input functions, such as ADC and comparator inputs, are not shown in the peripheral pin select lists. Digital output functions may continue to control the pin when it is in Analog mode.

TABLE 27-1: TIMER2 OPERATING MODES

Mode	MODE<4:0>		Output Operation	Operation	Timer Control		
	<4:3>	<2:0>			Start	Reset	Stop
Free Running Period	00	000	Period Pulse	Software gate (Figure 27-4)	ON = 1	—	ON = 0
		001		Hardware gate, active-high (Figure 27-5)	ON = 1 and TMRx_ers = 1	—	ON = 0 or TMRx_ers = 0
		010		Hardware gate, active-low	ON = 1 and TMRx_ers = 0	—	ON = 0 or TMRx_ers = 1
		011	Period Pulse with Hardware Reset	Rising or falling edge Reset	ON = 1	TMRx_ers ↕	ON = 0
		100		Rising edge Reset (Figure 27-6)		TMRx_ers ↑	
		101		Falling edge Reset		TMRx_ers ↓	
		110		Low level Reset		TMRx_ers = 0	ON = 0 or TMRx_ers = 0
		111		High level Reset (Figure 27-7)		TMRx_ers = 1	ON = 0 or TMRx_ers = 1
One-shot	01	000	One-shot	Software start (Figure 27-8)	ON = 1	—	ON = 0 or Next clock after TMRx = PRx (**Note 2**)
		001	Edge triggered start (**Note 1**)	Rising edge start (Figure 27-9)	ON = 1 and TMRx_ers ↑	—	
		010		Falling edge start	ON = 1 and TMRx_ers ↓	—	
		011		Any edge start	ON = 1 and TMRx_ers ↕	—	
		100	Edge triggered start and hardware Reset (**Note 1**)	Rising edge start and Rising edge Reset (Figure 27-10)	ON = 1 and TMRx_ers ↑	TMRx_ers ↑	
		101		Falling edge start and Falling edge Reset	ON = 1 and TMRx_ers ↓	TMRx_ers ↓	
		110		Rising edge start and Low level Reset (Figure 27-11)	ON = 1 and TMRx_ers ↑	TMRx_ers = 0	
		111		Falling edge start and High level Reset	ON = 1 and TMRx_ers ↓	TMRx_ers = 1	
Mono-stable	10	000	Reserved				
		001	Edge triggered start (**Note 1**)	Rising edge start (Figure 27-12)	ON = 1 and TMRx_ers ↑	—	ON = 0 or Next clock after TMRx = PRx (**Note 3**)
		010		Falling edge start	ON = 1 and TMRx_ers ↓	—	
		011		Any edge start	ON = 1 and TMRx_ers ↕	—	
Reserved		100	Reserved				
Reserved		101	Reserved				
One-shot		110	Level triggered start and hardware Reset	High level start and Low level Reset (Figure 27-13)	ON = 1 and TMRx_ers = 1	TMRx_ers = 0	ON = 0 or Held in Reset (**Note 2**)
		111		Low level start & High level Reset	ON = 1 and TMRx_ers = 0	TMRx_ers = 1	
Reserved	11	xxx	Reserved				

Note 1: If ON = 0 then an edge is required to restart the timer after ON = 1.

2: When TMRx = PRx then the next clock clears ON and stops TMRx at 00h.

3: When TMRx = PRx then the next clock stops TMRx at 00h but does not clear ON.

REGISTER 28-6: CCPTMRS1: CCP TIMERS CONTROL 1 REGISTER

U-0	U-0	U-0	U-0	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1
—	—	—	—	P2TSEL<1:0>		C1TSEL<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 Reset
‘1’ = Bit is set	‘0’ = Bit is cleared	

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

bit 3-2 **P2TSEL<1:0>:** PWM2 Timer Selection
11 = PWM2 based on TMR2
10 = PWM2 based on TMR2
01 = PWM2 based on TMR2
00 = Reserved

bit 1-0 **C1TSEL<1:0>:** CCP1 Timer Selection
11 = CCP1 based on TMR1 (Capture/Compare) or TMR2 (PWM)
10 = CCP1 based on TMR1 (Capture/Compare) or TMR2 (PWM)
01 = CCP1 based on TMR1 (Capture/Compare) or TMR2 (PWM)
00 = Reserved

FIGURE 30-3: SIMPLIFIED CWG BLOCK DIAGRAM (FORWARD AND REVERSE FULL-BRIDGE MODES)

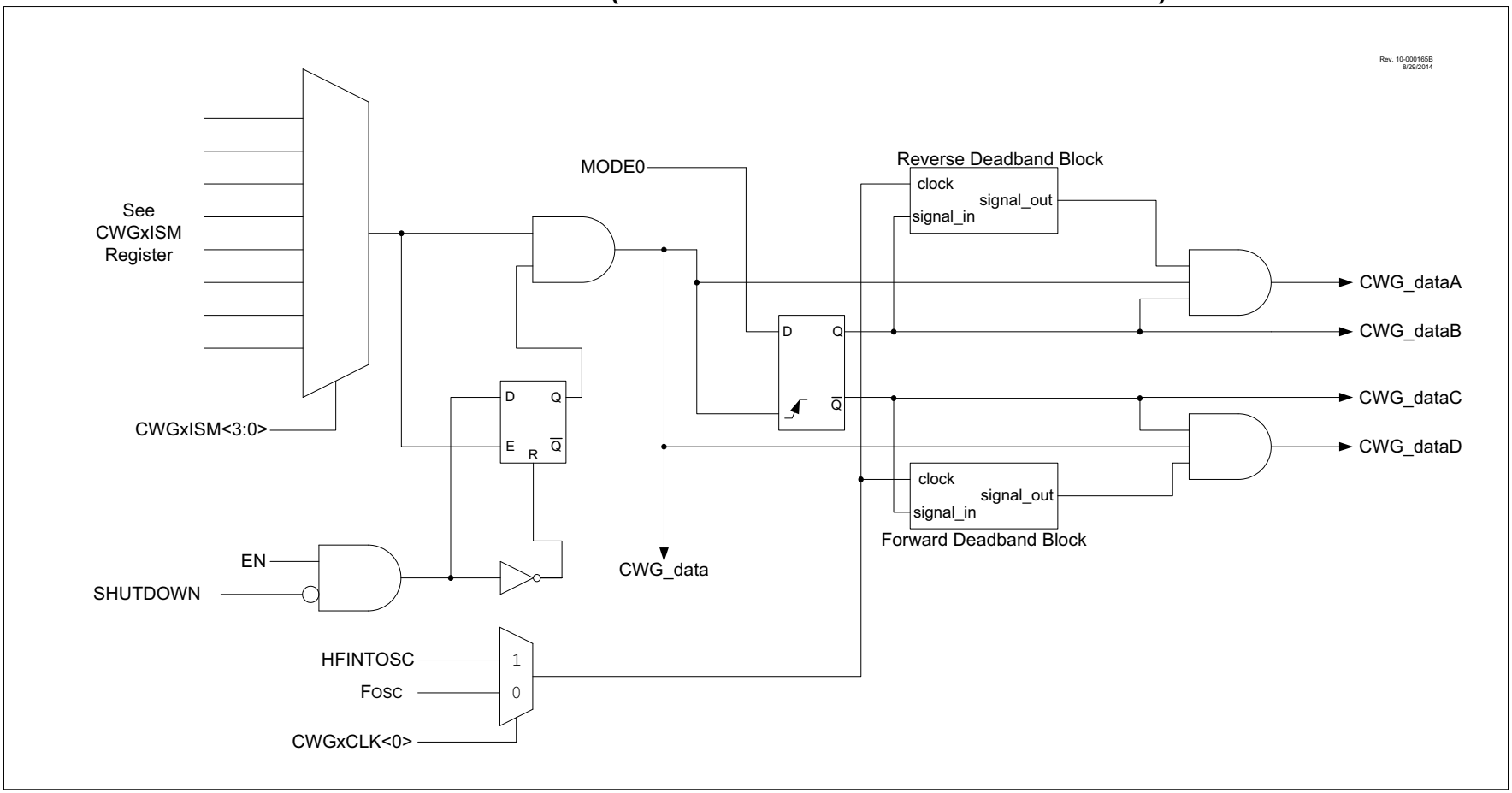


FIGURE 31-2: INPUT DATA SELECTION AND GATING

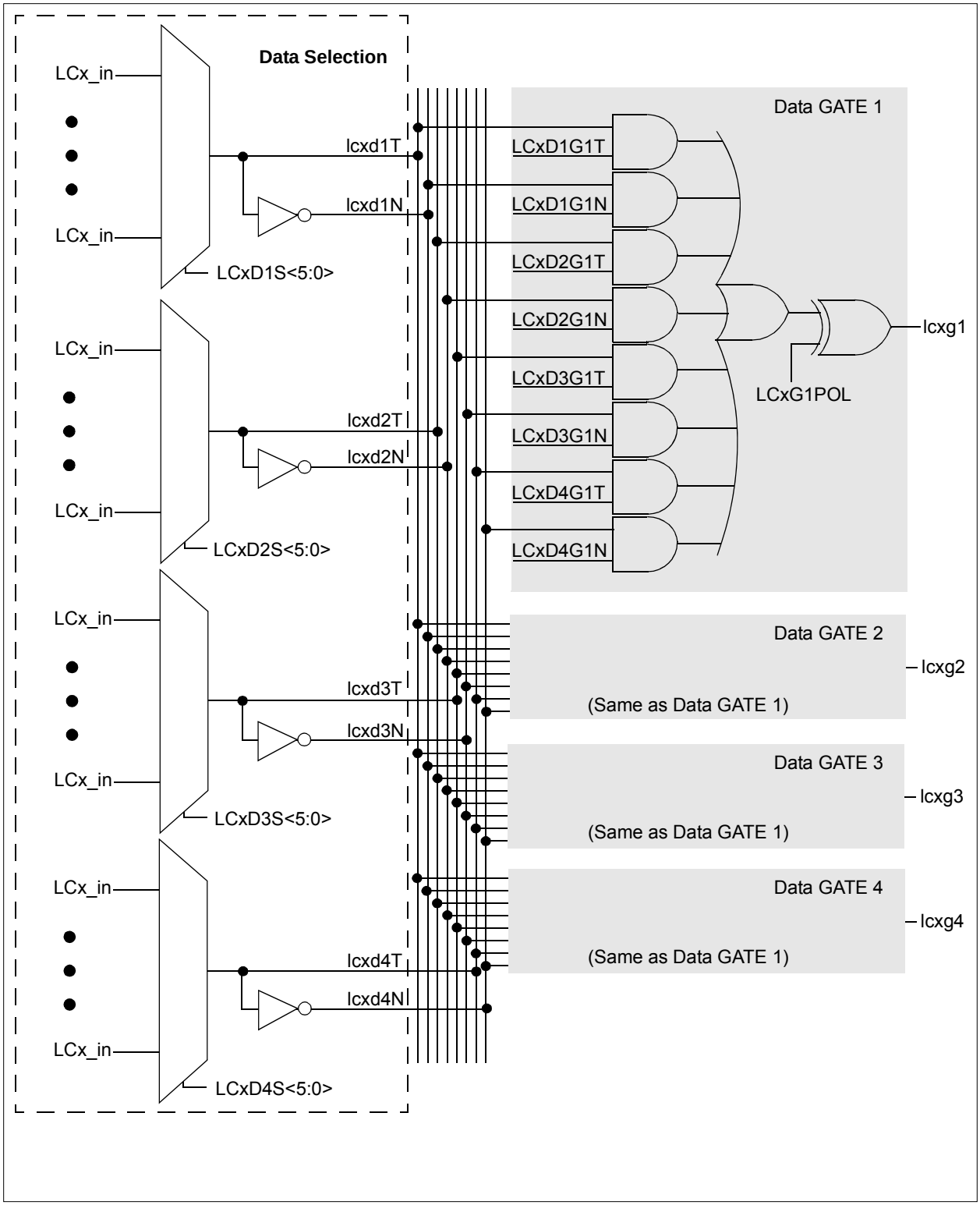


FIGURE 32-12: I²C START AND STOP CONDITIONS

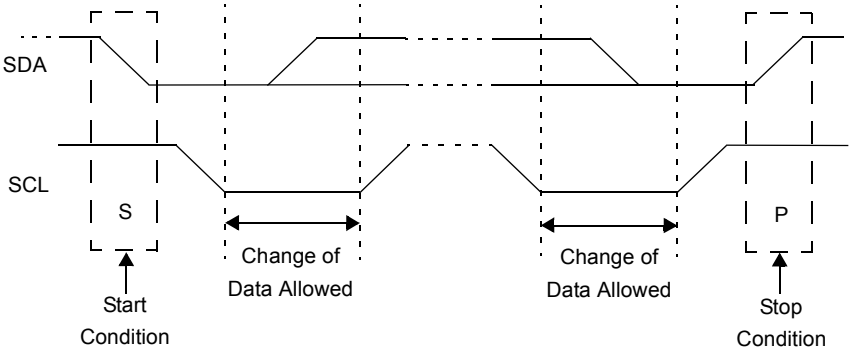
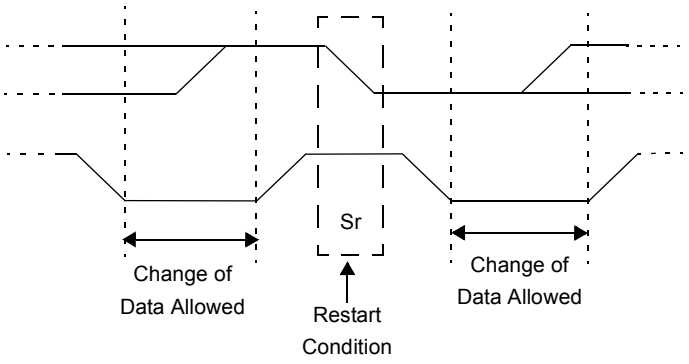


FIGURE 32-13: I²C RESTART CONDITION



32.4.9 ACKNOWLEDGE SEQUENCE

The 9th SCL pulse for any transferred byte in I²C is dedicated as an Acknowledge. It allows receiving devices to respond back to the transmitter by pulling the SDA line low. The transmitter must release control of the line during this time to shift in the response. The Acknowledge ($\overline{ACK}$) is an active-low signal, pulling the SDA line low indicates to the transmitter that the device has received the transmitted data and is ready to receive more.

The result of an $\overline{ACK}$ is placed in the ACKSTAT bit of the SSPxCON2 register.

Slave software, when the AHEN and DHEN bits are set, allow the user to set the $\overline{ACK}$ value sent back to the transmitter. The ACKDT bit of the SSPxCON2 register is set/cleared to determine the response.

There are certain conditions where an $\overline{ACK}$ will not be sent by the slave. If the BF bit of the SSPxSTAT register or the SSPOV bit of the SSPxCON1 register are set when a byte is received.

When the module is addressed, after the eighth falling edge of SCL on the bus, the ACKTIM bit of the SSPxCON3 register is set. The ACKTIM bit indicates the acknowledge time of the active bus. The ACKTIM Status bit is only active when the AHEN bit or DHEN bit is enabled.

32.5.3.3 7-bit Transmission with Address Hold Enabled

Setting the AHEN bit of the SSPxCON3 register enables additional clock stretching and interrupt generation after the eighth falling edge of a received matching address. Once a matching address has been clocked in, CKP is cleared and the SSPxIF interrupt is set.

Figure 32-19 displays a standard waveform of a 7-bit address slave transmission with AHEN enabled.

1. Bus starts Idle.
2. Master sends Start condition; the S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Master sends matching address with R/$\overline{W}$ bit set. After the eighth falling edge of the SCL line the CKP bit is cleared and SSPxIF interrupt is generated.
4. Slave software clears SSPxIF.
5. Slave software reads ACKTIM bit of SSPxCON3 register, and R/$\overline{W}$ and D/$\overline{A}$ of the SSPxSTAT register to determine the source of the interrupt.
6. Slave reads the address value from the SSPxBUF register clearing the BF bit.
7. Slave software decides from this information if it wishes to $\overline{ACK}$ or not $\overline{ACK}$ and sets the ACKDT bit of the SSPxCON2 register accordingly.
8. Slave sets the CKP bit releasing SCL.
9. Master clocks in the $\overline{ACK}$ value from the slave.
10. Slave hardware automatically clears the CKP bit and sets SSPxIF after the $\overline{ACK}$ if the R/$\overline{W}$ bit is set.
11. Slave software clears SSPxIF.
12. Slave loads value to transmit to the master into SSPxBUF setting the BF bit.

Note: SSPxBUF cannot be loaded until after the $\overline{ACK}$.

13. Slave sets the CKP bit releasing the clock.
14. Master clocks out the data from the slave and sends an $\overline{ACK}$ value on the ninth SCL pulse.
15. Slave hardware copies the $\overline{ACK}$ value into the ACKSTAT bit of the SSPxCON2 register.
16. Steps 10-15 are repeated for each byte transmitted to the master from the slave.
17. If the master sends a not $\overline{ACK}$ the slave releases the bus allowing the master to send a Stop and end the communication.

Note: Master must send a not $\overline{ACK}$ on the last byte to ensure that the slave releases the SCL line to receive a Stop.

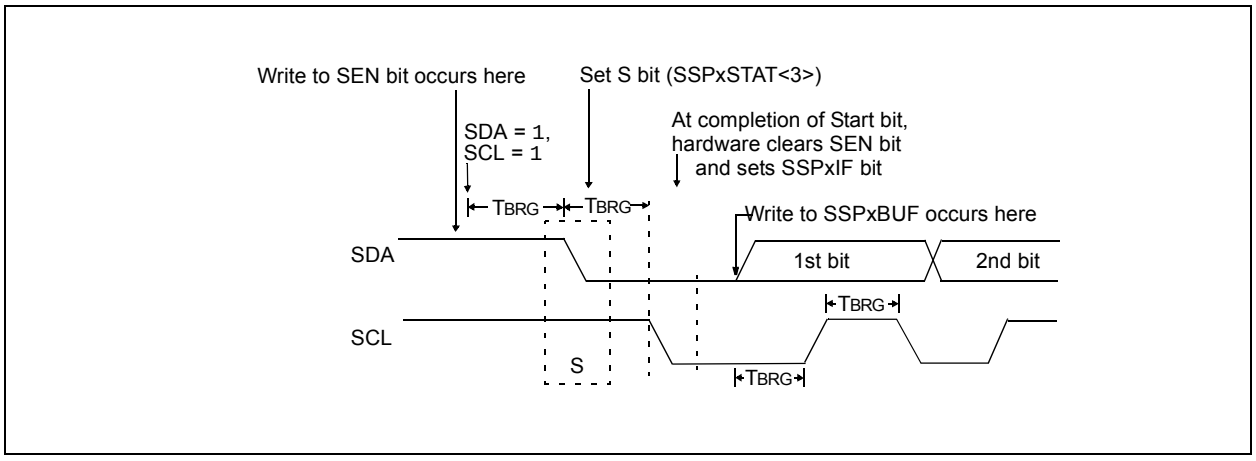
32.6.4 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition (Figure 32-26), the user sets the Start Enable bit, SEN bit of the SSPxCON2 register. If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low while SCL is high is the Start condition and causes the S bit of the SSPxSTAT1 register to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit of the SSPxCON2 register will be automatically cleared by hardware; the Baud Rate Generator is suspended, leaving the SDA line held low and the Start condition is complete.

Note 1: If at the beginning of the Start condition, the SDA and SCL pins are already sampled low, or if during the Start condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

2: The Philips I²C specification states that a bus collision cannot occur on a Start.

FIGURE 32-26: FIRST START BIT TIMING



REGISTER 32-3: SSPxCON2: SSPx CONTROL REGISTER 2 (I^2C MODE ONLY)(1)

R/W-0/0	R/HS/HC-0	R/W-0/0	R/S/HC-0/0	R/S/HC-0/0	R/S/HC-0/0	R/S/HC-0/0	R/S/HC-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 SSPxSR
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 SSPxBUF may not be written (or writes to the SSPxBUF are disabled).

33.1.1.5 TSR Status

The TRMT bit of the TXxSTA register indicates the status of the TSR register. This is a read-only bit. The TRMT bit is set when the TSR register is empty and is cleared when a character is transferred to the TSR register from the TXxREG. The TRMT bit remains clear until all bits have been shifted out of the TSR register. No interrupt logic is tied to this bit, so the user has to poll this bit to determine the TSR status.

Note:	The TSR register is not mapped in data memory, so it is not available to the user.

33.1.1.6 Transmitting 9-Bit Characters

The EUSART supports 9-bit character transmissions. When the TX9 bit of the TXxSTA register is set, the EUSART will shift nine bits out for each character transmitted. The TX9D bit of the TXxSTA register is the ninth, and Most Significant data bit. When transmitting 9-bit data, the TX9D data bit must be written before writing the eight Least Significant bits into the TXxREG. All nine bits of data will be transferred to the TSR shift register immediately after the TXxREG is written.

A special 9-bit Address mode is available for use with multiple receivers. See **Section 33.1.2.7 "Address Detection"** for more information on the Address mode.

33.1.1.7 Asynchronous Transmission Set-up:

1. Initialize the SPxBRGH, SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see **Section 33.3 "EUSART Baud Rate Generator (BRG)"**).
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If 9-bit transmission is desired, set the TX9 control bit. A set ninth data bit will indicate that the eight Least Significant data bits are an address when the receiver is set for address detection.
4. Set SCKP bit if inverted transmit is desired.
5. Enable the transmission by setting the TXEN control bit. This will cause the TXxIF interrupt bit to be set.
6. If interrupts are desired, set the TXxIE interrupt enable bit of the PIE3 register. An interrupt will occur immediately provided that the GIE and PEIE bits of the INTCON register are also set.
7. If 9-bit transmission is selected, the ninth bit should be loaded into the TX9D data bit.
8. Load 8-bit data into the TXxREG register. This will start the transmission.

FIGURE 33-3: ASYNCHRONOUS TRANSMISSION

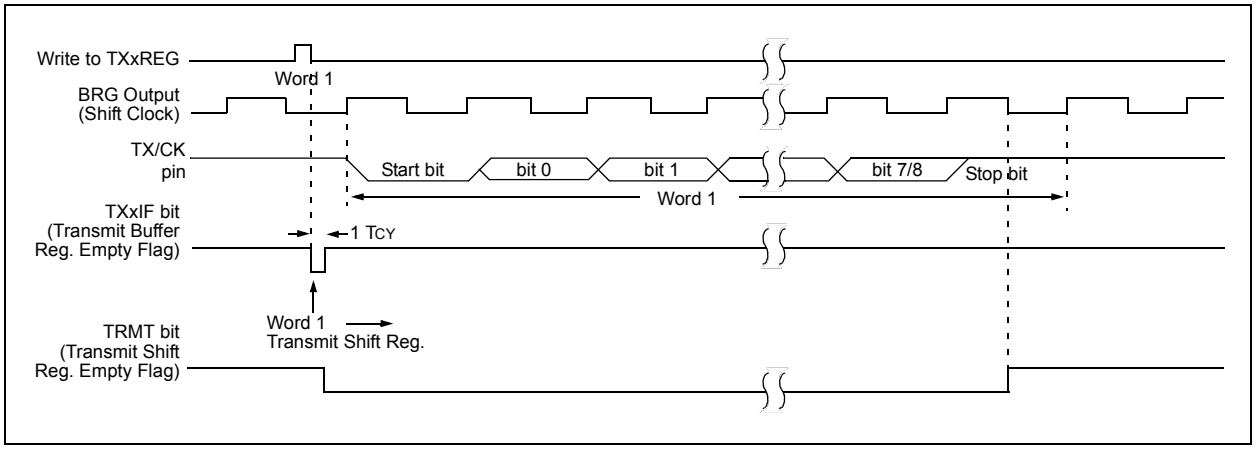


TABLE 33-2: SUMMARY OF REGISTERS ASSOCIATED WITH EUSART

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	119
PIR3	RC2IF	TX2IF	RC1IF	TX1IF	BCL2IF	SSP2IF	BCL1IF	SSP1IF	131
PIE3	RC2IE	TX2IE	RC1IE	TX1IE	BCL2IE	SSP2IE	BCL1IE	SSP1IE	123
RCxSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	442
TXxSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	441
BAUDxCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	443
RCxREG	EUSART Receive Data Register								444*
TXxREG	EUSART Transmit Data Register								444*
SPxBRGL	SPxBRG<7:0>								444*
SPxBRGH	SPxBRG<15:8>								445*
RXPPS	—	—	RXPPS<5:0>						196
CKPPS	—	—	CXPPS<5:0>						196
RxyPPS	—	—	—	RxyPPS<4:0>					197
CLCxSELy	—	—	LCxDyS<5:0>						363

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for the EUSART module.

* Page with register information.

DECFSZ **Decrement f, Skip if 0**

Syntax:	[*label*] DECFSZ f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) - 1 → (destination); skip if result = 0
Status Affected:	None
Description:	The contents of register 'f' are decremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', then a NOP is executed instead, making it a 2-cycle instruction.

GOTO **Unconditional Branch**

Syntax:	[*label*] GOTO k
Operands:	$0 \le k \le 2047$
Operation:	k → PC<10:0> PCLATH<6:3> → PC<14:11>
Status Affected:	None
Description:	GOTO is an unconditional branch. The 11-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a 2-cycle instruction.

INCF **Increment f**

Syntax:	[*label*] INCF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) + 1 → (destination)
Status Affected:	Z
Description:	The contents of register 'f' are incremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'.

INCFSZ **Increment f, Skip if 0**

Syntax:	[*label*] INCFSZ f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) + 1 → (destination), skip if result = 0
Status Affected:	None
Description:	The contents of register 'f' are incremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', a NOP is executed instead, making it a 2-cycle instruction.

IORLW **Inclusive OR literal with W**

Syntax:	[*label*] IORLW k
Operands:	$0 \le k \le 255$
Operation:	(W) .OR. k → (W)
Status Affected:	Z
Description:	The contents of the W register are OR'ed with the 8-bit literal 'k'. The result is placed in the W register.

IORWF **Inclusive OR W with f**

Syntax:	[*label*] IORWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) .OR. (f) → (destination)
Status Affected:	Z
Description:	Inclusive OR the W register with register 'f'. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'.

TABLE 37-2: SUPPLY CURRENT (I_{DD})(1,2,4)

PIC16LF15354/55			Standard Operating Conditions (unless otherwise stated)					
PIC16F15354/55								
Param. No.	Symbol	Device Characteristics	Min.	Typ.†	Max.	Units	Conditions	
							V_{DD}	Note
D100	$I_{DD_{XT4}}$	XT = 4 MHz	—	360	600	µA	3.0V	
D100	$I_{DD_{XT4}}$	XT = 4 MHz	—	380	700	µA	3.0V	
D101	$I_{DD_{HFO16}}$	HFINTOSC = 16 MHz	—	1.4	2.0	mA	3.0V	
D101	$I_{DD_{HFO16}}$	HFINTOSC = 16 MHz	—	1.5	2.1	mA	3.0V	
D102	$I_{DD_{HFOPLL}}$	HFINTOSC = 32 MHz	—	2.6	3.6	mA	3.0V	
D102	$I_{DD_{HFOPLL}}$	HFINTOSC = 32 MHz	—	2.7	3.7	mA	3.0V	
D103	$I_{DD_{HSPLL32}}$	HS+PLL = 32 MHz	—	2.6	3.6	mA	3.0V	
D103	$I_{DD_{HSPLL32}}$	HS+PLL = 32 MHz	—	2.7	3.7	mA	3.0V	
D104	$I_{DD_{IDLE}}$	IDLE mode, HFINTOSC = 16 MHz	—	1.05	—	mA	3.0V	
D104	$I_{DD_{IDLE}}$	IDLE mode, HFINTOSC = 16 MHz	—	1.15	—	mA	3.0V	
D105	$I_{DD_{DOZE}}$(3)	DOZE mode, HFINTOSC = 16 MHz, Doze Ratio = 16	—	1.1	—	mA	3.0V	
D105	$I_{DD_{DOZE}}$(3)	DOZE mode, HFINTOSC = 16 MHz, Doze Ratio = 16	—	1.2	—	mA	3.0V	

† Data in "Typ." column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: The test conditions for all I_{DD} measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins are outputs driven low; $\overline{MCLR}$ = V_{DD}; WDT disabled.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption.

3: $I_{DD_{DOZE}}$ = [$I_{DD_{IDLE}}$*(N-1)/N] + $I_{DD_{HFO}}$16/N where N = DOZE Ratio (Register 11-2).

4: PMD bits are all in the default state, no modules are disabled.

5: = LF device