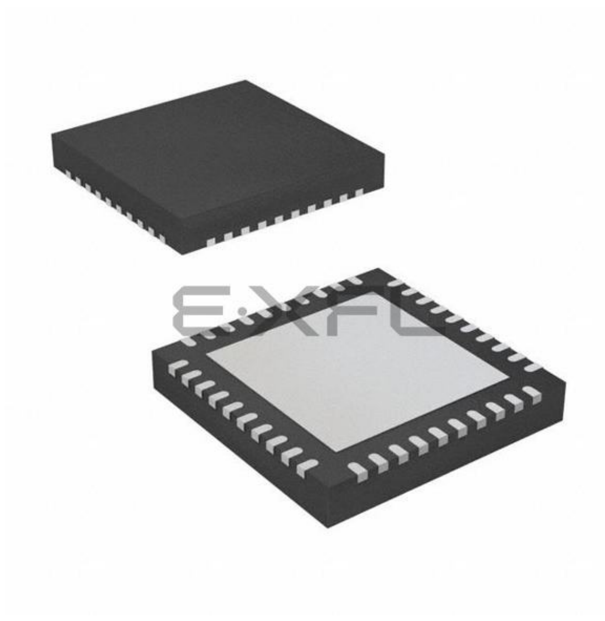


Welcome to E-XFL.COM

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"

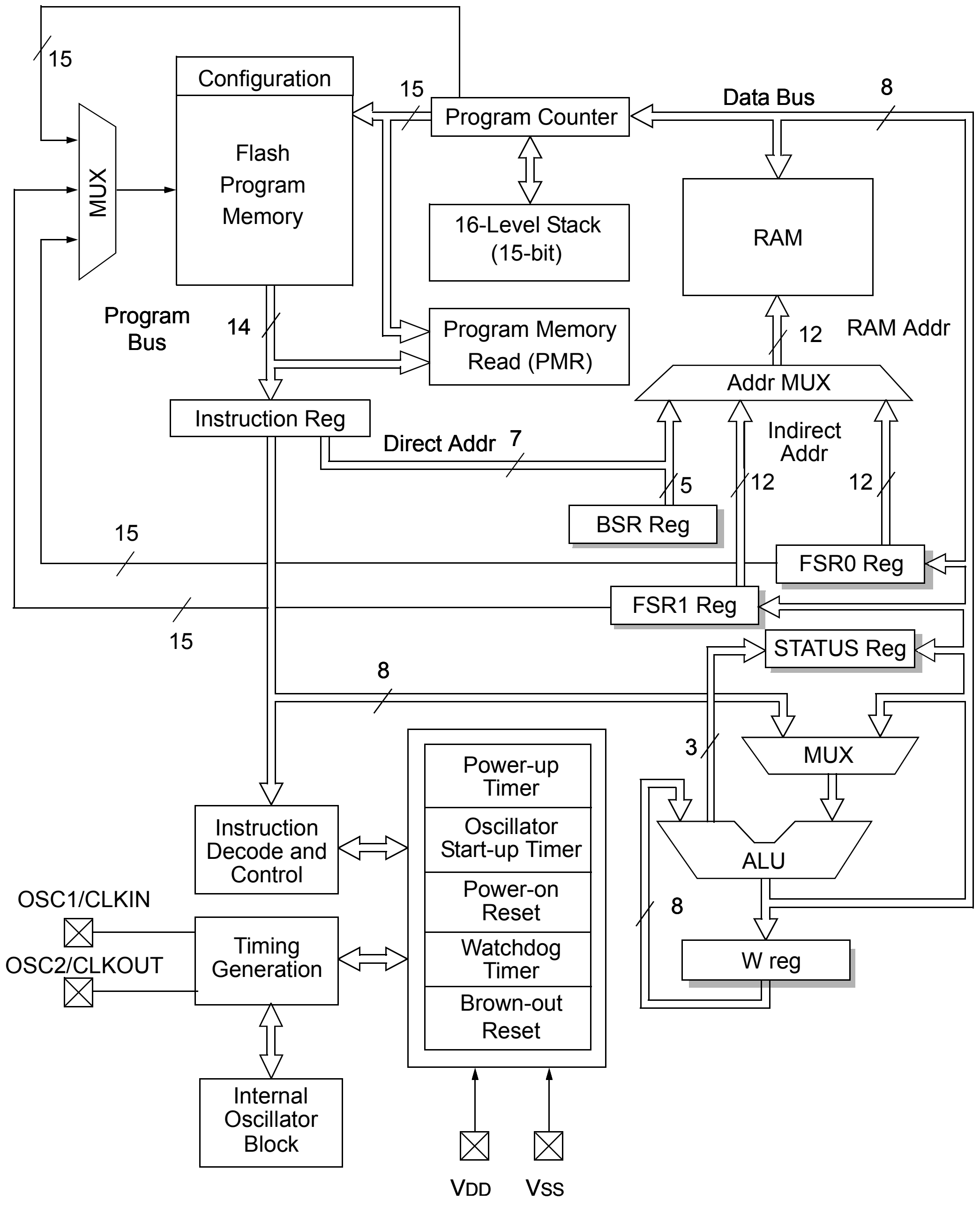
Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	32MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PSMC, PWM, WDT
Number of I/O	35
Program Memory Size	14KB (8K x 14)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 14x12b; D/A 1x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	40-UFQFN Exposed Pad
Supplier Device Package	40-UQFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f1787-e-mv

2.0 ENHANCED MID-RANGE CPU

This family of devices contain an enhanced mid-range 8-bit CPU core. The CPU has 49 instructions. Interrupt capability includes automatic context saving. The hardware stack is 16 levels deep and has Overflow and Underflow Reset capability. Direct, Indirect, and Relative addressing modes are available. Two File Select Registers (FSRs) provide the ability to read program and data memory.

- Automatic Interrupt Context Saving
- 16-level Stack with Overflow and Underflow
- File Select Registers
- Instruction Set

FIGURE 2-1: CORE BLOCK DIAGRAM



3.3.5 CORE FUNCTION REGISTERS SUMMARY

The Core Function registers listed in Table 3-11 can be addressed from any Bank.

TABLE 3-11: CORE FUNCTION REGISTERS SUMMARY

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 0-31											
x00h or x80h	INDF0	Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register)								`xxxx xxxx`	`uuuu uuuu`
x01h or x81h	INDF1	Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register)								`xxxx xxxx`	`uuuu uuuu`
x02h or x82h	PCL	Program Counter (PC) Least Significant Byte								`0000 0000`	`0000 0000`
x03h or x83h	STATUS	—	—	—	$\overline{TO}$	$\overline{PD}$	Z	DC	C	`---1 1000`	`---q quuu`
x04h or x84h	FSR0L	Indirect Data Memory Address 0 Low Pointer								`0000 0000`	`uuuu uuuu`
x05h or x85h	FSR0H	Indirect Data Memory Address 0 High Pointer								`0000 0000`	`0000 0000`
x06h or x86h	FSR1L	Indirect Data Memory Address 1 Low Pointer								`0000 0000`	`uuuu uuuu`
x07h or x87h	FSR1H	Indirect Data Memory Address 1 High Pointer								`0000 0000`	`0000 0000`
x08h or x88h	BSR	—	—	—	BSR4	BSR3	BSR2	BSR1	BSR0	`---0 0000`	`---0 0000`
x09h or x89h	WREG	Working Register								`0000 0000`	`uuuu uuuu`
x0Ah or x8Ah	PCLATH	—	Write Buffer for the upper 7 bits of the Program Counter							`-000 0000`	`-000 0000`
x0Bh or x8Bh	INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	`0000 0000`	`0000 0000`

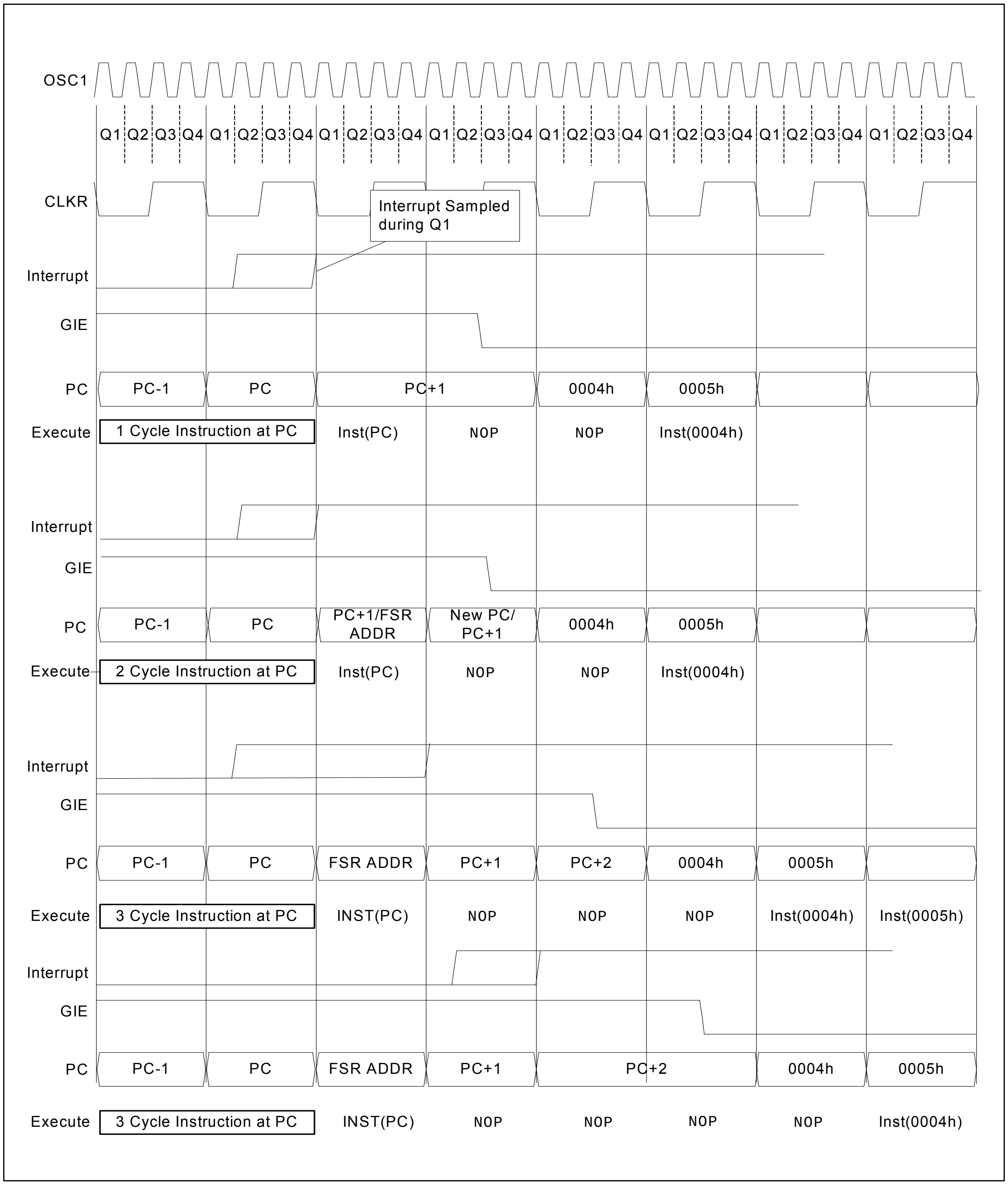
Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

TABLE 5-5: SUMMARY OF REGISTERS ASSOCIATED WITH RESETS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
BORCON	SBOREN	BORFS	—	—	—	—	—	BORRDY	61
PCON	STKOVF	STKUNF	—	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	65
STATUS	—	—	—	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	27
WDTCON	—	—	WDTPS<4:0>					SWDTEN	110

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used by Resets.

FIGURE 8-2: INTERRUPT LATENCY



REGISTER 8-3: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

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
OSFIE	C2IE	C1IE	EEIE	BCL1IE	C4IE	C3IE	CCP2IE
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 **OSFIE:** Oscillator Fail Interrupt Enable bit
`1` = Enables the Oscillator Fail interrupt
`0` = Disables the Oscillator Fail interrupt

bit 6 **C2IE:** Comparator C2 Interrupt Enable bit
`1` = Enables the Comparator C2 interrupt
`0` = Disables the Comparator C2 interrupt

bit 5 **C1IE:** Comparator C1 Interrupt Enable bit
`1` = Enables the Comparator C1 interrupt
`0` = Disables the Comparator C1 interrupt

bit 4 **EEIE:** EEPROM Write Completion Interrupt Enable bit
`1` = Enables the EEPROM Write Completion interrupt
`0` = Disables the EEPROM Write Completion interrupt

bit 3 **BCL1IE:** MSSP Bus Collision Interrupt Enable bit
`1` = Enables the MSSP Bus Collision Interrupt
`0` = Disables the MSSP Bus Collision Interrupt

bit 2 **C4IE:** Comparator C4 Interrupt Enable bit
`1` = Enables the Comparator C4 Interrupt
`0` = Disables the Comparator C4 Interrupt

bit 1 **C3IE:** Comparator C3 Interrupt Enable bit
`1` = Enables the Comparator C3 Interrupt
`0` = Disables the Comparator C3 Interrupt

bit 0 **CCP2IE:** CCP2 Interrupt Enable bit
`1` = Enables the CCP2 interrupt
`0` = Disables the CCP2 interrupt

Note: Bit PEIE of the INTCON register must be set to enable any peripheral interrupt.

TABLE 13-3: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSELA	ANSA7	—	ANSA5	ANSA4	ANSA3	ANSA2	ANSA1	ANSA0	132
INLVLA	INLVLA7	INLVLA6	INLVLA5	INLVLA4	INLVLA3	INLVLA2	INLVLA1	INLVLA0	133
LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	131
ODCONA	ODA7	ODA6	ODA5	ODA4	ODA3	ODA2	ODA1	ODA0	133
OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			198
PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	131
SLRCONA	SLRA7	SLRA6	SLRA5	SLRA4	SLRA3	SLRA2	SLRA1	SLRA0	133
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	131
WPUA	WPUA7	WPUA6	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0	132

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

TABLE 13-4: SUMMARY OF CONFIGURATION WORD WITH PORTA

Name	Bits	Bit -/7	Bit -/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0	Register on Page
CONFIG1	13:8	—	—	FCMEN	IESO	$\overline{\text{CLKOUTEN}}$	BOREN<1:0>		$\overline{\text{CPD}}$	54
	7:0	$\overline{\text{CP}}$	MCLRE	$\overline{\text{PWRTE}}$	WDTE<1:0>		FOSC<2:0>			

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by PORTA.

13.6 Register Definitions: PORTB

REGISTER 13-11: PORTB: PORTB 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
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
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 **RB<7:0>**: PORTB General Purpose I/O Pin bits(1)
1 = Port pin is $\geq$ VIH
0 = Port pin is $\leq$ VIL

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 13-12: TRISB: PORTB TRI-STATE REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
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 **TRISB<7:0>:** PORTB Tri-State Control bits
1 = PORTB pin configured as an input (tri-stated)
0 = PORTB pin configured as an output

REGISTER 13-13: LATB: PORTB 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
LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0
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 **LATB<7:0>**: PORTB Output Latch Value bits(1)

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 13-40: SLRCONE: PORTE SLEW RATE CONTROL REGISTER(1)

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	SLRE2	SLRE1	SLRE0
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 7-3	**Unimplemented:** Read as ‘0’
bit 2-0	**SLRE<2:0>:** PORTE Slew Rate Enable bits For RE<2:0> pins, respectively 1 = Port pin slew rate is limited 0 = Port pin slews at maximum rate

Note 1: SLRE<2:0> are available on PIC16(L)F1784/7 only.

REGISTER 13-41: INLVLE: PORTE INPUT LEVEL CONTROL REGISTER

U-0	U-0	U-0	U-0	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
—	—	—	—	INLVLE3	INLVLE2(1)	INLVLE1(1)	INLVLE0(1)
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	**Unimplemented:** Read as ‘0’
bit 3-0	**INLVLE<3:0>:** PORTE Input Level Select bit(1) 1 = ST input used for PORT reads and interrupt-on-change 0 = TTL input used for PORT reads and interrupt-on-change

Note 1: INLVLE<2:0> are available on PIC16(L)F1784/7 only.

TABLE 13-12: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ADCON0	ADRMD	CHS<4:0>					GO/$\overline{\text{DONE}}$	ADON	172
ANSELE	—	—	—	—	—	ANSE2	ANSE1	ANSE0	153
INLVLE	—	—	—	—	INLVLE3	INLVLE2(2)	INLVLE1(2)	INLVLE0(2)	155
LATE(2)	—	—	—	—	—	LATE2	LATE1	LATE0	153
ODCONE(2)	—	—	—	—	—	ODE2	ODE1	ODE0	154
PORTE	—	—	—	—	RE3	RE2(2)	RE1(2)	RE0(2)	152
SLRCONE(2)	—	—	—	—	—	SLRE2	SLRE1	SLRE0	155
TRISE	—	—	—	—	—(1)	TRISE2(2)	TRISE1(2)	TRISE0(2)	152
WPUE	—	—	—	—	WPUE3	WPUE2(2)	WPUE1(2)	WPUE0(2)	154

Legend: x = unknown, u = unchanged, – = unimplemented locations read as ‘0’. Shaded cells are not used by PORTE.
Note 1: Unimplemented, read as ‘1’.
2: PIC16(L)F1784/7 only

REGISTER 17-3: ADCON2: ADC CONTROL REGISTER 2

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
TRIGSEL<3:0>				CHSN<3:0>			
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 7-4 **TRIGSEL<3:0>:** ADC Auto-conversion Trigger Source Selection bits

`1111` = Reserved. Auto-conversion Trigger disabled.
`1110` = Reserved. Auto-conversion Trigger disabled.
`1101` = Reserved. Auto-conversion Trigger disabled.
`1100` = PSMC3 Falling Match Event(1)
`1011` = PSMC3 Rising Edge Event(1)
`1010` = PSMC3 Period Edge Event(1)
`1001` = PSMC2 Falling Edge Event
`1000` = PSMC2 Rising Edge Event
`0111` = PSMC2 Period Match Event
`0110` = PSMC1 Falling Edge Event
`0101` = PSMC1 Rising Edge Event
`0100` = PSMC1 Period Match Event
`0011` = Reserved. Auto-conversion Trigger disabled.
`0010` = CCP2, Auto-conversion Trigger
`0001` = CCP1, Auto-conversion Trigger
`0000` = Disabled

bit 3-0 **CHSN<3:0>:** Negative Differential Input Channel Select bits

When ADON = `0`, all multiplexer inputs are disconnected.

`1111` = ADC Negative reference - selected by ADNREF
`1110` = AN21(1)
`1101` = AN13
`1100` = AN12
`1011` = AN11
`1010` = AN10
`1001` = AN9
`1000` = AN8
`0111` = AN7(1)
`0110` = AN6(1)
`0101` = AN5(1)
`0100` = AN4
`0011` = AN3
`0010` = AN2
`0001` = AN1
`0000` = AN0

Note 1: PIC16(L)F1784/7 only. For PIC16(L)F1786, “Reserved. No channel connected.”

21.0 TIMER0 MODULE

The Timer0 module is an 8-bit timer/counter with the following features:

- 8-bit timer/counter register (TMR0)
- 8-bit prescaler (independent of Watchdog Timer)
- Programmable internal or external clock source
- Programmable external clock edge selection
- Interrupt on overflow
- TMR0 can be used to gate Timer1

Figure 21-1 is a block diagram of the Timer0 module.

21.1 Timer0 Operation

The Timer0 module can be used as either an 8-bit timer or an 8-bit counter.

21.1.1 8-BIT TIMER MODE

The Timer0 module will increment every instruction cycle, if used without a prescaler. 8-bit Timer mode is selected by clearing the TMR0CS bit of the OPTION_REG register.

When TMR0 is written, the increment is inhibited for two instruction cycles immediately following the write.

Note:	The value written to the TMR0 register can be adjusted, in order to account for the two instruction cycle delay when TMR0 is written.

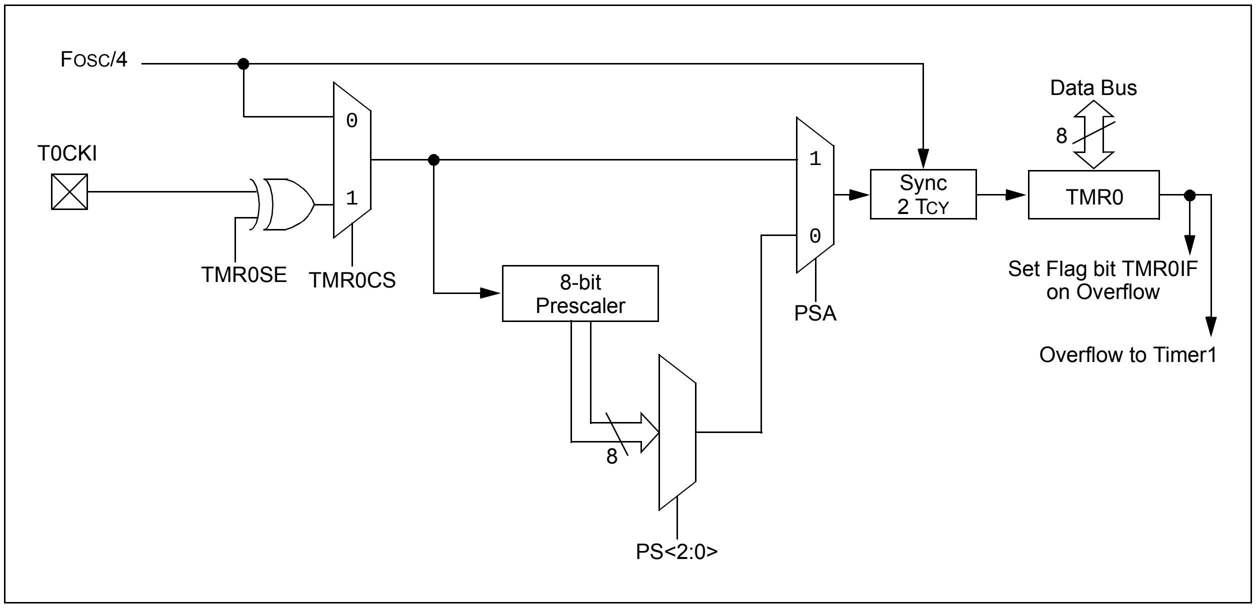
21.1.2 8-BIT COUNTER MODE

In 8-Bit Counter mode, the Timer0 module will increment on every rising or falling edge of the T0CKI pin.

8-Bit Counter mode using the T0CKI pin is selected by setting the TMR0CS bit in the OPTION_REG register to '1'.

The rising or falling transition of the incrementing edge for either input source is determined by the TMR0SE bit in the OPTION_REG register.

FIGURE 21-1: BLOCK DIAGRAM OF THE TIMER0



REGISTER 24-16: PSMCxASDL: PSMC AUTO-SHUTDOWN OUTPUT LEVEL REGISTER

U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	PxASDLF[(1)]	PxASDLE[(1)]	PxASDLD[(1)]	PxASDLC[(1)]	PxASDLB	PxASDLA
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 **PxASDLF:** PSMCx Output F Auto-Shutdown Pin Level bit[(1)]
1 = When auto-shutdown is asserted, pin PSMCxF will drive logic ‘1’
0 = When auto-shutdown is asserted, pin PSMCxF will drive logic ‘0’

bit 4 **PxASDLE:** PSMCx Output E Auto-Shutdown Pin Level bit[(1)]
1 = When auto-shutdown is asserted, pin PSMCxE will drive logic ‘1’
0 = When auto-shutdown is asserted, pin PSMCxE will drive logic ‘0’

bit 3 **PxASDLD:** PSMCx Output D Auto-Shutdown Pin Level bit[(1)]
1 = When auto-shutdown is asserted, pin PSMCxD will drive logic ‘1’
0 = When auto-shutdown is asserted, pin PSMCxD will drive logic ‘0’

bit 2 **PxASDLC:** PSMCx Output C Auto-Shutdown Pin Level bit[(1)]
1 = When auto-shutdown is asserted, pin PSMCxC will drive logic ‘1’
0 = When auto-shutdown is asserted, pin PSMCxC will drive logic ‘0’

bit 1 **PxASDLB:** PSMCx Output B Auto-Shutdown Pin Level bit
1 = When auto-shutdown is asserted, pin PSMCxB will drive logic ‘1’
0 = When auto-shutdown is asserted, pin PSMCxB will drive logic ‘0’

bit 0 **PxASDLA:** PSMCx Output A Auto-Shutdown Pin Level bit
1 = When auto-shutdown is asserted, pin PSMCxA will drive logic ‘1’
0 = When auto-shutdown is asserted, pin PSMCxA will drive logic ‘0’

Note 1: These bits are not implemented on PSMC2.

26.2 SPI Mode Overview

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

The SPI bus specifies four signal connections:

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

Figure 26-1 shows the block diagram of the MSSP module when operating in SPI mode.

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

Figure 26-4 shows a typical connection between a master device and multiple slave devices.

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

Transmissions involve two shift registers, 8 bits in size, one in the master and one in the slave. With either the master or the slave device, data is always shifted out one bit at a time, with the Most Significant bit (MSb) shifted out first. At the same time, a new Least Significant bit (LSb) is shifted into the same register.

Figure 26-5 shows a typical connection between two processors configured as master and slave devices.

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

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

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

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

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

After 8 bits have been shifted out, the master and slave have exchanged register values.

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

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

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

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

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

26.6 I^{2}C 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 TRIS 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 I^{2}C bus may be taken when the P bit is set, or the bus is Idle.

In Firmware Controlled Master mode, user code conducts all I^{2}C 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 I^{2}C Master mode, does not allow queueing 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: When in Master mode, Start/Stop detection is masked and an interrupt is generated when the SEN/PEN bit is cleared and the generation is complete.

26.6.1 I^{2}C MASTER MODE OPERATION

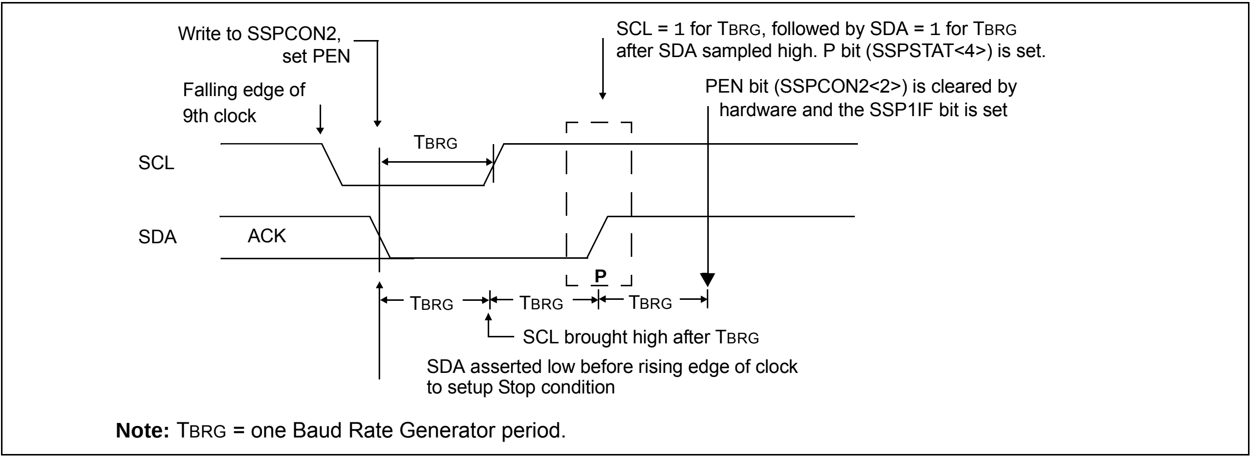
The master device generates all of the serial clock pulses and the Start and Stop conditions. A transfer is ended with a Stop condition or with a Repeated Start condition. Since the Repeated Start condition is also the beginning of the next serial transfer, the I^{2}C bus will not be released.

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/$\overline{\text{Write}}$ (R/$\overline{\text{W}}$) bit. In this case, the R/$\overline{\text{W}}$ bit will be logic ‘0’. Serial data is transmitted 8 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/$\overline{\text{W}}$ bit. In this case, the R/$\overline{\text{W}}$ 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 8 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 26.7 “Baud Rate Generator”** for more detail.

FIGURE 26-31: STOP CONDITION RECEIVE OR TRANSMIT MODE


Write to SSPCON2, set PEN
Falling edge of 9th clock
SCL = 1 for TBRG, followed by SDA = 1 for TBRG after SDA sampled high. P bit (SSPSTAT<4>) is set.
PEN bit (SSPCON2<2>) is cleared by hardware and the SSP1IF bit is set
TBRG
SCL
SDA
ACK
P
TBRG
TBRG
TBRG
SCL brought high after TBRG
SDA asserted low before rising edge of clock to setup Stop condition
Note: TBRG = one Baud Rate Generator period.

26.6.10 SLEEP OPERATION

While in Sleep mode, the I^{2}C slave module can receive addresses or data and when an address match or complete byte transfer occurs, wake the processor from Sleep (if the MSSP interrupt is enabled).

26.6.11 EFFECTS OF A RESET

A Reset disables the MSSP module and terminates the current transfer.

26.6.12 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I^{2}C bus may be taken when the P bit of the SSPSTAT register is set, or the bus is Idle, with both the S and P bits clear. When the bus is busy, enabling the SSP interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDA line must be monitored for arbitration to see if the signal level is the expected output level. This check is performed by hardware with the result placed in the BCL1IF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

26.6.13 MULTI -MASTER COMMUNICATION, BUS COLLISION AND BUS ARBITRATION

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDA pin, arbitration takes place when the master outputs a ‘1’ on SDA, by letting SDA float high and another master asserts a ‘0’. When the SCL pin floats high, data should be stable. If the expected data on SDA is a ‘1’ and the data sampled on the SDA pin is ‘0’, then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCL1IF and reset the I^{2}C port to its Idle state (Figure 26-31).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDA and SCL lines are deasserted and the SSPBUF can be written to. When the user services the bus collision Interrupt Service Routine and if the I^{2}C bus is free, the user can resume communication by asserting a Start condition.

If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are deasserted and the respective control bits in the SSPCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the I^{2}C bus is free, the user can resume communication by asserting a Start condition.

The master will continue to monitor the SDA and SCL pins. If a Stop condition occurs, the SSP1IF bit will be set.

A write to the SSPBUF will start the transmission of data at the first data bit, regardless of where the transmitter left off when the bus collision occurred.

In Multi-Master mode, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the I^{2}C bus can be taken when the P bit is set in the SSPSTAT register, or the bus is Idle and the S and P bits are cleared.

DECFSZ	**Decrement f, Skip if 0**
Syntax:	[*label*] DECFSZ f,d
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$
Operation:	(f) - 1 $\rightarrow$ (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:	$0 \leq k \leq 2047$
Operation:	k $\rightarrow$ PC<10:0> PCLATH<6:3> $\rightarrow$ 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:	$0 \leq f \leq 127$ $d \in [0,1]$
Operation:	(f) + 1 $\rightarrow$ (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:	$0 \leq f \leq 127$ $d \in [0,1]$
Operation:	(f) + 1 $\rightarrow$ (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:	$0 \leq k \leq 255$
Operation:	(W) .OR. k $\rightarrow$ (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:	$0 \leq f \leq 127$ $d \in [0,1]$
Operation:	(W) .OR. (f) $\rightarrow$ (destination)
Status Affected:	$\overline{\text{Z}}$
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’.

MOVIW	**Move INDFn to W**
Syntax:	[*label*] MOVIW ++FSRn [*label*] MOVIW --FSRn [*label*] MOVIW FSRn++ [*label*] MOVIW FSRn-- [*label*] MOVIW k[FSRn]
Operands:	n ∈ [0,1] mm ∈ [00,01, 10, 11] $-32 \leq k \leq 31$
Operation:	INDFn → W Effective address is determined by • FSR + 1 (preincrement) • FSR - 1 (predecrement) • FSR + k (relative offset) After the Move, the FSR value will be either: • FSR + 1 (all increments) • FSR - 1 (all decrements) • Unchanged
Status Affected:	Z

Mode	Syntax	mm
Preincrement	++FSRn	00
Predecrement	--FSRn	01
Postincrement	FSRn++	10
Postdecrement	FSRn--	11

Description:	This instruction is used to move data between W and one of the indirect registers (INDFn). Before/ after this move, the pointer (FSRn) is updated by pre/post incrementing/decrementing it. **Note:** The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the FSRn. FSRn is limited to the range 0000h - FFFFh. Incrementing/ decrementing it beyond these bounds will cause it to wrap-around.

MOVLB	**Move literal to BSR**
Syntax:	[*label*] MOVLB k
Operands:	$0 \leq k \leq 31$
Operation:	k → BSR
Status Affected:	None
Description:	The 5-bit literal 'k' is loaded into the Bank Select Register (BSR).

MOVLP	**Move literal to PCLATH**
Syntax:	[*label*] MOVLP k
Operands:	$0 \leq k \leq 127$
Operation:	k → PCLATH
Status Affected:	None
Description:	The 7-bit literal 'k' is loaded into the PCLATH register.

MOVLW	**Move literal to W**
Syntax:	[*label*] MOVLW k
Operands:	$0 \leq k \leq 255$
Operation:	k → (W)
Status Affected:	None
Description:	The 8-bit literal 'k' is loaded into W register. The "don't cares" will assemble as '0's.
Words:	1
Cycles:	1

Example:	`MOVLW 0x5A`
	After Instruction
	W = 0x5A

MOVWF	**Move W to f**
Syntax:	[*label*] MOVWF f
Operands:	$0 \leq f \leq 127$
Operation:	(W) → (f)
Status Affected:	None
Description:	Move data from W register to register 'f'.
Words:	1
Cycles:	1

Example:	`MOVWF OPTION_REG`
	Before Instruction
	OPTION_REG = 0xFF
	W = 0x4F
	After Instruction
	OPTION_REG = 0x4F
	W = 0x4F

TABLE 30-4: I/O PORTS

Standard Operating Conditions (unless otherwise stated)							
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
	V_{IL}	**Input Low Voltage**					
		I/O PORT:					
D034		with TTL buffer	—	—	0.8	V	$4.5V \le V_{DD} \le 5.5V$
D034A			—	—	0.15 V_{DD}	V	$1.8V \le V_{DD} \le 4.5V$
D035		with Schmitt Trigger buffer	—	—	0.2 V_{DD}	V	$2.0V \le V_{DD} \le 5.5V$
		with I^2C™ levels	—	—	0.3 V_{DD}	V	
		with SMBus levels	—	—	0.8	V	$2.7V \le V_{DD} \le 5.5V$
D036		$\overline{MCLR}$, OSC1 (RC mode)(1)	—	—	0.2 V_{DD}	V	
D036A		OSC1 (HS mode)	—	—	0.3 V_{DD}	V	
	V_{IH}	**Input High Voltage**					
		I/O ports:					
D040		with TTL buffer	2.0	—	—	V	$4.5V \le V_{DD} \le 5.5V$
D040A			0.25 V_{DD} + 0.8	—	—	V	$1.8V \le V_{DD} \le 4.5V$
D041		with Schmitt Trigger buffer	0.8 V_{DD}	—	—	V	$2.0V \le V_{DD} \le 5.5V$
		with I^2C™ levels	0.7 V_{DD}	—	—	V	
		with SMBus levels	2.1	—	—	V	$2.7V \le V_{DD} \le 5.5V$
D042		$\overline{MCLR}$	0.8 V_{DD}	—	—	V	
D043A		OSC1 (HS mode)	0.7 V_{DD}	—	—	V	
D043B		OSC1 (RC mode)	0.9 V_{DD}	—	—	V	**(Note 1)**
	I_{IL}	**Input Leakage Current(2)**					
D060		I/O ports	—	± 5 ± 5	± 125 ± 1000	nA nA	$V_{SS} \le V_{PIN} \le V_{DD}$, Pin at high-impedance @ 85°C 125°C
D061		$\overline{MCLR}$(3)	—	± 50	± 200	nA	$V_{SS} \le V_{PIN} \le V_{DD}$ @ 85°C
	I_{PUR}	**Weak Pull-up Current**					
D070*			25 25	100 140	200 300	μA	V_{DD} = 3.3V, V_{PIN} = V_{SS} V_{DD} = 5.0V, V_{PIN} = V_{SS}
	V_{OL}	**Output Low Voltage(4)**					
D080		I/O ports	—	—	0.6	V	I_{OL} = 8mA, V_{DD} = 5V I_{OL} = 6mA, V_{DD} = 3.3V I_{OL} = 1.8mA, V_{DD} = 1.8V
	V_{OH}	**Output High Voltage(4)**					
D090		I/O ports	V_{DD} - 0.7	—	—	V	I_{OH} = 3.5mA, V_{DD} = 5V I_{OH} = 3mA, V_{DD} = 3.3V I_{OH} = 1mA, V_{DD} = 1.8V

* These parameters are characterized but not tested.

† 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: In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended to use an external clock in RC mode.

2: Negative current is defined as current sourced by the pin.

3: The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

4: Including OSC2 in CLKOUT mode.

Note: Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

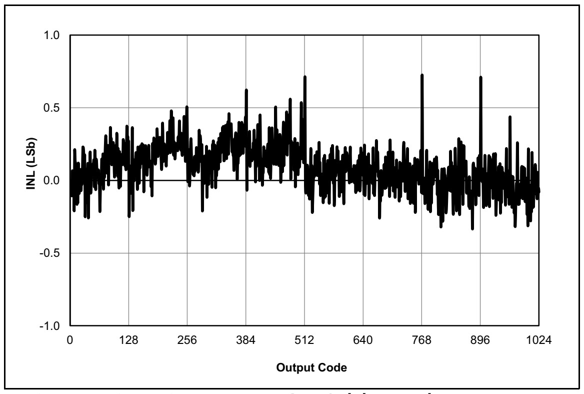


FIGURE 31-79: *ADC 10-bit Mode, Single-Ended INL, V_{DD} = 3.0V, T_{AD} = 1 μS, 25°C.*

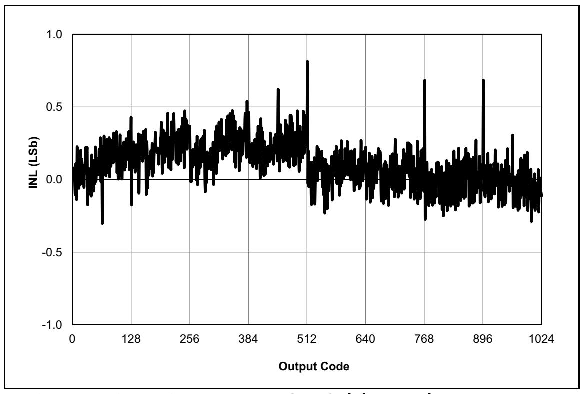


FIGURE 31-80: *ADC 10-bit Mode, Single-Ended INL, V_{DD} = 3.0V, T_{AD} = 4 μS, 25°C.*

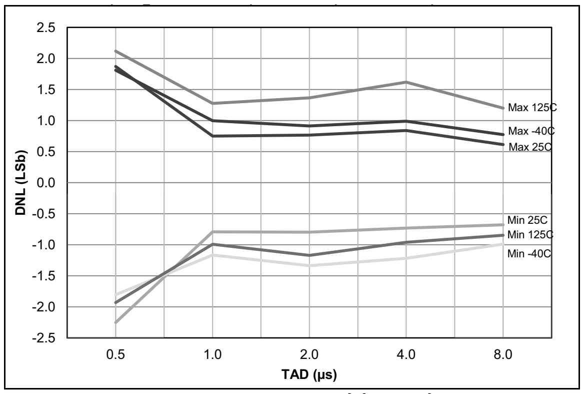


FIGURE 31-81: *ADC 10-bit Mode, Single-Ended DNL, V_{DD} = 3.0V, V_{REF} = 3.0V.*

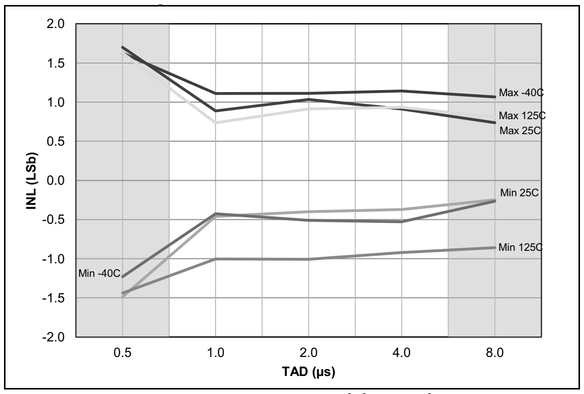


FIGURE 31-82: *ADC 10-bit Mode, Single-Ended INL, V_{DD} = 3.0V, V_{REF} = 3.0V.*

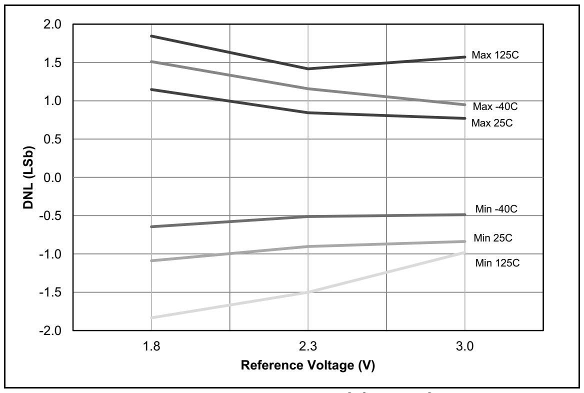


FIGURE 31-83: *ADC 10-bit Mode, Single-Ended DNL, V_{DD} = 3.0V, T_{AD} = 1 μS.*

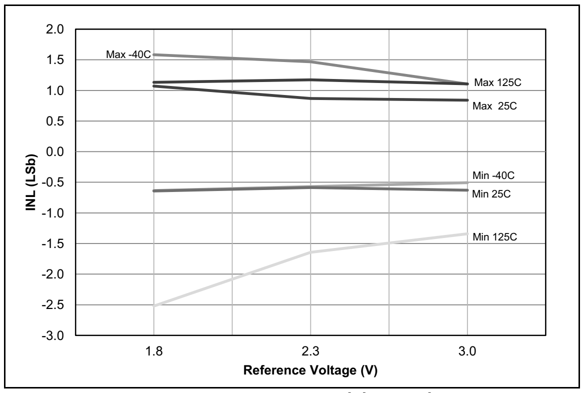


FIGURE 31-84: *ADC 10-bit Mode, Single-Ended INL, V_{DD} = 3.0V, T_{AD} = 1 μS.*

Note: Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

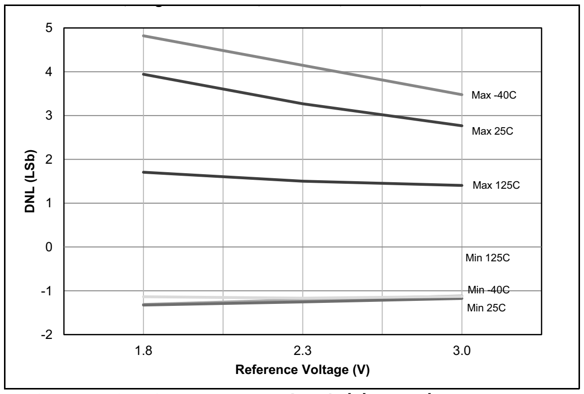


FIGURE 31-91: *ADC 12-bit Mode, Single-Ended DNL, V_{DD} = 3.0V, T_{AD} = 1 µS.*

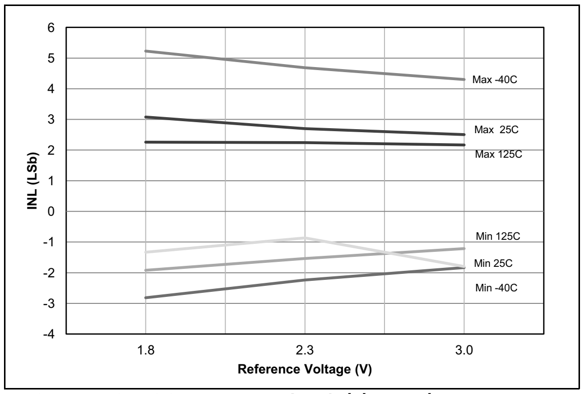


FIGURE 31-92: *ADC 12-bit Mode, Single-Ended INL, V_{DD} = 3.0V, T_{AD} = 1 µS.*

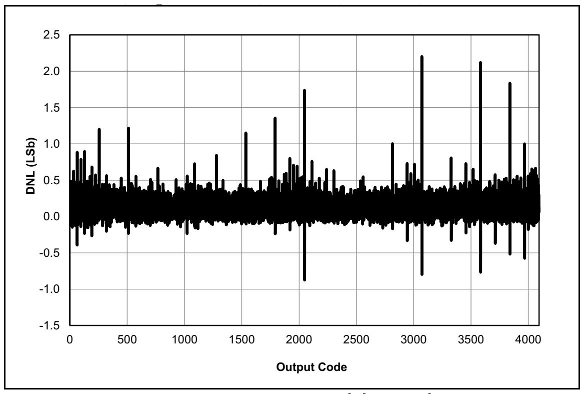


FIGURE 31-93: *ADC 12-bit Mode, Single-Ended DNL, V_{DD} = 5.5V, T_{AD} = 1 µS, 25°C.*

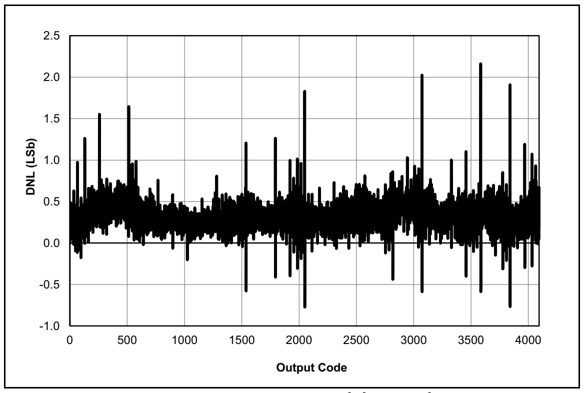


FIGURE 31-94: *ADC 12-bit Mode, Single-Ended DNL, V_{DD} = 5.5V, T_{AD} = 4 µS, 25°C.*

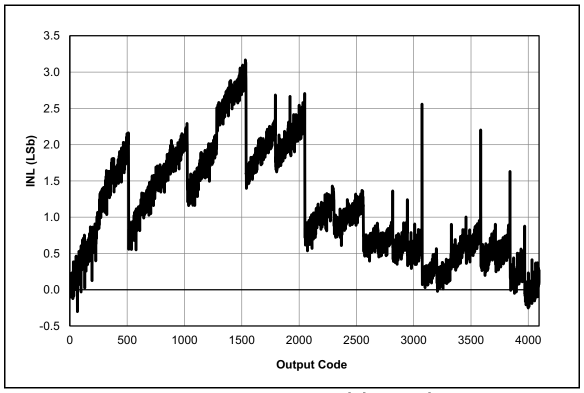


FIGURE 31-95: *ADC 12-bit Mode, Single-Ended INL, V_{DD} = 5.5V, T_{AD} = 1 µS, 25°C.*

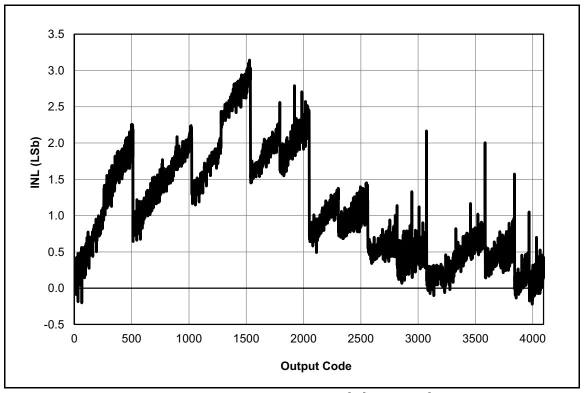


FIGURE 31-96: *ADC 12-bit Mode, Single-Ended INL, V_{DD} = 5.5V, T_{AD} = 4 µS, 25°C.*

APPENDIX A: DATA SHEET REVISION HISTORY

Revision A (06/2012)

Initial release.

Revision B (11/2012)

Minor updates.

Revision C (08/2014)

Change from Preliminary to Final data sheet.

Corrected the following Tables: Family Types Table on page 3, Table 3-3, Table 3-8, Table 20-3, Table 22-2, Table 22-3, Table 23-1, Table 25-3, Table 30-1, Table 30-2, Table 30-3, Table 30-6, Table 30-7, Table 30-13, Table 30-14, Table 30-15, Table 30-16, Table 30-20.

Corrected the following Sections: Section 3.2, Section 9.2, Section 13.3, Section 17.1.6, Section 15.1, Section 15.3, Section 17.2.5, Section 18.2, Section 18.3, Section 19.0, Section 22.6.5, Section 22.9, Section 23.0, Section 23.1, Section 24.2.4, Section 24.2.5, Section 24.2.7, Section 24.8, Section 25.0, Section 26.6.7.4, Section 30.3.

Corrected the following Registers: Register 4-2, Register 8-2, Register 8-5, Register 17-3, Register 18-1, Register 24-3, Register 24-4.

Corrected Equation 17-1.

Corrected Figure 30-9. Removed Figure 24-21.