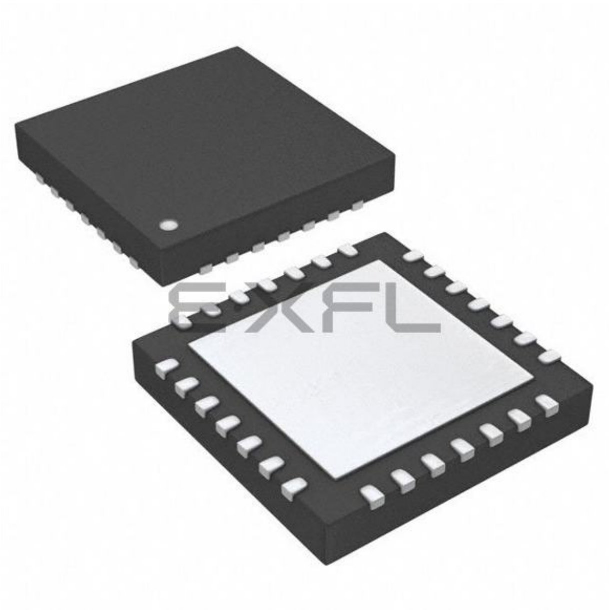
Microchip Technology - PIC16LF19156T-I/MV Datasheet



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, LCD, POR, PWM, WDT
Number of I/O	24
Program Memory Size	28KB (16K x 14)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 20x12b; 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/pic16lf19156t-i-mv

TABLE 4-12: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 PIC16(L)F19155/56/75/76/85/86 (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 62											
CPU CORE REGISTERS; see Table 4-3 for specifics											
1F0Ch	—	Unimplemented								---- ----	---- ----
1F0Dh	—	Unimplemented								---- ----	---- ----
1F0Eh	—	Unimplemented								---- ----	---- ----
1F0Fh	—	Unimplemented								---- ----	---- ----
1F10h	RA0PPS	—	—	—	RA0PPS4	RA0PPS3	RA0PPS2	RA0PPS1	RA0PPS0	0000 0000	--uu uuuu
1F11h	RA1PPS	—	—	—	RA1PPS4	RA1PPS3	RA1PPS2	RA1PPS1	RA1PPS0	0000 0000	--uu uuuu
1F12h	RA2PPS	—	—	—	RA2PPS4	RA2PPS3	RA2PPS2	RA2PPS1	RA2PPS0	0000 0000	--uu uuuu
1F13h	RA3PPS	—	—	—	RA3PPS4	RA3PPS3	RA3PPS2	RA3PPS1	RA3PPS0	0000 0000	--uu uuuu
1F14h	RA4PPS	—	—	—	RA4PPS4	RA4PPS3	RA4PPS2	RA4PPS1	RA4PPS0	0000 0000	--uu uuuu
1F15h	RA5PPS	—	—	—	RA5PPS4	RA5PPS3	RA5PPS2	RA5PPS1	RA5PPS0	0000 0000	--uu uuuu
1F16h	RA6PPS	—	—	—	RA6PPS4	RA6PPS3	RA6PPS2	RA6PPS1	RA6PPS0	0000 0000	--uu uuuu
1F17h	RA7PPS	—	—	—	RA7PPS4	RA7PPS3	RA7PPS2	RA7PPS1	RA7PPS0	0000 0000	--uu uuuu
1F18h	RB0PPS	—	—	—	RB0PPS4	RB0PPS3	RB0PPS2	RB0PPS1	RB0PPS0	0000 0000	--uu uuuu
1F19h	RB1PPS	—	—	—	RB1PPS4	RB1PPS3	RB1PPS2	RB1PPS1	RB1PPS0	0000 0000	--uu uuuu
1F1Ah	RB2PPS	—	—	—	RB2PPS4	RB2PPS3	RB2PPS2	RB2PPS1	RB2PPS0	0000 0000	--uu uuuu
1F1Bh	RB3PPS	—	—	—	RB3PPS4	RB3PPS3	RB3PPS2	RB3PPS1	RB3PPS0	0000 0000	--uu uuuu
1F1Ch	RB4PPS	—	—	—	RB4PPS4	RB4PPS3	RB4PPS2	RB4PPS1	RB4PPS0	0000 0000	--uu uuuu
1F1Dh	RB5PPS	—	—	—	RB5PPS4	RB5PPS3	RB5PPS2	RB5PPS1	RB5PPS0	0000 0000	--uu uuuu
1F1Eh	RB6PPS	—	—	—	RB6PPS4	RB6PPS3	RB6PPS2	RB6PPS1	RB6PPS0	0000 0000	--uu uuuu
1F1Fh	RB7PPS	—	—	—	RB7PPS4	RB7PPS3	RB7PPS2	RB7PPS1	RB7PPS0	0000 0000	--uu uuuu

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

Note 1: Unimplemented data memory locations, read as ‘0’.

5.6 Device ID and Revision ID

The 14-bit Device ID word is located at 8006h and the 14-bit Revision ID is located at 8005h. These locations are read-only and cannot be erased or modified.

Development tools, such as device programmers and debuggers, may be used to read the Device ID, Revision ID and Configuration Words. These locations can also be read from the NVMCON register.

5.7 Register Definitions: Device and Revision

REGISTER 5-6: DEVID: DEVICE ID REGISTER

R	R	R	R	R	R
DEV<13:8>					
bit 13					bit 8

R	R	R	R	R	R	R	R
DEV<7:0>							
bit 7							bit 0

Legend:
R = Readable bit
‘1’ = Bit is set ‘0’ = Bit is cleared

bit 13-0 **DEV<13:0>:** Device ID bits

Device	DEVID<13:0> Values
PIC16F19155/56	11 0000 1001 1110 (309Eh)
PIC16LF19155/56	11 0000 1001 1111 (309Fh)
PIC16F19175/76	11 0000 1010 0000 (30A0h)
PIC16LF19175/76	11 0000 1010 0001 (30A1h)
PIC16F19185/86	11 0000 1010 0010 (30A2h)
PIC16LF19185/86	11 0000 1010 0011 (30A3h)

REGISTER 11-2: CPUDOZE: DOZE AND IDLE REGISTER

R/W-0/0	R/W/HC/HS-0/0	R/W-0/0	R/W/HC/HS-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
IDLEN	DOZEN(1,2)	ROI	DOE	—	DOZE<2:0>		
bit 7							bit 0

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

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

bit 6 **DOZEN:** Doze Enable bit(1)
1 = Places the CPU into DOZE mode
0 = Places the CPU into normal mode of operation

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

bit 4 **DOE:** Doze on Exit bit(1)
1 = Exiting ISR makes DOZEN = 1
0 = Exiting ISR does not change DOZEN

bit 3 **Unimplemented:** Read as ‘0’

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

Note 1: Refer to Table 11-1 for more information.

REGISTER 16-5: PMD4: PMD CONTROL REGISTER 4

R/W-0/0	R/W-0/0	U-0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
UART2MD	UART1MD	—	MSSP1MD	—	—	—	CWG1MD
bit 7							bit 0

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

bit 7 **UART2MD:** Disable EUSART2 bit
1 = EUSART2 module disabled
0 = EUSART2 module enabled

bit 6 **UART1MD:** Disable EUSART1 bit
1 = EUSART1 module disabled
0 = EUSART1 module enabled

bit 5 **Unimplemented:** Read as ‘0’

bit 4 **MSSP1MD:** Disable MSSP1 bit
1 = MSSP1 module disabled
0 = MSSP1 module enabled

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

bit 0 **CWG1MD:** Disable CWG1 bit
1 = CWG1 module disabled
0 = CWG1 module enabled

19.2 ADC Operation

19.2.1 STARTING A CONVERSION

To enable the ADC module, the ON bit of the ADCON0 register must be set to a '1'. A conversion may be started by any of the following:

- Software setting the GO bit of ADCON0 to '1'
- An external trigger (selected by Register 19-3)
- A continuous-mode retrigger (see section **Section 19.5.8 "Continuous Sampling mode"**)

Note:	The GO bit should not be set in the same instruction that turns on the ADC. Refer to **Section 19.2.6 "ADC Conversion Procedure (Basic Mode)"**.

19.2.2 COMPLETION OF A CONVERSION

When any individual conversion is complete, the value already in ADRES is written into PREV (if ADPSIS = 1) and the new conversion results appear in ADRES. When the conversion completes, the ADC module will:

- Clear the GO bit (unless the CONT bit of ADCON0 is set)
- Set the ADIF Interrupt Flag bit
- Set the ADMATH bit
- Update ACC

When ADDSEN = 0 then after every conversion, or when ADDSEN = 1 then after every other conversion, the following events occur:

- ERR is calculated
- ADTIF is set if ERR calculation meets threshold comparison

Importantly, filter and threshold computations occur after the conversion itself is complete. As such, interrupt handlers responding to ADIF should check ADTIF before reading filter and threshold results.

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

19.2.3 ADC OPERATION DURING SLEEP

The ADC module can operate during Sleep. This requires the ADC clock source to be set to the ADCRC option. When the ADCRC oscillator source is selected, the ADC waits one additional instruction before starting the conversion. This allows the SLEEP instruction to be executed, which can reduce system noise during the conversion. If the ADC interrupt is enabled, the device will wake-up from Sleep when the conversion completes. If the ADC interrupt is disabled, the ADC module is turned off after the conversion completes, although the ON bit remains set.

19.2.4 EXTERNAL TRIGGER DURING SLEEP

If the external trigger is received during sleep while ADC clock source is set to the FRC, ADC module will perform the conversion and set the ADIF bit upon completion.

If an external trigger is received when the ADC clock source is something other than FRC, the trigger will be recorded, but the conversion will not begin until the device exits Sleep.

REGISTER 19-27: ADSTPTH: ADC THRESHOLD SETPOINT REGISTER HIGH

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
STPT<15:8>							
bit 7							bit 0

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

bit 7-0 **STPT<15:8>**: ADC Threshold Setpoint MSB. Upper byte of ADC threshold setpoint, depending on ADCALC, may be used to determine ERR, see Register 19-29 for more details.

REGISTER 19-28: ADSTPTL: ADC THRESHOLD SETPOINT REGISTER LOW

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
STPT<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 **STPT<7:0>**: ADC Threshold Setpoint LSB. Lower byte of ADC threshold setpoint, depending on ADCALC, may be used to determine ERR, see Register 19-30 for more details.

REGISTER 19-29: ADERRH: ADC SETPOINT ERROR REGISTER HIGH

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
ERR<15:8>							
bit 7							bit 0

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

bit 7-0 **ERR<15:8>**: ADC Setpoint Error MSB. Upper byte of ADC Setpoint Error. Setpoint Error calculation is determined by ADCALC bits of ADCON3, see Register 19-4 for more details.

REGISTER 24-12: ALRMMTH: ALARM MONTH CONTROL REGISTER

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	ALRMHMONTH	ALRMLMONTH <3:0>			
bit 7							bit 0

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

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

bit 4 **ALRMHMONTH:** Binary Coded Decimal value of months ‘10’ digit; valid value from 0 to 1

bit 3-0 **ALRMLMONTH<3:0>:** Binary Coded Decimal value of months ‘1’ digit; valid value from 0 to 9

REGISTER 24-13: ALRMWD: ALARM WEEKDAY CONTROL REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	ALRMLWDAY<2:0>		
bit 7							bit 0

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

bit 7-3 **Unimplemented:** Read as ‘0’

bit 2-0 **ALRMLWDAY<2:0>:** Binary Coded Decimal value of weekdays ‘1’ digit; valid values from 0 to 6.

REGISTER 24-14: ALRMDAY: ALARM DAY CONTROL REGISTER

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	ALRMHDAY<1:0>		ALRMLDAY<3:0>			
bit 7							bit 0

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

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

bit 5-4 **ALRMHDAY<1:0>:** Binary Coded Decimal value of days ‘10’ digit; valid value from 0 to 3

bit 3-0 **ALRMLDAY<3:0>:** Binary Coded Decimal value of days ‘1’ digit; valid value from 0 to 9

27.5.9 EDGE-TRIGGERED MONOSTABLE MODES

The Edge-Triggered Monostable modes start the timer on an edge from the external Reset signal input, after the ON bit is set, and stop incrementing the timer when the timer matches the PRx period value. The following edges will start the timer:

- Rising edge (MODE<4:0> = `10001`)
- Falling edge (MODE<4:0> = `10010`)
- Rising or Falling edge (MODE<4:0> = `10011`)

When an Edge-Triggered Monostable mode is used in conjunction with the CCP PWM operation the PWM drive goes active with the external Reset signal edge that starts the timer, but will not go active when the timer matches the PRx value. While the timer is incrementing, additional edges on the external Reset signal will not affect the CCP PWM.

28.2 SMT Operation

The core of the module is the 24-bit counter, SMTxTMR combined with a complex data acquisition front-end. Depending on the mode of operation selected, the SMT can perform a variety of measurements summarized in Table .

28.2.1 CLOCK SOURCES

Clock sources available to the SMT include:

- FOSC
- FOSC/4
- HFINTOSC
- MFINTOSC (500 kHz and 31.25 kHz)
- LFINTOSC
- SOSC

The SMT clock source is selected by configuring the CSEL<2:0> bits in the SMTxCLK register. The clock source can also be prescaled using the PS<1:0> bits of the SMTxCON0 register. The prescaled clock source is used to clock both the counter and any synchronization logic used by the module.

28.2.2 PERIOD MATCH INTERRUPT

Similar to other timers, the SMT triggers an interrupt when SMTxTMR rolls over to '0'. This happens when SMTxTMR = SMTxPR, regardless of mode. Hence, in any mode that relies on an external signal or a window to reset the timer, proper operation requires that SMTxPR be set to a period larger than that of the expected signal or window.

28.3 Basic Timer Function Registers

The SMTxTMR time base and the SMTxCPW/SMTxPR/SMTxCPR buffer registers serve several functions and can be manually updated using software.

28.3.1 TIME BASE

The SMTxTMR is the 24-bit counter that is the center of the SMT. It is used as the basic counter/timer for measurement in each of the modes of the SMT. It can be reset to a value of 24'h00_0000 by setting the RST bit of the SMTxSTAT register. It can be written to and read from software, but it is not guarded for atomic access, therefore reads and writes to the SMTxTMR should only be made when the GO = 0, or the software should have other measures to ensure integrity of SMTxTMR reads/writes.

28.3.2 PULSE-WIDTH LATCH REGISTERS

The SMTxCPW registers are the 24-bit SMT pulse width latch. They are used to latch in the value of the SMTxTMR when triggered by various signals, which are determined by the mode the SMT is currently in.

The SMTxCPW registers can also be updated with the current value of the SMTxTMR value by setting the CPWUP bit of the SMTxSTAT register.

28.3.3 PERIOD LATCH REGISTERS

The SMTxCPR registers are the 24-bit SMT period latch. They are used to latch in values of the SMTxTMR when triggered by various other signals, which are determined by the mode the SMT is currently in.

The SMTxCPR registers can also be updated with the current value of the SMTxTMR value by setting the CPRUP bit in the SMTxSTAT register.

28.4 Halt Operation

The counter can be prevented from rolling-over using the STP bit in the SMTxCON0 register. When halting is enabled, the period match interrupt persists until the SMTxTMR is reset (either by a manual Reset, **Section 28.3.1 "Time Base"**) or by clearing the SMTxGO bit of the SMTxCON1 register and writing the SMTxTMR values in software.

28.5 Polarity Control

The three input signals for the SMT have polarity control to determine whether or not they are active high/positive edge or active low/negative edge signals.

The following bits apply to Polarity Control:

- WSEL bit (Window Polarity)
- SSEL bit (Signal Polarity)
- CSEL bit (Clock Polarity)

These bits are located in the SMTxCON0 register.

28.6 Status Information

The SMT provides input status information for the user without requiring the need to deal with the polarity of the incoming signals.

28.6.1 WINDOW STATUS

Window status is determined by the WS bit of the SMTxSTAT register. This bit is only used in Windowed Measure, Gated Counter and Gated Window Measure modes, and is only valid when TS = 1, and will be delayed in time by synchronizer delays in non-Counter modes.

28.6.2 SIGNAL STATUS

Signal status is determined by the AS bit of the SMTxSTAT register. This bit is used in all modes except Window Measure, Time of Flight and Capture modes, and is only valid when TS = 1, and will be delayed in time by synchronizer delays in non-Counter modes.

REGISTER 32-6: CLCxSEL3: GENERIC CLCx DATA 3 SELECT REGISTER

U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	LCxD4S<5:0>					
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 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **LCxD4S<5:0>:** CLCx Data 4 Input Selection bits
See Table 32-2.

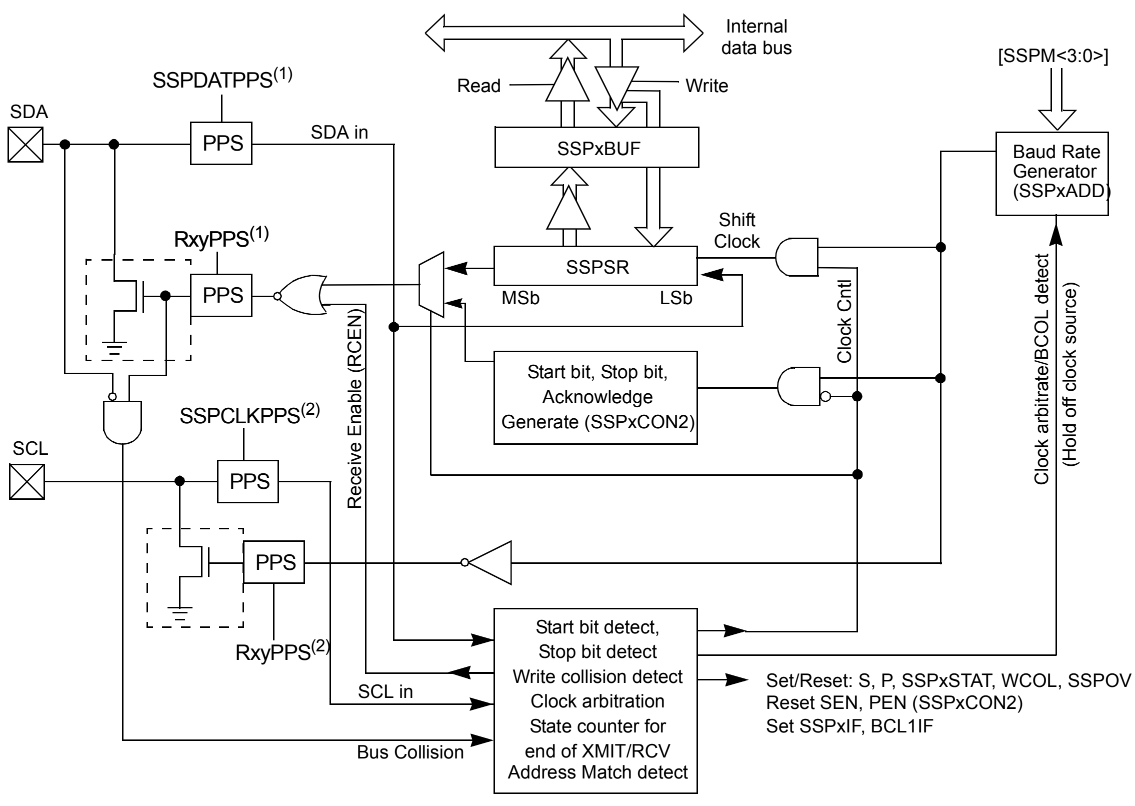
The I^{2}C interface supports the following modes and features:

- Master mode
- Slave mode
- Byte NACKing (Slave mode)
- Limited multi-master support
- 7-bit and 10-bit addressing
- Start and Stop interrupts
- Interrupt masking
- Clock stretching
- Bus collision detection
- General call address matching
- Address masking
- Selectable SDA hold times

Figure 33-2 is a block diagram of the I^{2}C interface module in Master mode. Figure 33-3 is a diagram of the I^{2}C interface module in Slave mode.

Note 1:	In devices with more than one MSSP module, it is very important to pay close attention to SSPxCONx register names. SSPxCON1 and SSPxCON2 registers control different operational aspects of the same module, while SSPxCON1 and SSP2CON1 control the same features for two different modules.
2:	Throughout this section, generic references to an MSSPx module in any of its operating modes may be interpreted as being equally applicable to MSSPx or MSSP2. Register names, module I/O signals, and bit names may use the generic designator ‘x’ to indicate the use of a numeral to distinguish a particular module when required.

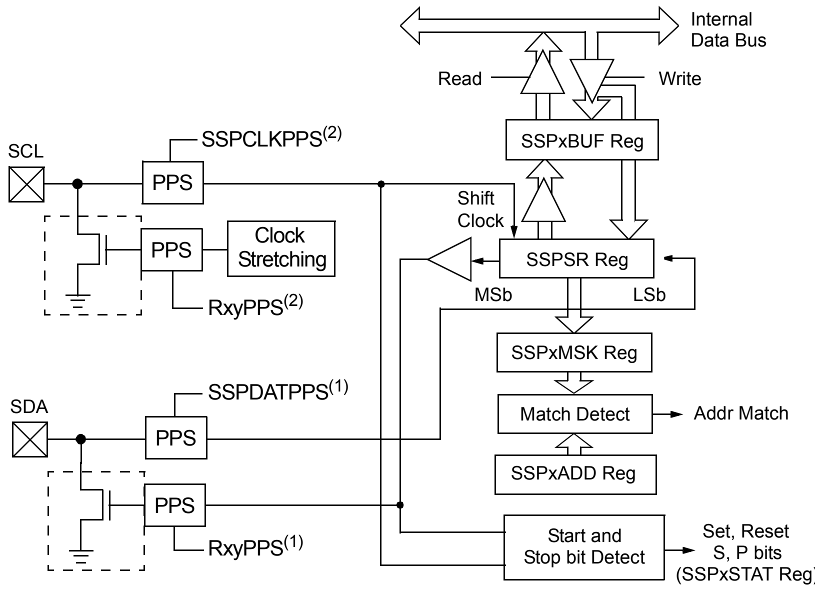
FIGURE 33-2: MSSP BLOCK DIAGRAM (I^{2}C MASTER MODE)



Note 1: SDA pin selections must be the same for input and output

2: SCL pin selections must be the same for input and output

FIGURE 33-3: MSSP BLOCK DIAGRAM (I^2C SLAVE MODE)



Note 1: SDA pin selections must be the same for input and output

2: SCL pin selections must be the same for input and output

REGISTER 34-2: RCxSTA: RECEIVE STATUS AND CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R-0/0	R-0/0	R-0/0
SPEN[(1)]	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
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 **SPEN:** Serial Port Enable bit[(1)]
1 = Serial port enabled
0 = Serial port disabled (held in Reset)

bit 6 **RX9:** 9-Bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN:** Single Receive Enable bit
Asynchronous mode:
Unused in this mode – value ignored
Synchronous mode – Master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.
Synchronous mode – Slave
Unused in this mode – value ignored

bit 4 **CREN:** Continuous Receive Enable bit
Asynchronous mode:
1 = Enables continuous receive until enable bit CREN is cleared
0 = Disables continuous receive
Synchronous mode:
1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN:** Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
1 = Enables address detection – enable interrupt and load of the receive buffer when the ninth bit in the receive buffer is set
0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit
Asynchronous mode 8-bit (RX9 = 0):
Unused in this mode – value ignored

bit 2 **FERR:** Framing Error bit
1 = Framing error (can be updated by reading RCxREG register and receive next valid byte)
0 = No framing error

bit 1 **OERR:** Overrun Error bit
1 = Overrun error (can be cleared by clearing bit CREN)
0 = No overrun error

bit 0 **RX9D:** Ninth bit of Received Data
This can be address/data bit or a parity bit and must be calculated by user firmware.

Note 1: The EUSART module automatically changes the pin from tri-state to drive as needed. Configure the associated TRIS bits for TX/CK and RX/DT to 1.

35.0 LIQUID CRYSTAL DISPLAY (LCD) CONTROLLER

The Liquid Crystal Display (LCD) driver module generates the timing control to drive a static or multiplexed LCD panel. The module drives panels with up to eight commons and up to 38 segments pins. It also provides control of the LCD pixel data.

The LCD driver module supports:

- Direct driving of LCD panel
- Two LCD clock sources with selectable prescaler
- Up to eight commons: See Table 35-1 for multiplexing options.
  - Static (One common)
  - 1/2 Multiplex (two commons)
  - 1/3 Multiplex (three commons)
  - 1/4 Multiplex (four commons)
  - 1/5 Multiplex (five commons)
  - 1/6 Multiplex (six commons)
  - 1/7 Multiplex (seven commons)
  - 1/8 Multiplex (eight commons)
- Static, 1/2 (1/2 Multiplex only) or 1/3 LCD bias (1/3 Multiplex and higher)
- On-chip bias generator with dedicated charge pump to support a range of fixed and variable bias options
- Internal resistors for bias voltage generation
- Software contrast control through internal biasing

A simplified block diagram of the module is shown in Figure 35-1.

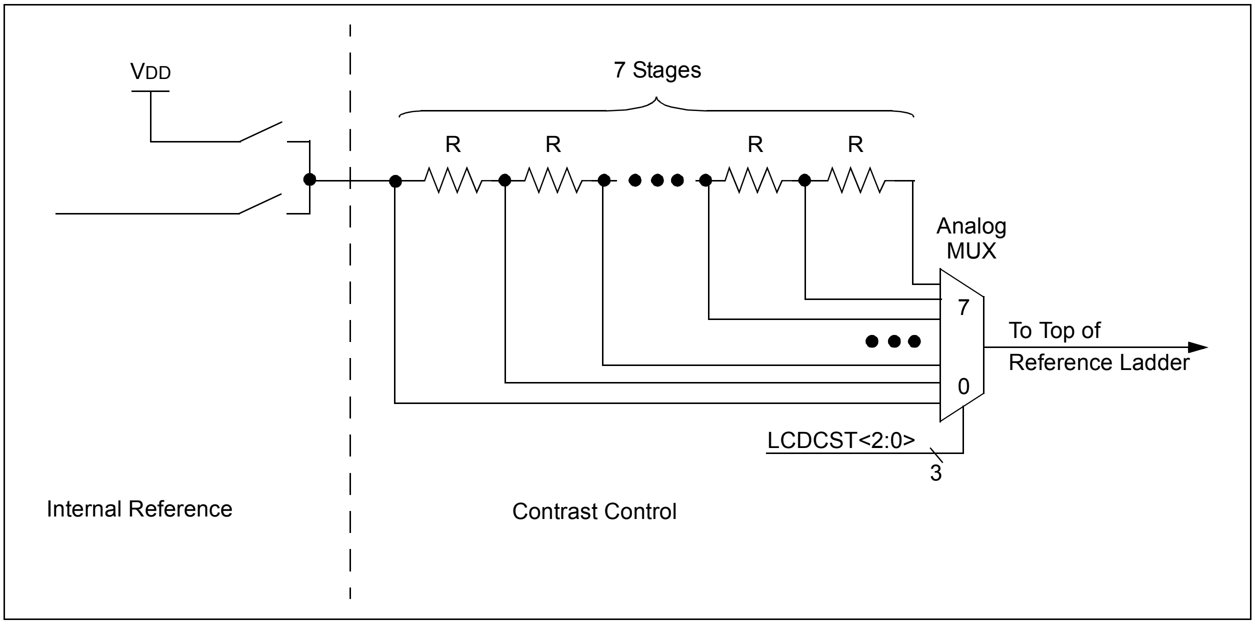
TABLE 35-1: MULTIPLEXING OPTIONS

Device Pin Count	Number of COM Used	Number of SEG Pins Available	Number of Possible Segments
28-Pin	1	19	19
	2	18	36
	3	17	51
	4	16	64
	5	15	75
	6	14	84
	7	13	91
	8	12	96
40-Pin, 44-Pin	1	30	30
	2	29	58
	3	28	84
	4	27	108
	5	26	130
	6	25	150
	7	24	168
	8	23	184
48-Pin	1	38	38
	2	37	74
	3	36	108
	4	35	140
	5	34	170
	6	33	198
	7	32	198
	8	31	248

35.5.3 CONTRAST CONTROL

The LCD contrast control circuit consists of a 7-tap resistor ladder, controlled by the LCDCSTx bits (see Figure 35-4).

FIGURE 35-4: INTERNAL REFERENCE AND CONTRAST CONTROL BLOCK DIAGRAM



35.6 Bias Generation

35.6.1 INTERNAL REFERENCE

An internal reference for the LCD bias voltage can be enabled under firmware control. When enabled, the source of this voltage can be VDD, LCD charge pump or 3x FVR.

When no internal reference is selected, the LCD contrast control circuit is disabled and LCD bias must be provided externally. Whenever the LCD module is inactive (LCDA = 0), the internal reference will be turned off.

35.6.2 VLCDx PINS AND EXTERNAL BIAS

The VLCD3, VLCD2 and VLCD1 pins provide the ability for an external LCD bias network to be used instead of the internal ladder. Use of the VLCDx pins does not prevent use of the internal ladder.

35.6.3 LCD BIAS GENERATION

The LCD driver module is capable of generating the required bias voltages for LCD operation with a minimum of external components. This includes the ability to generate the different voltage levels required by the different bias types that are required by the LCD. The driver module can also provide bias voltages, both above and below microcontroller VDD, through the use of an on-chip LCD charge pump.

35.6.4 LCD CHARGE PUMP

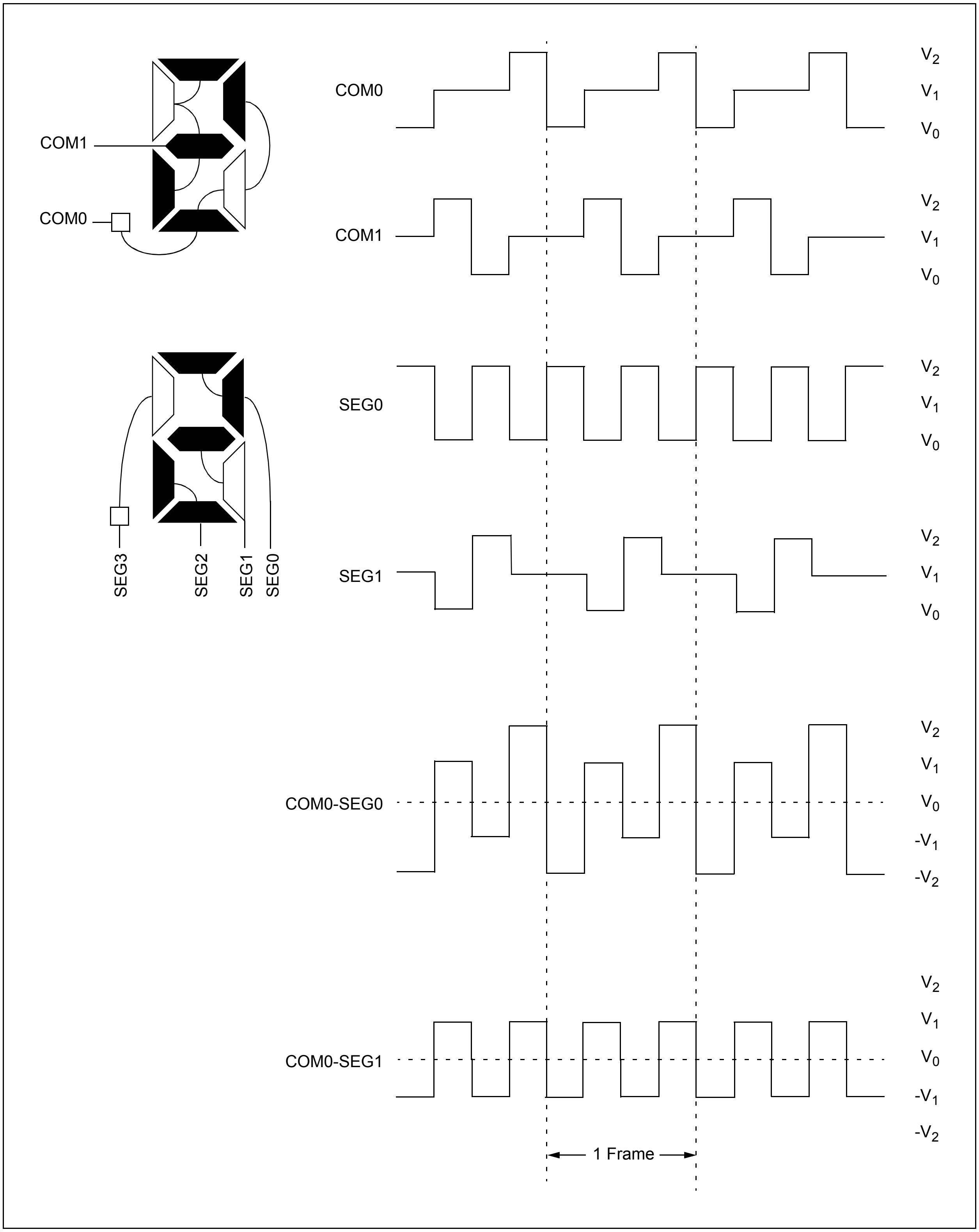
The purpose of the LCD charge pump is to provide proper bias voltage and good contrast for the LCD, regardless of VDD levels. This module contains a charge pump and internal voltage reference. The charge pump can be configured by using external components to boost bias voltage above VDD. It can also operate a display at a constant voltage below VDD. The charge pump can also be selectively disabled to allow bias voltages to be generated by an external resistor network.

The LCD charge pump is controlled through the LCDVCONx registers.

35.6.5 VLCD3 MONITORING

The ADC can be used to measure the VLCD3 voltage via a VLCD3 divided by 4 channel on the ADC. This feature is useful when active adjustment of the LCDCST<2:0> or BIAS<2:0> bits need to be made to account of contrast changes due to extreme temperatures and/or a high number of large active pixels. See **Section 19.0 "Analog-to-Digital Converter with Computation (ADC2) Module"** for additional details.

FIGURE 35-9: TYPE-A WAVEFORMS IN 1/2 MUX, 1/2 BIAS DRIVE



37.3 Instruction Descriptions

ADDFSR — Add Literal to FSRn

Syntax:	[*label*] ADDFSR FSRn, k
Operands:	$-32 \leq k \leq 31$ $n \in [0, 1]$
Operation:	$FSR(n) + k \rightarrow FSR(n)$
Status Affected:	None
Description:	The signed 6-bit literal 'k' is added to the contents of the FSRnH:FSRnL register pair. FSRn is limited to the range 0000h-FFFFh. Moving beyond these bounds will cause the FSR to wrap-around.

ADDLW — Add literal and W

Syntax:	[*label*] ADDLW k
Operands:	$0 \leq k \leq 255$
Operation:	$(W) + k \rightarrow (W)$
Status Affected:	C, DC, Z
Description:	The contents of the W register are added to the 8-bit literal 'k' and the result is placed in the W register.

ADDWF — Add W and f

Syntax:	[*label*] ADDWF f,d
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$
Operation:	(W) + (f) → (destination)
Status Affected:	C, DC, Z
Description:	Add the contents of the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

ADDWFC — ADD W and CARRY bit to f

Syntax:	[*label*] ADDWFC f {,d}
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$
Operation:	(W) + (f) + (C) → dest
Status Affected:	C, DC, Z
Description:	Add W, the Carry flag and data memory location 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in data memory location 'f'.

ANDLW — AND literal with W

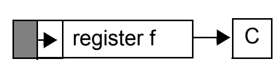
Syntax:	[*label*] ANDLW k
Operands:	$0 \leq k \leq 255$
Operation:	(W) .AND. (k) → (W)
Status Affected:	Z
Description:	The contents of W register are AND'ed with the 8-bit literal 'k'. The result is placed in the W register.

ANDWF — AND W with f

Syntax:	[*label*] ANDWF f,d
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$
Operation:	(W) .AND. (f) → (destination)
Status Affected:	Z
Description:	AND the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

ASRF — Arithmetic Right Shift

Syntax:	[*label*] ASRF f {,d}
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$
Operation:	(f<7>)→ dest<7> (f<7:1>) → dest<6:0>, (f<0>) → C,
Status Affected:	C, Z
Description:	The contents of register 'f' are shifted one bit to the right through the Carry flag. The MSb remains unchanged. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.



BCF — Bit Clear f

Syntax:	[*label*] BCF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	0 → (f<b>)
Status Affected:	None
Description:	Bit ‘b’ in register ‘f’ is cleared.

BRA — Relative Branch

Syntax:	[*label*] BRA label [*label*] BRA $+k
Operands:	$-256 \leq label - PC + 1 \leq 255$ $-256 \leq k \leq 255$
Operation:	(PC) + 1 + k → PC
Status Affected:	None
Description:	Add the signed 9-bit literal ‘k’ to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 1 + k. This instruction is a 2-cycle instruction. This branch has a limited range.

BRW — Relative Branch with W

Syntax:	[*label*] BRW
Operands:	None
Operation:	(PC) + (W) → PC
Status Affected:	None
Description:	Add the contents of W (unsigned) to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 1 + (W). This instruction is a 2-cycle instruction.

BSF — Bit Set f

Syntax:	[*label*] BSF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	1 → (f<b>)
Status Affected:	None
Description:	Bit ‘b’ in register ‘f’ is set.

BTFSC — Bit Test f, Skip if Clear

Syntax:	[*label*] BTFSC f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	skip if (f<b>) = 0
Status Affected:	None
Description:	If bit ‘b’ in register ‘f’ is ‘1’, the next instruction is executed. If bit ‘b’, in register ‘f’, is ‘0’, the next instruction is discarded, and a NOP is executed instead, making this a 2-cycle instruction.

BTFSS — Bit Test f, Skip if Set

Syntax:	[*label*] BTFSS f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b < 7$
Operation:	skip if (f<b>) = 1
Status Affected:	None
Description:	If bit ‘b’ in register ‘f’ is ‘0’, the next instruction is executed. If bit ‘b’ is ‘1’, then the next instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction.

TABLE 38-1: REGISTER FILE SUMMARY FOR PIC16(L)F19155/56/75/76/85/86 DEVICES

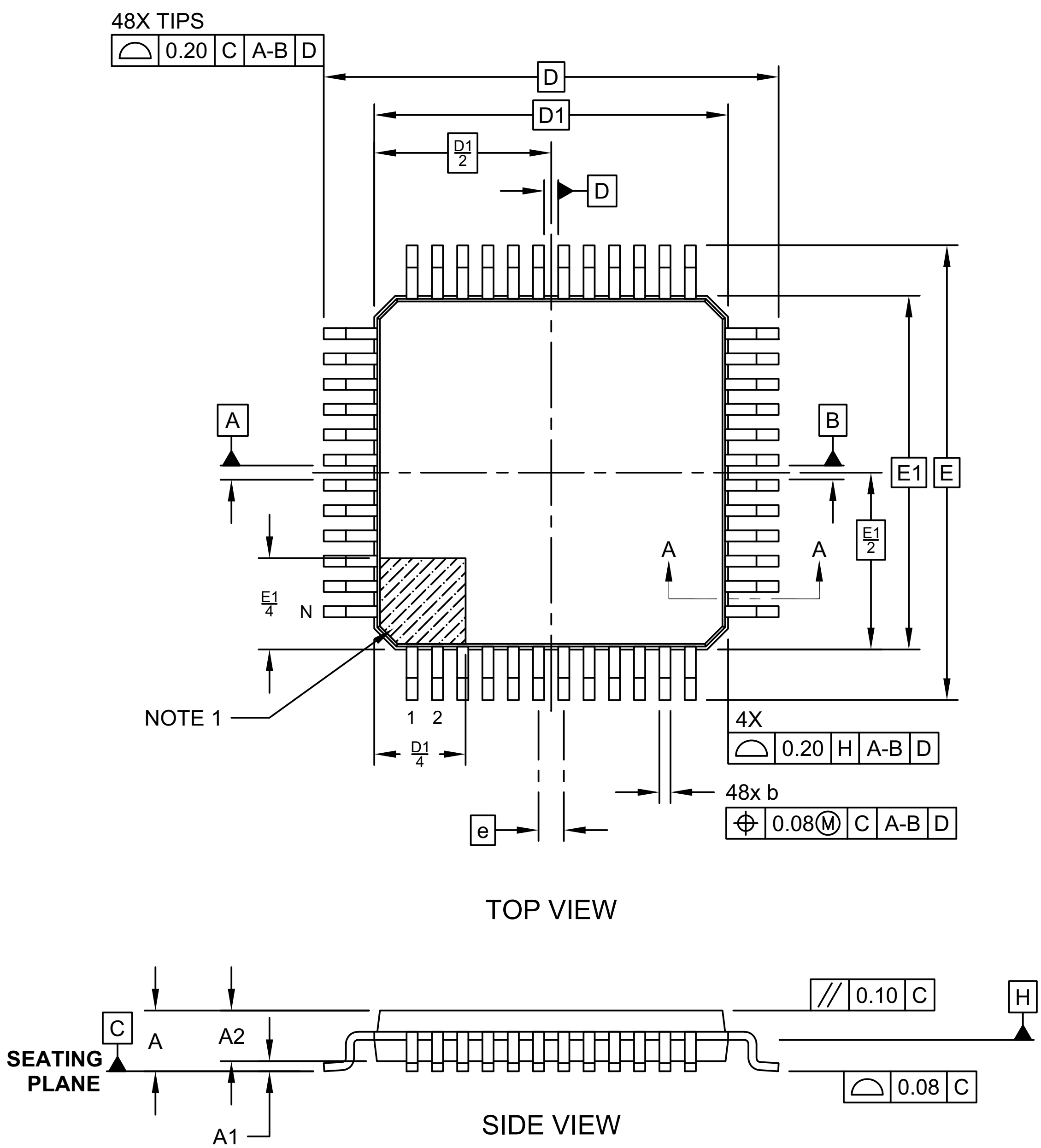
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
994h	CM2CON0	ON	OUT	—	POL	—	—	HYS	SYNC	340
995h	CM2CON1	—	—	—	—	—	—	INTP	INTN	341
996h	CM2NSEL	—	—	—	—	—	NCH<2:0>			342
996h		—	—	—	—	—	NCH2	NCH1	NCH0	342
997h	CM2PSEL	—	—	—	—	—	PCH<2:0>			342
997h		—	—	—	—	—	PCH2	PCH1	PCH0	342
998h	—	Unimplemented								
999h	—	Unimplemented								
99Ah	—	Unimplemented								
99Bh	—	Unimplemented								
99Ch	—	Unimplemented								
99Dh	—	Unimplemented								
99Eh	—	Unimplemented								
99Fh	—	Unimplemented								
A0Ch	—	Unimplemented								
A0Dh	—	Unimplemented								
A0Eh	—	Unimplemented								
A0Fh	—	Unimplemented								
A10h	—	Unimplemented								
A11h	—	Unimplemented								
A12h	—	Unimplemented								
A13h	—	Unimplemented								
A14h	—	Unimplemented								
A15h	—	Unimplemented								
A16h	—	Unimplemented								
A17h	—	Unimplemented								
A18h	—	Unimplemented								
A19h	RC2REG	RC2REG								
A1Ah	TX2REG	TX2REG								
A1Bh	SP2BRGL	SP2BRGL								
A1Ch	SP2BRGH	SP2BRGH								
A1Dh	RC2STA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	
A1Eh	TX2STA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	
A1Fh	BAUD2CON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	
A8Ch — A9Fh	—	Unimplemented								
B0Ch — B1Fh	—	Unimplemented								
B8Ch — B9Fh	—	Unimplemented								

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

Note 1: Unimplemented data memory locations, read as '0'.

48-Lead Thin Quad Flatpack (PT) - 7x7x1.0 mm Body [TQFP]

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



Microchip Technology Drawing C04-300-PT Rev A Sheet 1 of 2