

Welcome to [E-XFL.COM](https://www.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	I ² C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	11
Program Memory Size	7KB (4K x 14)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 11x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Through Hole
Package / Case	14-DIP (0.300", 7.62mm)
Supplier Device Package	14-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf1554-e-p

3.2 Data Memory Organization

The data memory is partitioned in 32 memory banks with 128 bytes in a bank. Each bank consists of (Figure 3-3):

- 12 core registers
- 20 Special Function Registers (SFR)
- Up to 80 bytes of General Purpose RAM (GPR)
- 16 bytes of common RAM

The active bank is selected by writing the bank number into the Bank Select Register (BSR). Unimplemented memory will read as '0'. All data memory can be accessed either directly (via instructions that use the file registers) or indirectly via the two File Select Registers (FSR). See **Section 3.5 “Indirect Addressing”** for more information.

Data memory uses a 12-bit address. The upper five bits of the address define the Bank address and the lower seven bits select the registers/RAM in that bank.

3.2.1 CORE REGISTERS

The core registers contain the registers that directly affect the basic operation. The core registers occupy the first 12 addresses of every data memory bank (addresses x00h/x08h through x0Bh/x8Bh). These registers are listed below in Table 3-2. For detailed information, see Table 3-8.

TABLE 3-2: CORE REGISTERS

Addresses	BANKx
x00h or x80h	INDF0
x01h or x81h	INDF1
x02h or x82h	PCL
x03h or x83h	STATUS
x04h or x84h	FSR0L
x05h or x85h	FSR0H
x06h or x86h	FSR1L
x07h or x87h	FSR1H
x08h or x88h	BSR
x09h or x89h	WREG
x0Ah or x8Ah	PCLATH
x0Bh or x8Bh	INTCON

TABLE 3-5: PIC16LF1554/1559 MEMORY MAP, BANKS 8-15

BANK 8		BANK 9		BANK 10		BANK 11		BANK 12		BANK 13		BANK 14		BANK 15	
400h	INDF0	480h	INDF0	500h	INDF0	580h	INDF0	600h	INDF0	680h	INDF0	700h	INDF0	780h	INDF0
401h	INDF1	481h	INDF1	501h	INDF1	581h	INDF1	601h	INDF1	681h	INDF1	701h	INDF1	781h	INDF1
402h	PCL	482h	PCL	502h	PCL	582h	PCL	602h	PCL	682h	PCL	702h	PCL	782h	PCL
403h	STATUS	483h	STATUS	503h	STATUS	583h	STATUS	603h	STATUS	683h	STATUS	703h	STATUS	783h	STATUS
404h	FSR0L	484h	FSR0L	504h	FSR0L	584h	FSR0L	604h	FSR0L	684h	FSR0L	704h	FSR0L	784h	FSR0L
405h	FSR0H	485h	FSR0H	505h	FSR0H	585h	FSR0H	605h	FSR0H	685h	FSR0H	705h	FSR0H	785h	FSR0H
406h	FSR1L	486h	FSR1L	506h	FSR1L	586h	FSR1L	606h	FSR1L	686h	FSR1L	706h	FSR1L	786h	FSR1L
407h	FSR1H	487h	FSR1H	507h	FSR1H	587h	FSR1H	607h	FSR1H	687h	FSR1H	707h	FSR1H	787h	FSR1H
408h	BSR	488h	BSR	508h	BSR	588h	BSR	608h	BSR	688h	BSR	708h	BSR	788h	BSR
409h	WREG	489h	WREG	509h	WREG	589h	WREG	609h	WREG	689h	WREG	709h	WREG	789h	WREG
40Ah	PCLATH	48Ah	PCLATH	50Ah	PCLATH	58Ah	PCLATH	60Ah	PCLATH	68Ah	PCLATH	70Ah	PCLATH	78Ah	PCLATH
40Bh	INTCON	48Bh	INTCON	50Bh	INTCON	58Bh	INTCON	60Bh	INTCON	68Bh	INTCON	70Bh	INTCON	78Bh	INTCON
40Ch	—	48Ch	—	50Ch	—	58Ch	—	60Ch	—	68Ch	—	70Ch	—	78Ch	—
40Dh	—	48Dh	—	50Dh	—	58Dh	—	60Dh	—	68Dh	—	70Dh	—	78Dh	—
40Eh	—	48Eh	—	50Eh	—	58Eh	—	60Eh	—	68Eh	—	70Eh	—	78Eh	—
40Fh	—	48Fh	—	50Fh	—	58Fh	—	60Fh	—	68Fh	—	70Fh	—	78Fh	—
410h	—	490h	—	510h	—	590h	—	610h	—	690h	—	710h	—	790h	—
411h	—	491h	—	511h	—	591h	—	611h	PWM1DCL	691h	—	711h	AD1CON0/ AAD1CON0 ⁽¹⁾	791h	AD2CON0/ AAD2CON0
412h	—	492h	—	512h	—	592h	—	612h	PWM1DCH	692h	—	712h	AADCON1/ ADCOMCON ⁽¹⁾	792h	—
413h	—	493h	—	513h	—	593h	—	613h	PWM1CON	693h	—	713h	AD1CON2/ AAD1CON2 ⁽¹⁾	793h	AD2CON2/ AAD2CON2
414h	—	494h	—	514h	—	594h	—	614h	PWM2DCL	694h	—	714h	AAD1CON3	794h	AAD2CON3
415h	—	495h	—	515h	—	595h	—	615h	PWM2DCH	695h	—	715h	AADSTAT	795h	—
416h	—	496h	—	516h	—	596h	—	616h	PWM2CON	696h	—	716h	AAD1PRE	796h	AAD2PRE
417h	—	497h	—	517h	—	597h	—	617h	—	697h	—	717h	AAD1ACQ	797h	AAD2ACQ
418h	—	498h	—	518h	—	598h	—	618h	—	698h	—	718h	AAD1GRD	798h	AAD2GRD
419h	—	499h	—	519h	—	599h	—	619h	—	699h	—	719h	AAD1CAP	799h	AAD2CAP
41Ah	—	49Ah	—	51Ah	—	59Ah	—	61Ah	—	69Ah	—	71Ah	AD1RES0L/ AAD1RES0L ⁽¹⁾	79Ah	AD2RES0L/ AAD2RES0L
41Bh	—	49Bh	—	51Bh	—	59Bh	—	61Bh	—	69Bh	—	71Bh	AD1RES0H/ AAD1RES0H ⁽¹⁾	79Bh	AD2RES0H/ AAD2RES0H
41Ch	—	49Ch	—	51Ch	—	59Ch	—	61Ch	—	69Ch	—	71Ch	AD1RES1L/ AAD1RES1L	79Ch	AD2RES1L/ AAD2RES1L
41Dh	—	49Dh	—	51Dh	—	59Dh	—	61Dh	—	69Dh	—	71Dh	AD1RES1H/ AAD1RES1H	79Dh	AD2RES1H/ AAD2RES1H
41Eh	—	49Eh	—	51Eh	—	59Eh	—	61Eh	—	69Eh	—	71Eh	AAD1CH	79Eh	AAD2CH
41Fh	—	49Fh	—	51Fh	—	59Fh	—	61Fh	—	69Fh	—	71Fh	—	79Fh	—
420h	Unimplemented Read as '0'	4A0h	Unimplemented Read as '0'	520h	Unimplemented Read as '0'	5A0h	Unimplemented Read as '0'	620h	Unimplemented Read as '0'	6A0h	Unimplemented Read as '0'	720h	Unimplemented Read as '0'	7A0h	Unimplemented Read as '0'
46Fh	Accesses 70h – 7Fh	4EFh	Accesses 70h – 7Fh	56Fh	Accesses 70h – 7Fh	5EFh	Accesses 70h – 7Fh	66Fh	Accesses 70h – 7Fh	6EFh	Accesses 70h – 7Fh	76Fh	Accesses 70h – 7Fh	7EFh	Accesses 70h – 7Fh
470h		4F0h		570h		5F0h		670h		6F0h		770h		7F0h	
47Fh		4FFh		57Fh		5FFh		67Fh		6FFh		77Fh		7FFh	

Note 1: These ADC registers are the same as the registers in Bank 1.

10.4 User ID, Device ID and Configuration Word Access

Instead of accessing program memory, the User ID's, Device ID/Revision ID and Configuration Words can be accessed when $CFG5 = 1$ in the PMCON1 register. This is the region that would be pointed to by $PC<15> = 1$, but not all addresses are accessible. Different access may exist for reads and writes. Refer to Table 10-2.

When read access is initiated on an address outside the parameters listed in Table 10-2, the PMDATH:PMDATL register pair is cleared, reading back '0's.

TABLE 10-2: USER ID, DEVICE ID AND CONFIGURATION WORD ACCESS ($CFG5 = 1$)

Address	Function	Read Access	Write Access
8000h-8003h	User IDs	Yes	Yes
8006h	Device ID/Revision ID	Yes	No
8007h-8008h	Configuration Words 1 and 2	Yes	No

EXAMPLE 10-4: CONFIGURATION WORD AND DEVICE ID ACCESS

```

* This code block will read 1 word of program memory at the memory address:
*   PROG_ADDR_LO (must be 00h-08h) data will be returned in the variables;
*   PROG_DATA_HI, PROG_DATA_LO

BANKSEL  PMADRL          ; Select correct Bank
MOVLW    PROG_ADDR_LO    ;
MOVWF    PMADRL          ; Store LSB of address
CLRF     PMADRH          ; Clear MSB of address

BSF      PMCON1,CFG5     ; Select Configuration Space
BCF      INTCON,GIE      ; Disable interrupts
BSF      PMCON1,RD       ; Initiate read
NOP      ; Executed (See Figure 10-2)
NOP      ; Ignored (See Figure 10-2)
BSF      INTCON,GIE      ; Restore interrupts

MOVF     PMDATL,W        ; Get LSB of word
MOVWF    PROG_DATA_LO    ; Store in user location
MOVF     PMDATH,W        ; Get MSB of word
MOVWF    PROG_DATA_HI    ; Store in user location

```

PIC16LF1554/1559

11.6 Register Definitions: PORTB

REGISTER 11-7: PORTB: PORTB REGISTER

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	U-0	U-0	U-0	U-0
RB7	RB6	RB5	RB4	—	—	—	—
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 7-4 **RB<7:4>**: PORTB I/O Value bits⁽¹⁾
 1 = Port pin is $\geq V_{IH}$
 0 = Port pin is $\leq V_{IL}$

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

Note 1: Writes to PORTB are actually written to corresponding LATB register. Reads from PORTB register is return of actual I/O pin values.

REGISTER 11-8: TRISB: PORTB TRI-STATE REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	U-0	U-0	U-0	U-0
TRISB7	TRISB6	TRISB5	TRISB4	—	—	—	—
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 7-4 **TRISB<7:4>**: PORTB Tri-State Control bits
 1 = PORTB pin configured as an input (tri-stated)
 0 = PORTB pin configured as an output

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

REGISTER 11-14: LATC: PORTC DATA LATCH 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
LATC7 ⁽¹⁾	LATC6 ⁽¹⁾	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0
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 **LATC<7:0>**: PORTC Output Latch Value bits⁽²⁾

Note 1: Functions not available on PIC16LF1554.

2: Writes to PORTC are actually written to corresponding LATC register. Reads from PORTC register is return of actual I/O pin values.

REGISTER 11-15: ANSEL: PORTC ANALOG SELECT 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
ANSC7 ⁽¹⁾	ANSC6 ⁽¹⁾	ANSC5	ANSC4	ANSC3	ANSC2	ANSC1	ANSC0
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 **ANSC<7:0>**: Analog Select between Analog or Digital Function on pins RC<7:0>, respectively
1 = Analog input. Pin is assigned as analog input⁽²⁾. Digital input buffer disabled.
0 = Digital I/O. Pin is assigned to port or digital special function.

Note 1: Functions not available on PIC16LF1554.

2: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

TABLE 11-9: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSEL	ANSC7 ⁽¹⁾	ANSC6 ⁽¹⁾	ANSC5	ANSC4	ANSC3	ANSC2	ANSC1	ANSC0	117
LATC	LATC7 ⁽¹⁾	LATC6 ⁽¹⁾	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	117
PORTC	RC7 ⁽¹⁾	RC6 ⁽¹⁾	RC5	RC4	RC3	RC2	RC1	RC0	116
TRISC	TRISC7 ⁽¹⁾	TRISC6 ⁽¹⁾	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	116

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

Note 1: Functions not available on PIC16LF1554.

PIC16LF1554/1559

FIGURE 16-1: DIFFERENTIAL CVD MEASUREMENT WAVEFORM

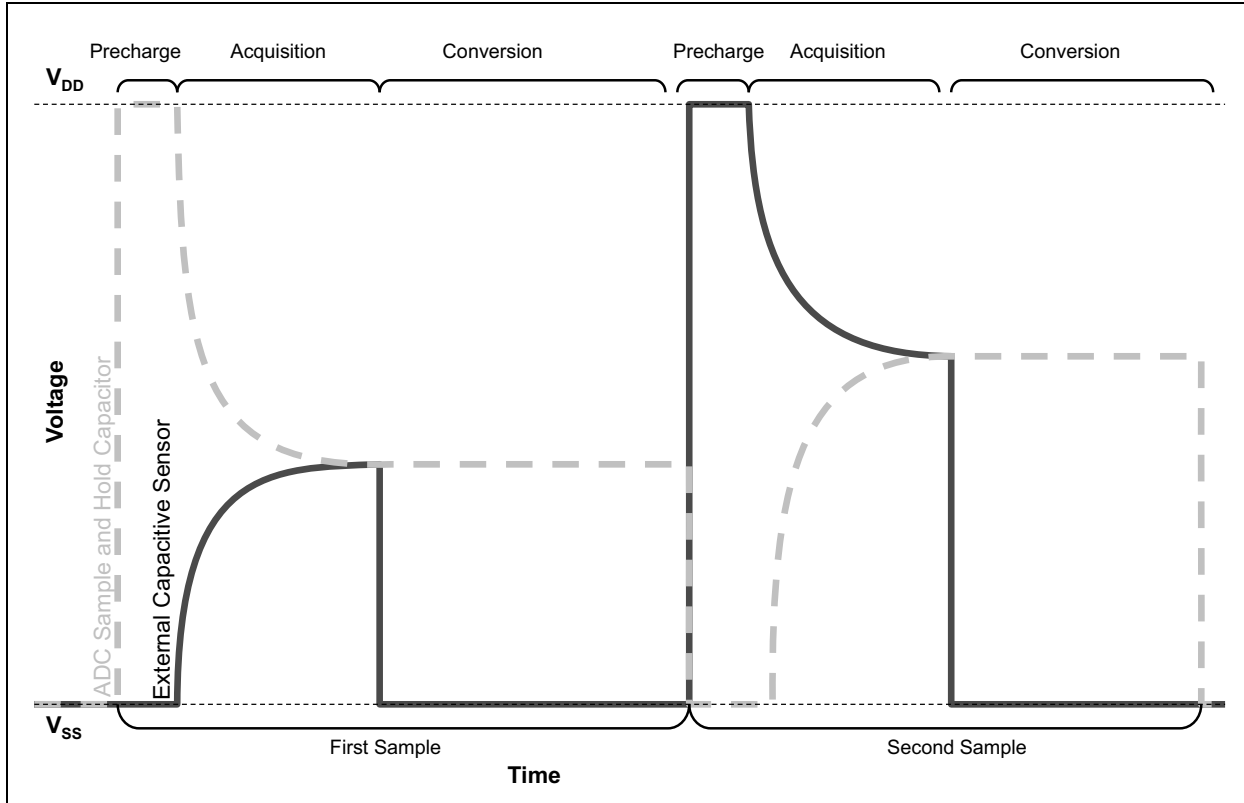
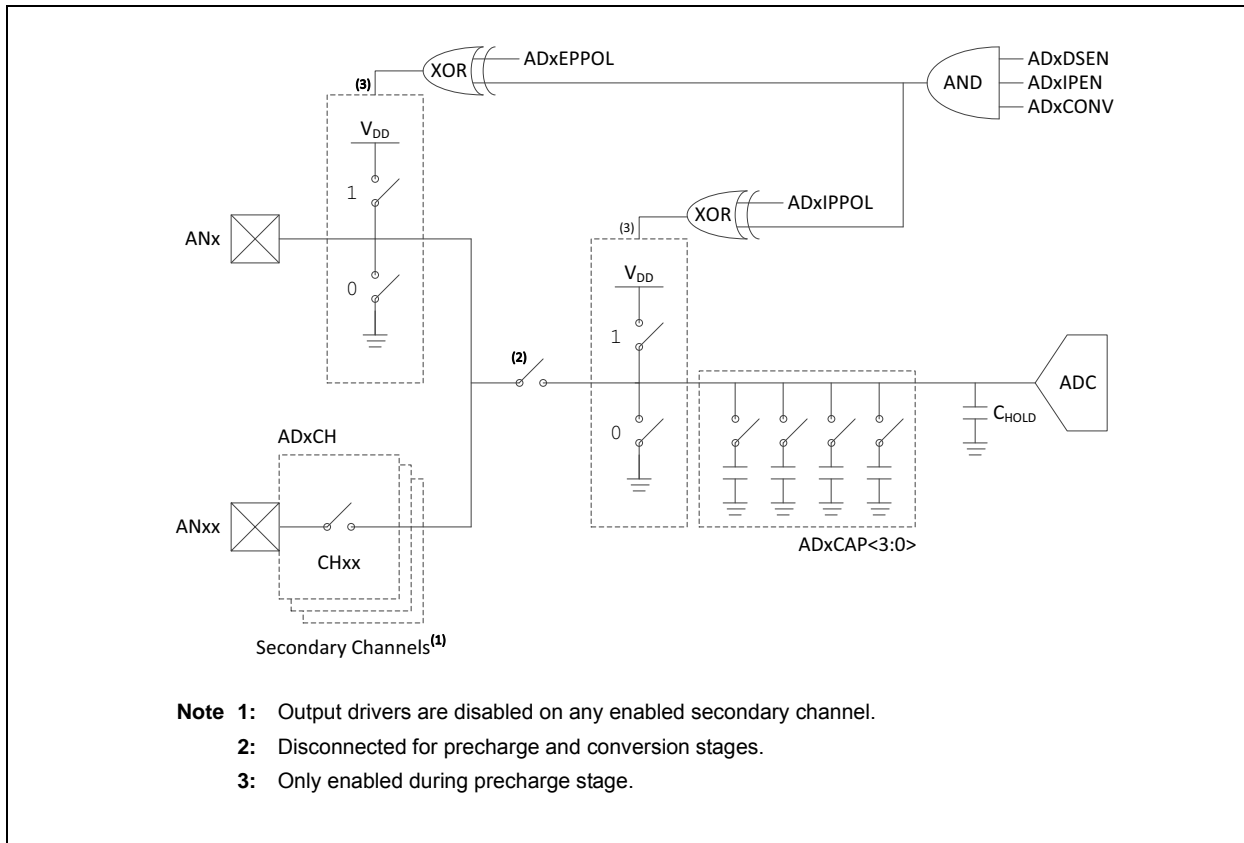


FIGURE 16-2: HARDWARE CAPACITIVE VOLTAGE DIVIDER BLOCK DIAGRAM



PIC16LF1554/1559

16.1.5 TERMINATING A CONVERSION

If a conversion must be terminated before completion, clear the $\overline{\text{GO/DONE}}_x$ bit. The AADxRESxH and AADxRESxL registers will be updated with the partially complete Analog-to-Digital conversion sample. Incomplete bits will match the last bit converted.

The AADSTAT register can be used to track the status of the hardware CVD module during a conversion.

Note: A device Reset forces all registers to their Reset state. Thus, the ADC module is turned off and any pending conversion is terminated.

16.1.6 DOUBLE SAMPLE CONVERSION

Double sampling can be enabled by setting the AADxSEN bit of the AADxCON3 register. When this bit is set, two conversions are completed each time the $\overline{\text{GO/DONE}}_x$, $\overline{\text{GO/DONE_ALL}}$ bit is set or a Special Event Trigger occurs. The $\overline{\text{GO/DONE}}_x$ or $\overline{\text{GO/DONE_ALL}}$ bits remain set for the duration of both conversions and is used to signal the end of the conversion.

Without setting the ADxIPEN bit, the double conversion will have identical charge/discharge on the internal and external capacitor for these two conversions. Setting the ADxIPEN bit prior to a double conversion will allow the user to perform a pseudo-differential CVD measurement by subtracting the results from the double conversion. This is highly recommended for noise immunity purposes.

The result of the first conversion is written to the AADxRES0H and AADxRES0L registers. The second conversion starts two clock cycles after the first has completed, while the $\overline{\text{GO/DONE}}_x$ and $\overline{\text{GO/DONE_ALL}}$ bits remain set. When the ADxIPEN bit of AADxCON3 is set, the value used by the ADC for the ADxEPPOL , ADxIPPOL and GRDxPOL bits are inverted. The value stored in those bit locations is unchanged. All other control signals remain unchanged from the first conversion. The result of the second conversion is stored in the AADxRES1H and AADxRES1L registers. See Figure 16-4 and Figure 16-5 for more information.

16.1.7 GUARD RING OUTPUTS

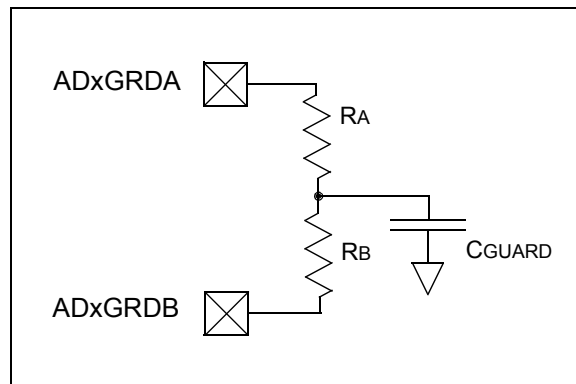
The guard ring outputs consist of a pair of digital outputs from the hardware CVD module. Each ADC has its own pair of guard ring outputs. This function is enabled by the GRDxAOE and GRDxBOE bits of the AADxGRD register. Polarity of the output is controlled by the GRDxPOL bit.

Once enabled and while $\text{ADxON} = 1$, the guard ring outputs of the ADC are active at all times. The outputs are initialized at the start of the precharge stage to match the polarity of the GRDxPOL bit. The guard output signal changes polarity at the start of the acquisition phase. The value stored by the GRDPOL bit does not change. When in Double Sampling mode, the ring output levels are inverted during the second precharge and acquisition phases if $\text{ADDxSEN} = 1$ and $\text{ADxIPEN} = 1$. For more information on the timing of the guard ring output, refer to Figure 16-4 and Figure 16-5.

A typical guard ring circuit is displayed in Figure 16-2. CGUARD represents the capacitance of the guard ring trace placed on a PCB board. The user selects values for R_A and R_B that will create a voltage profile on CGUARD , which will match the selected channel during acquisition.

The purpose of the guard ring is to generate a signal in phase with the CVD sensing signal to minimize the effects of the parasitic capacitance on sensing electrodes. It also can be used as a mutual drive for mutual capacitive sensing. For more information about active guard and mutual drive, see Application Note AN1478, "mTouch® Sensing Solution Acquisition Methods Capacitive Voltage Divider" (DS01478).

FIGURE 16-3: GUARD RING CIRCUIT



PIC16LF1554/1559

16.1.10 HARDWARE CVD DOUBLE CONVERSION PROCEDURE

This is an example procedure for using hardware CVD to perform a double conversion for differential CVD measurement with active guard drive.

1. Configure Port:
 - Enable pin output driver (Refer to the TRISx register).
 - Configure pin output low (Refer to the LATx register).
 - Disable weak pull-up (Refer to the WPUx register).
2. Configure the ADC module:
 - Select an appropriate ADC conversion clock for your oscillator frequency.
 - Configure voltage reference.
 - Select ADC input channel.
 - Turn on the ADC module.
3. Configure the hardware CVD module:
 - Configure charge polarity and double conversion.
 - Configure precharge and acquisition timer.
 - Configure guard ring (optional).
 - Select additional capacitance (optional).
4. Configure ADC interrupt (optional):
 - Clear ADC interrupt flag
 - Enable ADC interrupt
 - Enable peripheral interrupt
 - Enable global interrupt⁽¹⁾
5. Start conversion by setting the $\overline{\text{GO/DONE}}_{\text{EX}}$, $\overline{\text{GO/DONE}}_{\text{ALL}}$ bit or by enabling the Special Event Trigger in the ADDxCON2 register.
6. Wait for the ADC conversion to complete by one of the following:
 - Polling the $\overline{\text{GO/DONE}}_{\text{EX}}$ or $\overline{\text{GO/DONE}}_{\text{ALL}}$ bit.
 - Waiting for the ADC interrupt (interrupts enabled).
7. Read ADC result:
 - Conversion 1 result in ADDxRES0H and ADDxRES0L
 - Conversion 2 result in ADDxRES1H and ADDxRES1L
8. Clear the ADC interrupt flag (required if interrupt is enabled).

Note 1: The global interrupt can be disabled if the user is attempting to wake-up from Sleep and resume in-line code execution.

EXAMPLE 16-1: HARDWARE CVD DOUBLE CONVERSION

```
;This code block configures the ADC
;for polling, VDD and VSS references, Fosc/16
;clock and AN0 input.
;
;The Hardware CVD1 will perform an inverted
;double conversion, Guard A and B drive are
;both enabled.
;Conversion start & polling for completion
;are included.
;
BANKSEL    TRISA
BCF        TRISA,0      ;Set RA0 to output
BANKSEL    LATA
BCF        LATA,0       ;RA0 output low
BANKSEL    ANSELA
BCF        ANSELA,0     ;Set RA0 to digital
BANKSEL    WPUA
BCF        WPUA,0       ;Disable pull-up on
RA0
;Initialize ADC and Hardware CVD

BANKSEL    AAD1CON0
MOVLW     B'00000001'   ;Select channel AN0
MOVWF     AAD1CON0
BANKSEL    AADCON1
MOVLW     B'11010000'   ;VDD and VSS VREF
MOVWF     AADCON1
MOVLW     B'00000000'   ;No secondary channel
MOVWF     AAD1CH

BANKSEL    AAD1CON3
MOVLW     B'01000011'   ;Double and inverted
MOVWF     AAD1CON3
BANKSEL    AAD1PRE
MOVLW     .10
MOVWF     AAD1PRE       ;Pre-charge Timer
BANKSEL    AAD1ACQ
MOVLW     .10
MOVWF     AAD1ACQ       ;Acquisition Timer
BANKSEL    AAD1GRD
MOVLW     B'11000000'   ;Guard on A and B
MOVWF     AAD1GRD
BANKSEL    AAD1CAP
MOVLW     B'00000000'   ;No additional
MOVWF     AAD1CAP       ;Capacitance

BANKSEL    AD1CON0
BSF        AD1CON0, GO
BTFSCL    AD1CON0, GO
GOTO      $-1           ;No, test again

;RESULTS OF CONVERSIONS 1.
BANKSEL    AAD1RES0H
MOVF       AAD1RES0H,W   ;Read upper 2 bits
MOVWF     RESULT0H      ;Store in GPR space
MOVF       AAD1RES0L,W   ;Read lower 8 bits
MOVWF     RESULT0L      ;Store in GPR space

;RESULTS OF CONVERSIONS 2.
BANKSEL    AAD1RES1H
MOVF       AAD1RES1H,W   ;Read upper 2 bits
MOVWF     RESULT1H      ;Store in GPR space
MOVF       AAD1RES1L,W   ;Read lower 8 bits
MOVWF     RESULT1L      ;Store in GPR space
```

PIC16LF1554/1559

REGISTER 16-15: AADxRESxH: HARDWARE CVD RESULT REGISTER MSB ADFM = 1⁽¹⁾

U-0	U-0	U-0	U-0	U-0	U-0	R/W-x/u	R/W-x/u
—	—	—	—	—	—	ADRESx<9:8>	
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 7-2 **Reserved:** Do not use.

bit 1-0 **ADRESx<9:8>:** ADC Result Register bits
Upper two bits of 10-bit conversion result

Note 1: See Section 16.1.11 “Hardware CVD Register Mapping” for more information.

REGISTER 16-16: AADxRESxL: HARDWARE CVD RESULT REGISTER LSB ADFM = 1⁽¹⁾

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
ADRESx<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 **ADRESx<7:0>:** ADC Result Register bits
Lower eight bits of 10-bit conversion result

Note 1: See Section 16.1.11 “Hardware CVD Register Mapping” for more information.

PIC16LF1554/1559

REGISTER 18-2: T1GCON: TIMER1 GATE CONTROL REGISTER

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W/HC-0/u	R-x/x	U-0	R/W-0/u
TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/DONE	T1GVAL	—	T1GSS
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

bit 7 **TMR1GE:** Timer1 Gate Enable bit

If TMR1ON = 0:

This bit is ignored

If TMR1ON = 1:

1 = Timer1 counting is controlled by the Timer1 gate function

0 = Timer1 counts regardless of Timer1 gate function

bit 6 **T1GPOL:** Timer1 Gate Polarity bit

1 = Timer1 gate is active-high (Timer1 counts when gate is high)

0 = Timer1 gate is active-low (Timer1 counts when gate is low)

bit 5 **T1GTM:** Timer1 Gate Toggle Mode bit

1 = Timer1 Gate Toggle mode is enabled

0 = Timer1 Gate Toggle mode is disabled and toggle flip-flop is cleared

Timer1 gate flip-flop toggles on every rising edge.

bit 4 **T1GSPM:** Timer1 Gate Single-Pulse Mode bit

1 = Timer1 gate Single-Pulse mode is enabled and is controlling Timer1 gate

0 = Timer1 gate Single-Pulse mode is disabled

bit 3 **T1GGO/DONE:** Timer1 Gate Single-Pulse Acquisition Status bit

1 = Timer1 gate single-pulse acquisition is ready, waiting for an edge

0 = Timer1 gate single-pulse acquisition has completed or has not been started

bit 2 **T1GVAL:** Timer1 Gate Value Status bit

Indicates the current state of the Timer1 gate that could be provided to TMR1H:TMR1L.

Unaffected by Timer1 Gate Enable (TMR1GE).

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

bit 0 **T1GSS:** Timer1 Gate Source Select bits

01 = Timer0 overflow output (T0_overflow)

00 = Timer1 gate pin (T1G)

20.2.4 SPI SLAVE MODE

In Slave mode, the data is transmitted and received as external clock pulses appear on SCK. When the last bit is latched, the SSP1IF interrupt flag bit is set.

Before enabling the module in SPI Slave mode, the clock line must match the proper Idle state. The clock line can be observed by reading the SCK pin. The Idle state is determined by the CKP bit of the SSPCON1 register.

While in Slave mode, the external clock is supplied by the external clock source on the SCK pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in Sleep mode, the slave can transmit/receive data. The shift register is clocked from the SCK pin input and when a byte is received, the device will generate an interrupt. If enabled, the device will wake-up from Sleep.

20.2.4.1 Daisy-Chain Configuration

The SPI bus can sometimes be connected in a daisy-chain configuration. The first slave output is connected to the second slave input, the second slave output is connected to the third slave input, and so on. The final slave output is connected to the master input. Each slave sends out, during a second group of clock pulses, an exact copy of what was received during the first group of clock pulses. The whole chain acts as one large communication shift register. The daisy-chain feature only requires a single Slave Select line from the master device.

Figure 20-7 shows the block diagram of a typical daisy-chain connection when operating in SPI mode.

In a daisy-chain configuration, only the most recent byte on the bus is required by the slave. Setting the BOEN bit of the SSPCON3 register will enable writes to the SSPBUF register, even if the previous byte has not been read. This allows the software to ignore data that may not apply to it.

20.2.5 SLAVE SELECT SYNCHRONIZATION

The Slave Select can also be used to synchronize communication. The Slave Select line is held high until the master device is ready to communicate. When the Slave Select line is pulled low, the slave knows that a new transmission is starting.

If the slave fails to receive the communication properly, it will be reset at the end of the transmission, when the Slave Select line returns to a high state. The slave is then ready to receive a new transmission when the Slave Select line is pulled low again. If the Slave Select line is not used, there is a risk that the slave will eventually become out of sync with the master. If the slave misses a bit, it will always be one bit off in future transmissions. Use of the Slave Select line allows the slave and master to align themselves at the beginning of each transmission.

The \overline{SS} pin allows a Synchronous Slave mode. The SPI must be in Slave mode with \overline{SS} pin control enabled (SSPCON1<3:0> = 0100).

When the \overline{SS} pin is low, transmission and reception are enabled and the SDO pin is driven.

When the \overline{SS} pin goes high, the SDO pin is no longer driven, even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

- | |
|---|
| <p>Note 1: When the SPI is in Slave mode with \overline{SS} pin control enabled (SSPCON1<3:0> = 0100), the SPI module will reset if the \overline{SS} pin is set to VDD.</p> <p>2: When the SPI is used in Slave mode with CKE set; the user must enable \overline{SS} pin control.</p> <p>3: While operated in SPI Slave mode the SMP bit of the SSPSTAT register must remain clear.</p> |
|---|

When the SPI module resets, the bit counter is forced to '0'. This can be done by either forcing the \overline{SS} pin to a high level or clearing the SSPEN bit.

PIC16LF1554/1559

FIGURE 20-7: SPI DAISY-CHAIN CONNECTION

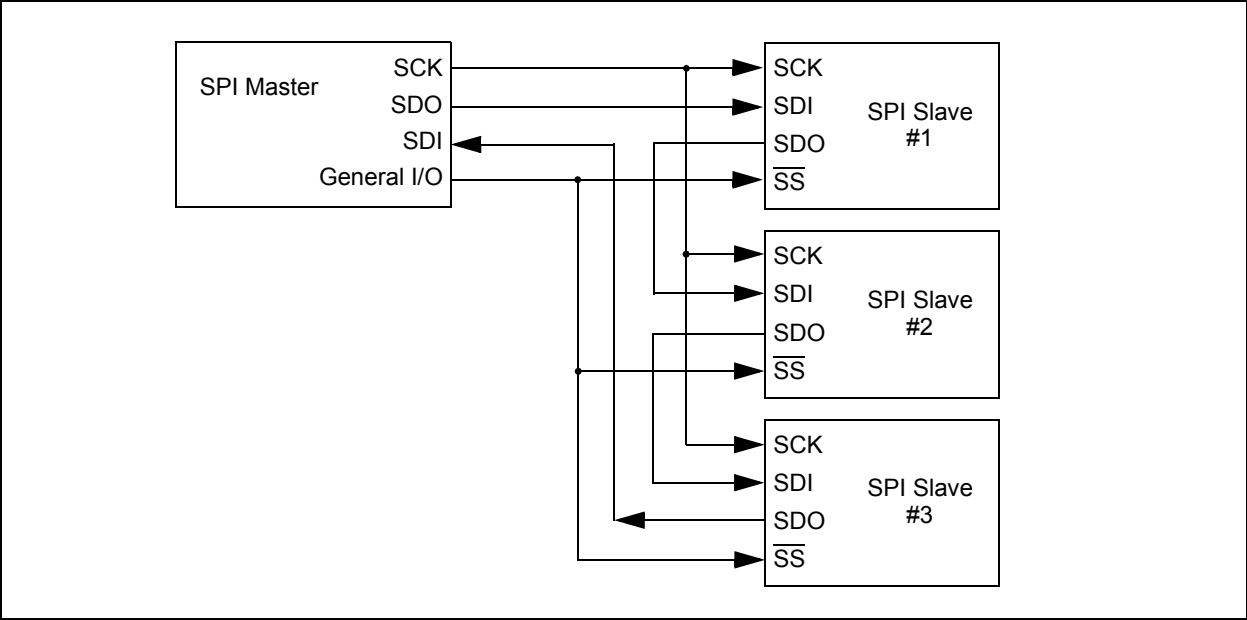
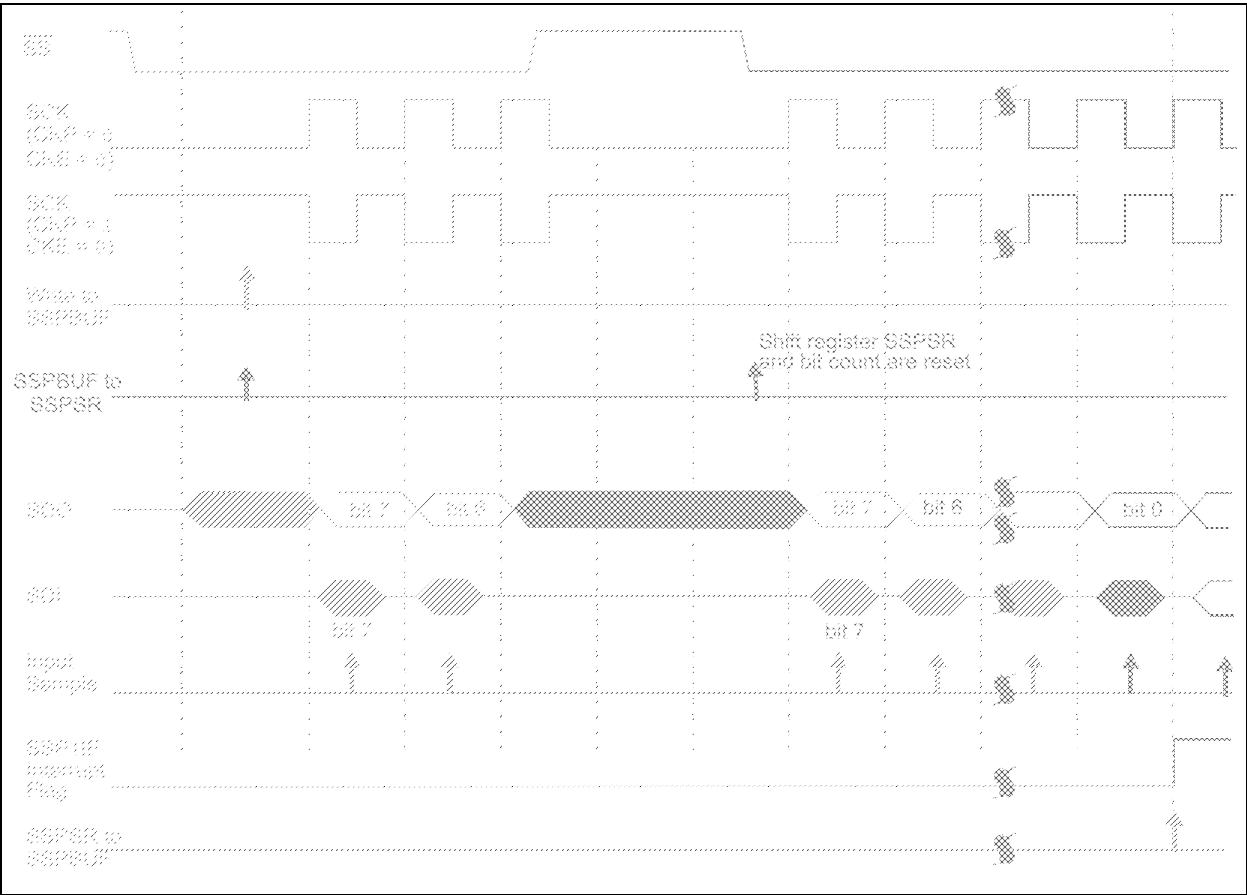


FIGURE 20-8: SLAVE SELECT SYNCHRONOUS WAVEFORM



20.4 I²C MODE OPERATION

All MSSP I²C communication is byte oriented and shifted out MSb first. Six SFR registers and two interrupt flags interface the module with the PIC[®] microcontroller and user software. Two pins, SDA and SCL, are exercised by the module to communicate with other external I²C devices.

20.4.1 BYTE FORMAT

All communication in I²C is done in 9-bit segments. A byte is sent from a master to a slave or vice-versa, followed by an Acknowledge bit sent back. After the 8th falling edge of the SCL line, the device outputting data on the SDA changes that pin to an input and reads in an acknowledge value on the next clock pulse.

The clock signal, SCL, is provided by the master. Data is valid to change while the SCL signal is low, and sampled on the rising edge of the clock. Changes on the SDA line while the SCL line is high define special conditions on the bus, explained below.

20.4.2 DEFINITION OF I²C TERMINOLOGY

There is language and terminology in the description of I²C communication that have definitions specific to I²C. That word usage is defined below and may be used in the rest of this document without explanation. This table was adapted from the Philips I²C specification.

20.4.3 SDA AND SCL PINS

Selection of any I²C mode with the SSPEN bit set, forces the SCL and SDA pins to be open-drain. These pins should be set by the user to inputs by setting the appropriate TRISx bits.

Note: Data is tied to output zero when an I²C mode is enabled.

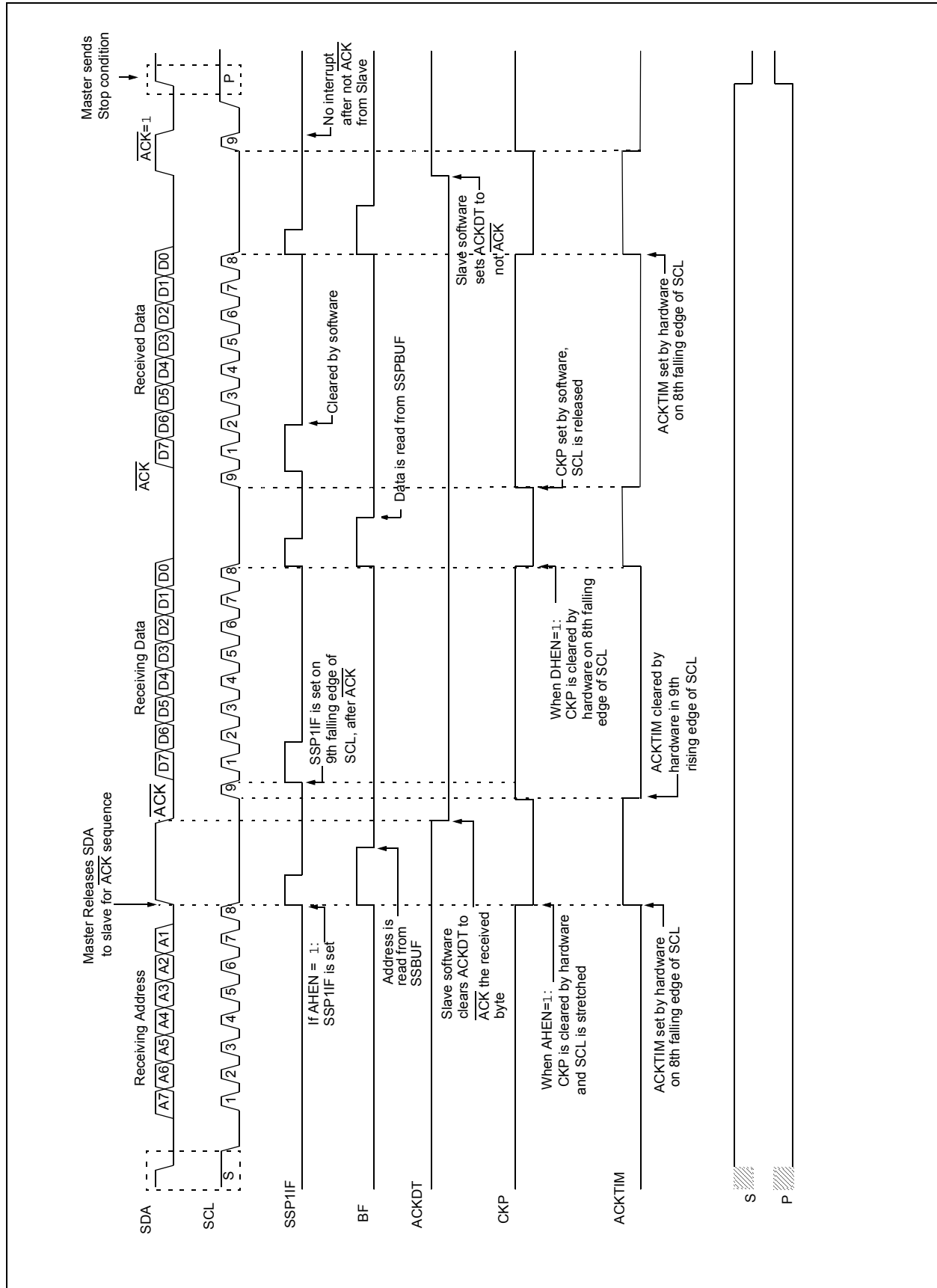
20.4.4 SDA HOLD TIME

The hold time of the SDA pin is selected by the SDAHT bit of the SSPCON3 register. Hold time is the time SDA is held valid after the falling edge of SCL. Setting the SDAHT bit selects a longer 300 ns minimum hold time and may help on buses with large capacitance.

TABLE 20-2: I²C BUS TERMS

TERM	Description
Transmitter	The device which shifts data out onto the bus.
Receiver	The device which shifts data in from the bus.
Master	The device that initiates a transfer, generates clock signals and terminates a transfer.
Slave	The device addressed by the master.
Multi-master	A bus with more than one device that can initiate data transfers.
Arbitration	Procedure to ensure that only one master at a time controls the bus. Winning arbitration ensures that the message is not corrupted.
Synchronization	Procedure to synchronize the clocks of two or more devices on the bus.
Idle	No master is controlling the bus, and both SDA and SCL lines are high.
Active	Any time one or more master devices are controlling the bus.
Addressed Slave	Slave device that has received a matching address and is actively being clocked by a master.
Matching Address	Address byte that is clocked into a slave that matches the value stored in SSPADD.
Write Request	Slave receives a matching address with R/W bit clear, and is ready to clock in data.
Read Request	Master sends an address byte with the R/W bit set, indicating that it wishes to clock data out of the slave. This data is the next and all following bytes until a Restart or Stop.
Clock Stretching	When a device on the bus hold SCL low to stall communication.
Bus Collision	Any time the SDA line is sampled low by the module while it is outputting and expected high state.

FIGURE 20-16: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 1)



20.6 I²C MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPM bits in the SSPCON1 register and by setting the SSPEN bit. In Master mode, the SDA and SCK pins must be configured as inputs. The MSSP peripheral hardware will override the output driver TRISx controls when necessary to drive the pins low.

Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I²C bus may be taken when the P bit is set, or the bus is Idle.

In Firmware Controlled Master mode, user code conducts all I²C bus operations based on Start and Stop bit condition detection. Start and Stop condition detection is the only active circuitry in this mode. All other communication is done by the user software directly manipulating the SDA and SCL lines.

The following events will cause the SSP interrupt flag bit, SSP1IF, to be set (SSP interrupt, if enabled):

- Start condition detected
- Stop condition detected
- Data transfer byte transmitted/received
- Acknowledge transmitted/received
- Repeated Start generated

Note 1: The MSSP module, when configured in I²C Master mode, does not allow queuing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPBUF register to initiate transmission before the Start condition is complete. In this case, the SSPBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPBUF did not occur

2: Master mode suspends Start/Stop detection when sending the Start/Stop condition by means of the SEN/PEN control bits. The SSPIF bit is set at the end of the Start/Stop generation when hardware clears the Control bit.

20.6.1 I²C MASTER MODE OPERATION

The master device generates all of the serial clock pulses and the Start and Stop conditions. A transfer is ended with a Stop condition or with a Repeated Start condition. Since the Repeated Start condition is also the beginning of the next serial transfer, the I²C bus will not be released.

In Master Transmitter mode, serial data is output through SDA, while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/W) bit. In this case, the R/W bit will be logic '0'. Serial data is transmitted eight bits at a time. After each byte is transmitted, an Acknowledge bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/W bit. In this case, the R/W bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate the receive bit. Serial data is received via SDA, while SCL outputs the serial clock. Serial data is received eight bits at a time. After each byte is received, an Acknowledge bit is transmitted. Start and Stop conditions indicate the beginning and end of transmission.

A Baud Rate Generator is used to set the clock frequency output on SCL. See **Section 20.7 "Baud Rate Generator"** for more detail.

20.6.2 CLOCK ARBITRATION

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, releases the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD and begins counting. This ensures that the SCL high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 20-25).

PIC16LF1554/1559

20.6.13.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

- After the SDA pin has been deasserted and allowed to float high, SDA is sampled low after the BRG has timed out (Case 1).
- After the SCL pin is deasserted, SCL is sampled low before SDA goes high (Case 2).

The Stop condition begins with SDA asserted low. When SDA is sampled low, the SCL pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPADD and counts down to 0. After the BRG times out, SDA is sampled. If SDA is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 20-38). If the SCL pin is sampled low before SDA is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 20-39).

FIGURE 20-38: BUS COLLISION DURING A STOP CONDITION (CASE 1)

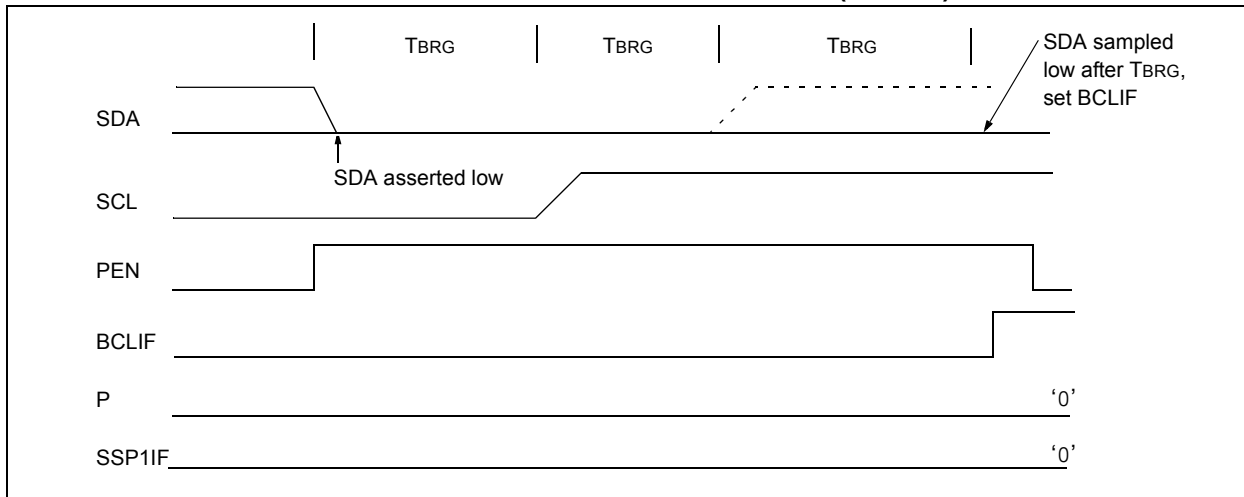


FIGURE 20-39: BUS COLLISION DURING A STOP CONDITION (CASE 2)

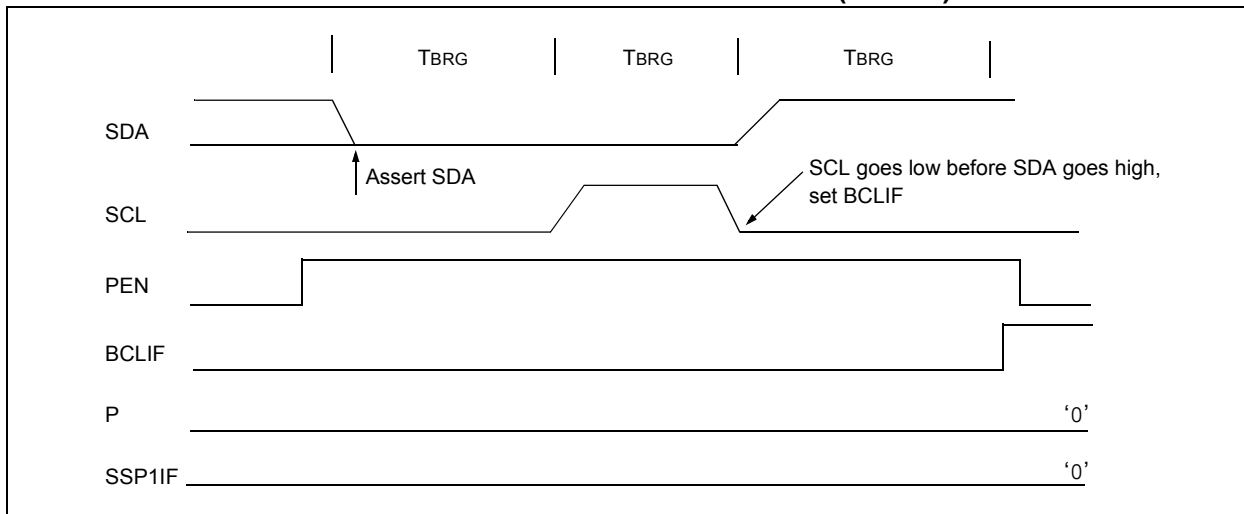
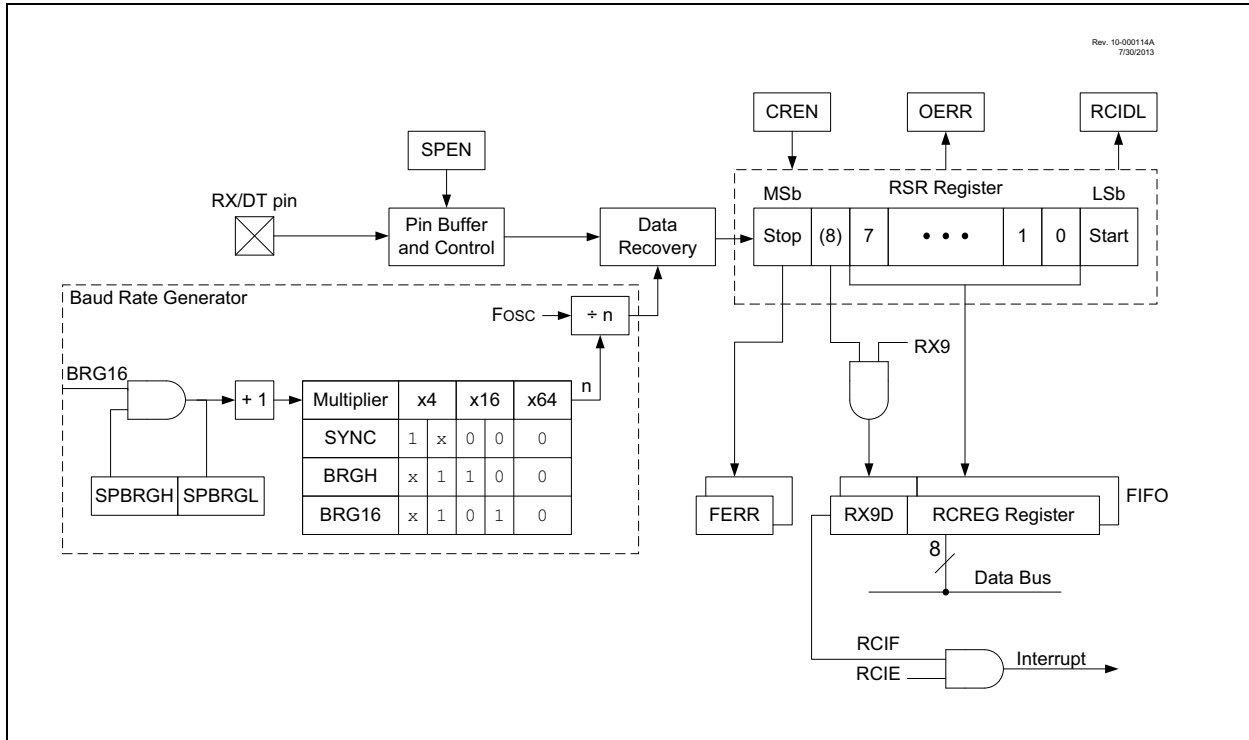


FIGURE 21-2: EUSART RECEIVE BLOCK DIAGRAM



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

- Transmit Status and Control (TXSTA)
- Receive Status and Control (RCSTA)
- Baud Rate Control (BAUDCON)

These registers are detailed in Register 21-1, Register 21-2 and Register 21-3, respectively.

When the receiver or transmitter section is not enabled then the corresponding RX or TX pin may be used for general purpose input and output.

21.1 EUSART Asynchronous Mode

The EUSART transmits and receives data using the standard non-return-to-zero (NRZ) format. NRZ is implemented with two levels: a VoH Mark state which represents a '1' data bit, and a VoL Space state which represents a '0' data bit. NRZ refers to the fact that consecutively transmitted data bits of the same value stay at the output level of that bit without returning to a neutral level between each bit transmission. An NRZ transmission port idles in the Mark state. Each character transmission consists of one Start bit followed by eight or nine data bits and is always terminated by one or more Stop bits. The Start bit is always a space and the Stop bits are always marks. The most common data format is eight bits. Each transmitted bit persists for a period of 1/(Baud Rate). An on-chip dedicated 8-bit/16-bit Baud Rate Generator is used to derive standard baud rate frequencies from the system oscillator. See Table 21-5 for examples of baud rate configurations.

The EUSART transmits and receives the LSb first. The EUSART's transmitter and receiver are functionally independent, but share the same data format and baud rate. Parity is not supported by the hardware, but can be implemented in software and stored as the ninth data bit.

21.1.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in Figure 21-1. The heart of the transmitter is the serial Transmit Shift Register (TSR), which is not directly accessible by software. The TSR obtains its data from the transmit buffer, which is the TXREG register.

21.1.1.1 Enabling the Transmitter

The EUSART transmitter is enabled for asynchronous operations by configuring the following three control bits:

- TXEN = 1
- SYNC = 0
- SPEN = 1

All other EUSART control bits are assumed to be in their default state.

Setting the TXEN bit of the TXSTA register enables the transmitter circuitry of the EUSART. Clearing the SYNC bit of the TXSTA register configures the EUSART for asynchronous operation. Setting the SPEN bit of the RCSTA register enables the EUSART and automatically configures the TX/CK I/O pin as an output. If the TX/CK pin is shared with an analog peripheral, the analog I/O function must be disabled by clearing the corresponding ANSELx bit.

Note: The TXIF Transmitter Interrupt flag is set when the TXEN enable bit is set.

21.1.1.2 Transmitting Data

A transmission is initiated by writing a character to the TXREG register. If this is the first character, or the previous character has been completely flushed from the TSR, the data in the TXREG is immediately transferred to the TSR register. If the TSR still contains all or part of a previous character, the new character data is held in the TXREG until the Stop bit of the previous character has been transmitted. The pending character in the TXREG is then transferred to the TSR in one Tcy immediately following the Stop bit transmission. The transmission of the Start bit, data bits and Stop bit sequence commences immediately following the transfer of the data to the TSR from the TXREG.

21.1.1.3 Transmit Data Polarity

The polarity of the transmit data can be controlled with the SCKP bit of the BAUDCON register. The default state of this bit is '0' which selects high true transmit idle and data bits. Setting the SCKP bit to '1' will invert the transmit data resulting in low true idle and data bits. The SCKP bit controls transmit data polarity in Asynchronous mode only. In Synchronous mode, the SCKP bit has a different function. See **Section 21.5.1.2 "Clock Polarity"**.

21.1.1.4 Transmit Interrupt Flag

The TXIF interrupt flag bit of the PIR1 register is set whenever the EUSART transmitter is enabled and no character is being held for transmission in the TXREG. In other words, the TXIF bit is only clear when the TSR is busy with a character and a new character has been queued for transmission in the TXREG. The TXIF flag bit is not cleared immediately upon writing TXREG. TXIF becomes valid in the second instruction cycle following the write execution. Polling TXIF immediately following the TXREG write will return invalid results. The TXIF bit is read-only, it cannot be set or cleared by software.

The TXIF interrupt can be enabled by setting the TXIE interrupt enable bit of the PIE1 register. However, the TXIF flag bit will be set whenever the TXREG is empty, regardless of the state of TXIE enable bit.

To use interrupts when transmitting data, set the TXIE bit only when there is more data to send. Clear the TXIE interrupt enable bit upon writing the last character of the transmission to the TXREG.

PIC16LF1554/1559

21.4.4 BREAK CHARACTER SEQUENCE

The EUSART module has the capability of sending the special Break character sequences that are required by the LIN bus standard. A Break character consists of a Start bit, followed by 12 '0' bits and a Stop bit.

To send a Break character, set the SENDB and TXEN bits of the TXSTA register. The Break character transmission is then initiated by a write to the TXREG. The value of data written to TXREG will be ignored and all '0's will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN specification).

The TRMT bit of the TXSTA register indicates when the transmit operation is active or idle, just as it does during normal transmission. See Figure 21-9 for the timing of the Break character sequence.

21.4.4.1 Break and Sync Transmit Sequence

The following sequence will start a message frame header made up of a Break, followed by an auto-baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the EUSART for the desired mode.
2. Set the TXEN and SENDB bits to enable the Break sequence.
3. Load the TXREG with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXREG to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware and the Sync character is then transmitted.

When the TXREG becomes empty, as indicated by the TXIF, the next data byte can be written to TXREG.

21.4.5 RECEIVING A BREAK CHARACTER

The Enhanced EUSART module can receive a Break character in two ways.

The first method to detect a Break character uses the FERR bit of the RCSTA register and the received data as indicated by RCREG. The Baud Rate Generator is assumed to have been initialized to the expected baud rate.

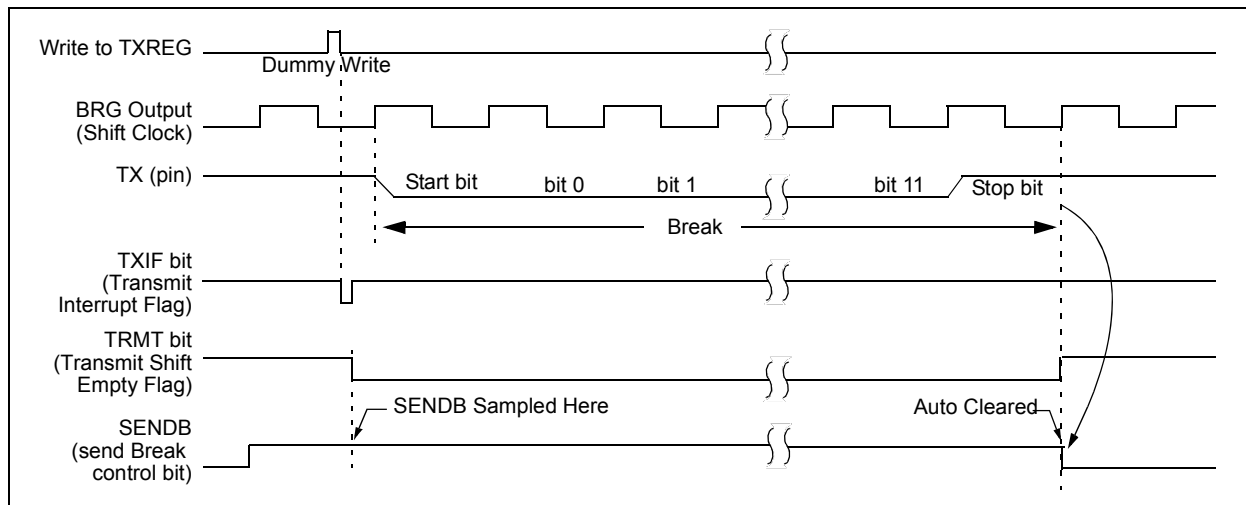
A Break character has been received when;

- RCIF bit is set
- FERR bit is set
- RCREG = 00h

The second method uses the Auto-Wake-up feature described in **Section 21.4.3 “Auto-Wake-up on Break”**. By enabling this feature, the EUSART will sample the next two transitions on RX/DT, cause an RCIF interrupt, and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Detect feature. For both methods, the user can set the ABDEN bit of the BAUDCON register before placing the EUSART in Sleep mode.

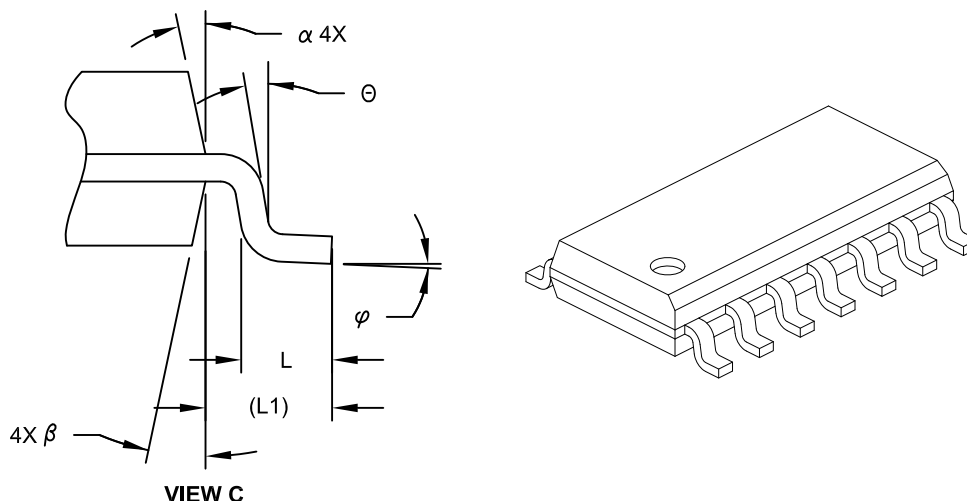
FIGURE 21-9: SEND BREAK CHARACTER SEQUENCE



PIC16LF1554/1559

14-Lead Plastic Small Outline (SL) - Narrow, 3.90 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	14		
Pitch	e	1.27 BSC		
Overall Height	A	-	-	1.75
Molded Package Thickness	A2	1.25	-	-
Standoff §	A1	0.10	-	0.25
Overall Width	E	6.00 BSC		
Molded Package Width	E1	3.90 BSC		
Overall Length	D	8.65 BSC		
Chamfer (Optional)	h	0.25	-	0.50
Foot Length	L	0.40	-	1.27
Footprint	L1	1.04 REF		
Lead Angle	Θ	0°	-	-
Foot Angle	φ	0°	-	8°
Lead Thickness	c	0.10	-	0.25
Lead Width	b	0.31	-	0.51
Mold Draft Angle Top	α	5°	-	15°
Mold Draft Angle Bottom	β	5°	-	15°

Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- § Significant Characteristic
- Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm per side.
- Dimensioning and tolerancing per ASME Y14.5M
 - BSC: Basic Dimension. Theoretically exact value shown without tolerances.
 - REF: Reference Dimension, usually without tolerance, for information purposes only.
- Datums A & B to be determined at Datum H.

Microchip Technology Drawing No. C04-065C Sheet 2 of 2