

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, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	17
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)	1.8V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-VFQFN Exposed Pad
Supplier Device Package	20-QFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f1829t-i-ml

5.2 Clock Source Types

Clock sources can be classified as external or internal.

External clock sources rely on external circuitry for the clock source to function. Examples are: oscillator modules (EC mode), quartz crystal resonators or ceramic resonators (LP, XT and HS modes) and Resistor-Capacitor (RC) mode circuits.

Internal clock sources are contained internally within the oscillator module. The internal oscillator block has two internal oscillators and a dedicated Phase-Lock Loop (HFPLL) that are used to generate three internal system clock sources: the 16 MHz High-Frequency Internal Oscillator (HFINTOSC), 500 kHz (MFINTOSC) and the 31 kHz Low-Frequency Internal Oscillator (LFINTOSC).

The system clock can be selected between external or internal clock sources via the System Clock Select (SCS) bits in the OSCCON register. See **Section 5.3 “Clock Switching”** for additional information.

5.2.1 EXTERNAL CLOCK SOURCES

An external clock source can be used as the device system clock by performing one of the following actions:

- Program the FOSC<2:0> bits in the Configuration Word 1 to select an external clock source that will be used as the default system clock upon a device Reset.
- Write the SCS<1:0> bits in the OSCCON register to switch the system clock source to:
 - Timer1 Oscillator during run-time, or
 - An external clock source determined by the value of the FOSC bits.

See **Section 5.3 “Clock Switching”** for more information.

5.2.1.1 EC Mode

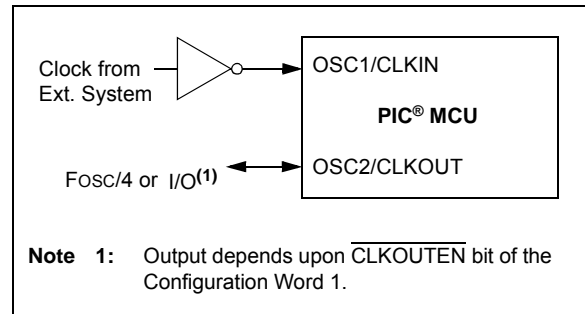
The External Clock (EC) mode allows an externally generated logic level signal to be the system clock source. When operating in this mode, an external clock source is connected to the OSC1 input. OSC2/CLKOUT is available for general purpose I/O or CLKOUT. Figure 5-2 shows the pin connections for EC mode.

EC mode has three power modes to select from through Configuration Word 1:

- High power, 4-32 MHz (FOSC = 111)
- Medium power, 0.5-4 MHz (FOSC = 110)
- Low power, 0-0.5 MHz (FOSC = 101)

The Oscillator Start-up Timer (OST) is disabled when EC mode is selected. Therefore, there is no delay in operation after a Power-on Reset (POR) or wake-up from Sleep. Because the PIC® MCU design is fully static, stopping the external clock input will have the effect of halting the device while leaving all data intact. Upon restarting the external clock, the device will resume operation as if no time had elapsed.

FIGURE 5-2: EXTERNAL CLOCK (EC) MODE OPERATION



5.2.1.2 LP, XT, HS Modes

The LP, XT and HS modes support the use of quartz crystal resonators or ceramic resonators connected to OSC1 and OSC2 (Figure 5-3). The three modes select a low, medium or high gain setting of the internal inverter-amplifier to support various resonator types and speed.

LP Oscillator mode selects the lowest gain setting of the internal inverter-amplifier. LP mode current consumption is the least of the three modes. This mode is designed to drive only 32.768 kHz tuning-fork type crystals (watch crystals).

XT Oscillator mode selects the intermediate gain setting of the internal inverter-amplifier. XT mode current consumption is the medium of the three modes. This mode is best suited to drive resonators with a medium drive level specification.

HS Oscillator mode selects the highest gain setting of the internal inverter-amplifier. HS mode current consumption is the highest of the three modes. This mode is best suited for resonators that require a high drive setting.

Figure 5-3 and Figure 5-4 show typical circuits for quartz crystal and ceramic resonators, respectively.

5.2.2.7 Internal Oscillator Clock Switch Timing

When switching between the HFINTOSC, MFINTOSC and the LFINTOSC, the new oscillator may already be shut down to save power (see Figure 5-7). If this is the case, there is a delay after the IRCF<3:0> bits of the OSCCON register are modified before the frequency selection takes place. The OSCSTAT register will reflect the current active status of the HFINTOSC, MFINTOSC and LFINTOSC oscillators. The sequence of a frequency selection is as follows:

1. IRCF<3:0> bits of the OSCCON register are modified.
2. If the new clock is shut down, a clock start-up delay is started.
3. Clock switch circuitry waits for a falling edge of the current clock.
4. The current clock is held low and the clock switch circuitry waits for a rising edge in the new clock.
5. The new clock is now active.
6. The OSCSTAT register is updated as required.
7. Clock switch is complete.

See Figure 5-7 for more details.

If the internal oscillator speed is switched between two clocks of the same source, there is no start-up delay before the new frequency is selected. Clock switching time delays are shown in Table 5-1.

Start-up delay specifications are located in the oscillator tables of **Section 30.0 “Electrical Specifications”**

7.10 Determining the Cause of a Reset

Upon any Reset, multiple bits in the STATUS and PCON register are updated to indicate the cause of the Reset. Table 7-3 and Table 7-4 show the Reset conditions of these registers.

TABLE 7-3: RESET STATUS BITS AND THEIR SIGNIFICANCE

STKOVF	STKUNF	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Condition
0	0	1	1	0	x	1	1	Power-on Reset
0	0	1	1	0	x	0	x	Illegal, $\overline{\text{TO}}$ is set on $\overline{\text{POR}}$
0	0	1	1	0	x	x	0	Illegal, $\overline{\text{PD}}$ is set on $\overline{\text{POR}}$
0	0	1	1	u	0	1	1	Brown-out Reset
u	u	u	u	u	u	0	u	WDT Reset
u	u	u	u	u	u	0	0	WDT Wake-up from Sleep
u	u	u	u	u	u	1	0	Interrupt Wake-up from Sleep
u	u	0	u	u	u	u	u	$\overline{\text{MCLR}}$ Reset during normal operation
u	u	0	u	u	u	1	0	$\overline{\text{MCLR}}$ Reset during Sleep
u	u	u	0	u	u	u	u	RESET Instruction Executed
1	u	u	u	u	u	u	u	Stack Overflow Reset (STVREN = 1)
u	1	u	u	u	u	u	u	Stack Underflow Reset (STVREN = 1)

TABLE 7-4: RESET CONDITION FOR SPECIAL REGISTERS⁽²⁾

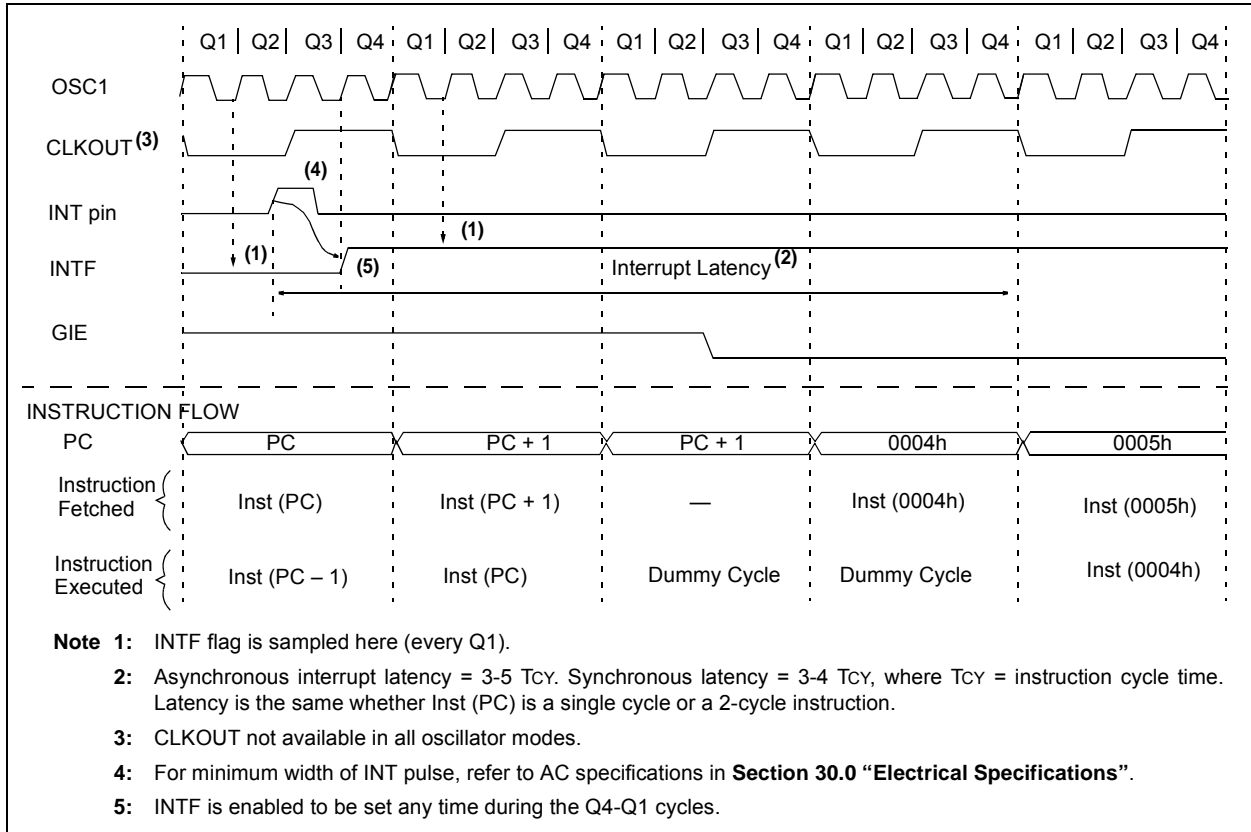
Condition	Program Counter	STATUS Register	PCON Register
Power-on Reset	0000h	---1 1000	00-- 110x
$\overline{\text{MCLR}}$ Reset during normal operation	0000h	---u uuuu	uu-- 0uuu
$\overline{\text{MCLR}}$ Reset during Sleep	0000h	---1 0uuu	uu-- 0uuu
WDT Reset	0000h	---0 uuuu	uu-- uuuu
WDT Wake-up from Sleep	PC + 1	---0 0uuu	uu-- uuuu
Brown-out Reset	0000h	---1 1uuu	00-- 11u0
Interrupt Wake-up from Sleep	PC + 1 ⁽¹⁾	---1 0uuu	uu-- uuuu
RESET Instruction Executed	0000h	---u uuuu	uu-- u0uu
Stack Overflow Reset (STVREN = 1)	0000h	---u uuuu	1u-- uuuu
Stack Underflow Reset (STVREN = 1)	0000h	---u uuuu	u1-- uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, reads as '0'.

Note 1: When the wake-up is due to an interrupt and Global Enable bit (GIE) is set, the return address is pushed on the stack and PC is loaded with the interrupt vector (0004h) after execution of PC + 1.

2: If a Status bit is not implemented, that bit will be read as '0'.

FIGURE 8-3: INT PIN INTERRUPT TIMING



PIC16(L)F1825/9

8.6.8 PIR3 REGISTER

The PIR3 register contains the interrupt flag bits, as shown in Register 8-8.

Note 1: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable bit, GIE, of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 8-8: PIR3: PERIPHERAL INTERRUPT REQUEST REGISTER 3

U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	U-0
—	—	CCP4IF	CCP3IF	TMR6IF	—	TMR4IF	—
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 **CCP4IF:** CCP4 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 4 **CCP3IF:** CCP3 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 3 **TMR6IF:** TMR6 to PR6 Match Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 2 **Unimplemented:** Read as '0'
- bit 1 **TMR4IF:** TMR4 to PR4 Match Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 0 **Unimplemented:** Read as '0'

PIC16(L)F1825/9

9.0 POWER-DOWN MODE (SLEEP)

The Power-Down mode is entered by executing a `SLEEP` instruction.

Upon entering Sleep mode, the following conditions exist:

1. WDT will be cleared but keeps running, if enabled for operation during Sleep.
2. \overline{PD} bit of the STATUS register is cleared.
3. \overline{TO} bit of the STATUS register is set.
4. CPU clock is disabled.
5. 31 kHz LFINTOSC is unaffected and peripherals that operate from it may continue operation in Sleep.
6. Timer1 oscillator is unaffected and peripherals that operate from it may continue operation in Sleep.
7. ADC is unaffected, if the dedicated FRC clock is selected.
8. Capacitive Sensing oscillator is unaffected.
9. I/O ports maintain the status they had before `SLEEP` was executed (driving high, low or high-impedance).
10. Resets other than WDT are not affected by Sleep mode.

Refer to individual chapters for more details on peripheral operation during Sleep.

To minimize current consumption, the following conditions should be considered:

- I/O pins should not be floating
- External circuitry sinking current from I/O pins
- Internal circuitry sourcing current from I/O pins
- Current draw from pins with internal weak pull-ups
- Modules using 31 kHz LFINTOSC
- Modules using Timer1 oscillator

I/O pins that are high-impedance inputs should be pulled to V_{DD} or V_{SS} externally to avoid switching currents caused by floating inputs.

Examples of internal circuitry that might be sourcing current include modules such as the DAC and FVR modules. See **Section 17.0 “Digital-to-Analog Converter (DAC) Module”** and **Section 14.0 “Fixed Voltage Reference (FVR)”** for more information on these modules.

9.1 Wake-up from Sleep

The device can wake-up from Sleep through one of the following events:

1. External Reset input on \overline{MCLR} pin, if enabled
2. BOR Reset, if enabled
3. POR Reset
4. Watchdog Timer, if enabled
5. Any external interrupt
6. Interrupts by peripherals capable of running during Sleep (see individual peripheral for more information)

The first three events will cause a device Reset. The last three events are considered a continuation of program execution. To determine whether a device Reset or wake-up event occurred, refer to **Section 7.10 “Determining the Cause of a Reset”**.

When the `SLEEP` instruction is being executed, the next instruction ($PC + 1$) is prefetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be enabled. Wake-up will occur regardless of the state of the GIE bit. If the GIE bit is disabled, the device continues execution at the instruction after the `SLEEP` instruction. If the GIE bit is enabled, the device executes the instruction after the `SLEEP` instruction, the device will then call the Interrupt Service Routine. In cases where the execution of the instruction following `SLEEP` is not desirable, the user should have a `NOP` after the `SLEEP` instruction.

The WDT is cleared when the device wakes up from Sleep, regardless of the source of wake-up.

9.1.1 WAKE-UP USING INTERRUPTS

When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the execution of a `SLEEP` instruction
 - `SLEEP` instruction will execute as a `NOP`.
 - WDT and WDT prescaler will not be cleared
 - \overline{TO} bit of the STATUS register will not be set
 - \overline{PD} bit of the STATUS register will not be cleared.
- If the interrupt occurs **during or after** the execution of a `SLEEP` instruction
 - `SLEEP` instruction will be completely executed
 - Device will immediately wake-up from Sleep
 - WDT and WDT prescaler will be cleared
 - \overline{TO} bit of the STATUS register will be set
 - \overline{PD} bit of the STATUS register will be cleared.

12.2.2 PORTA FUNCTIONS AND OUTPUT PRIORITIES

Each PORTA pin is multiplexed with other functions. The pins, their combined functions and their output priorities are briefly described here. For additional information, refer to the appropriate section in this data sheet.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the lowest number in the following lists.

Analog input functions, such as ADC, comparator and Cap Sense inputs, are not shown in the priority lists. These inputs are active when the I/O pin is set for Analog mode using the ANSELx registers. Digital output functions may control the pin when it is in Analog mode with the priority shown below.

TABLE 12-2: PORTA OUTPUT PRIORITY

Pin Name	Function Priority ⁽¹⁾
RA0	ICSPDAT ICDDAT DACOUT
RA1	ICSPCLK ICDCLK RX/DT ⁽²⁾
RA2	SRQ C1OUT CCP3
RA3	None (input only)
RA4	CLKOUT T1OSO CLKR SDO1 P2B ⁽²⁾
RA5	SDO2 (PIC16(L)F1829 only) CCP2 ⁽²⁾ /P2A ⁽²⁾

- Note 1:** Priority listed from highest to lowest.
Note 2: Pin function is selectable via the APFCON0 or APFCON1 register.

PIC16(L)F1825/9

17.7 DAC Control Registers

REGISTER 17-1: DACCON0: VOLTAGE REFERENCE CONTROL REGISTER 0

R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	U-0	R/W-0/0
DACEN	DACLPS	DACOE	—	DACPSS<1:0>		—	DACNSS
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 **DACEN:** DAC Enable bit
1 = DAC is enabled
0 = DAC is disabled
- bit 6 **DACLPS:** DAC Low-Power Voltage State Select bit
1 = DAC Positive reference source selected
0 = DAC Negative reference source selected
- bit 5 **DACOE:** DAC Voltage Output Enable bit
1 = DAC voltage level is also an output on the DACOUT pin
0 = DAC voltage level is disconnected from the DACOUT pin
- bit 4 **Unimplemented:** Read as '0'
- bit 3-2 **DACPSS<1:0>:** DAC Positive Source Select bits
00 = VDD
01 = VREF+
10 = FVR Buffer2 output
11 = Reserved, do not use
- bit 1 **Unimplemented:** Read as '0'
- bit 0 **DACNSS:** DAC Negative Source Select bits
1 = VREF-
0 = VSS

REGISTER 17-2: DACCON1: VOLTAGE REFERENCE CONTROL REGISTER 1

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	DACR<4:0>				
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 7-5 **Unimplemented:** Read as '0'
- bit 4-0 **DACR<4:0>:** DAC Voltage Output Select bits
 $V_{OUT} = ((V_{SRC+}) - (V_{SRC-})) * (DACR_{<4:0>} / (2^5)) + V_{SRC-}$

Note 1: The output select bits are always right justified to ensure that any number of bits can be used without affecting the register layout

PIC16(L)F1825/9

FIGURE 21-3: TIMER1 GATE ENABLE MODE

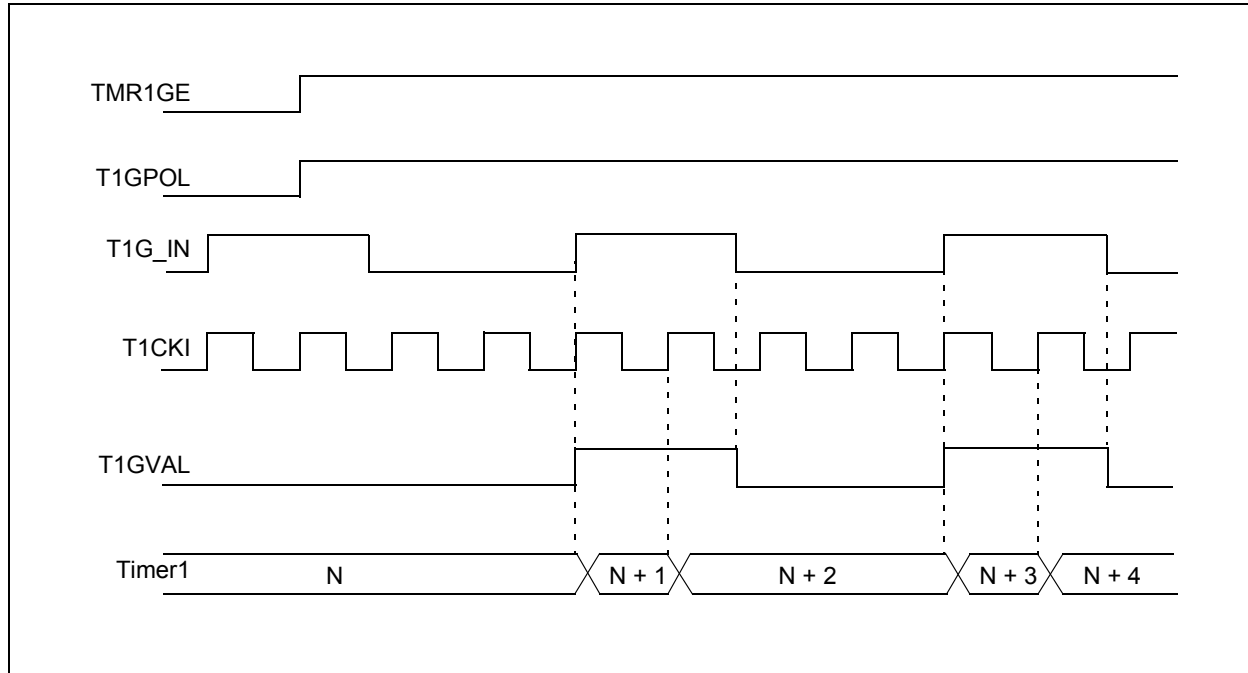


FIGURE 21-4: TIMER1 GATE TOGGLE MODE

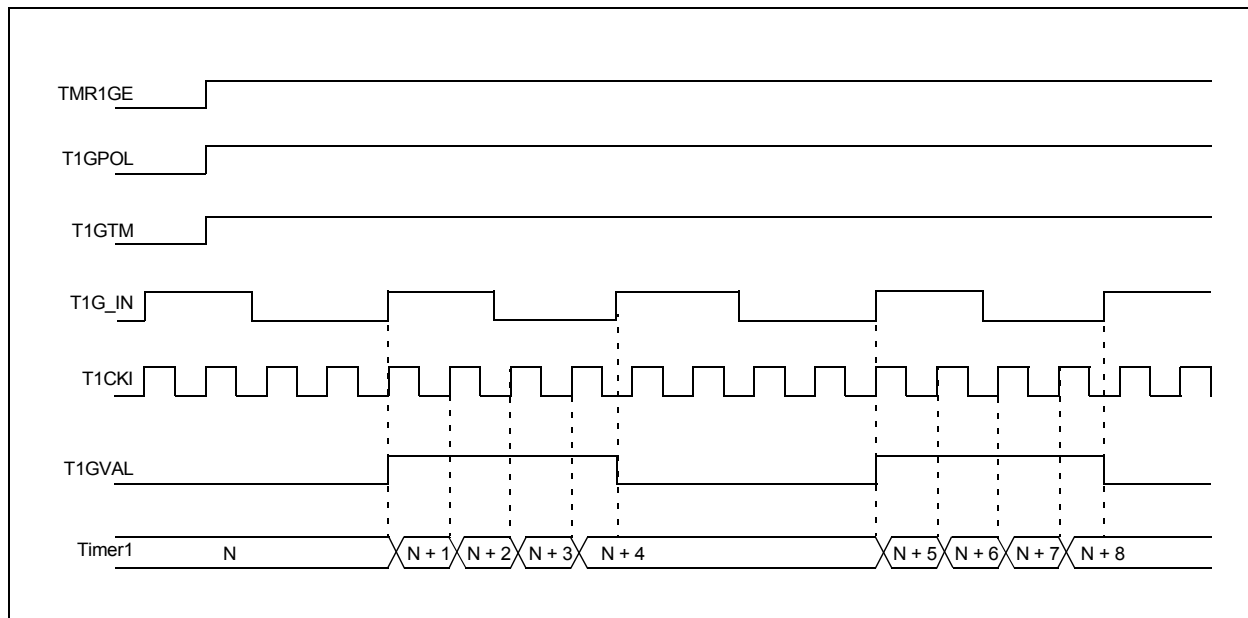


TABLE 21-5: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER1

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSELA	—	—	—	ANSA4	—	ANSA2	ANSA1	ANSA0	123
CCP1CON	P1M<1:0>		DC1B<1:0>		CCP1M<3:0>				224
CCP2CON	P2M<1:0>		DC2B<1:0>		CCP2M<3:0>				224
INLVLA	—	—	INLVLA5	INLVLA4	INLVLA3	INLVLA2	INLVLA1	INLVLA0	124
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	87
PIE1	TMR1GIE	ADIE	RCIE	TXIE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	88
PIR1	TMR1GIF	ADIF	RCIF	TXIF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	92
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								181*
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								181*
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	122
T1CON	TMR1CS<1:0>		T1CKPS<1:0>		T1OSCEN	$\overline{T1SYNC}$	—	TMR1ON	185
T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	$\overline{T1GGO/DONE}$	T1GVAL	T1GSS<1:0>		186

Legend: — Unimplemented location, read as '0'. Shaded cells are not used by the Timer1 module.
 * Page provides register information.

PIC16(L)F1825/9

24.3 PWM Overview

Pulse-Width Modulation (PWM) is a scheme that provides power to a load by switching quickly between fully on and fully off states. The PWM signal resembles a square wave where the high portion of the signal is considered the on state and the low portion of the signal is considered the off state. The high portion, also known as the pulse width, can vary in time and is defined in steps. A larger number of steps applied, which lengthens the pulse width, also supplies more power to the load. Lowering the number of steps applied, which shortens the pulse width, supplies less power. The PWM period is defined as the duration of one complete cycle or the total amount of on and off time combined.

PWM resolution defines the maximum number of steps that can be present in a single PWM period. A higher resolution allows for more precise control of the pulse width time and in turn the power that is applied to the load.

The term duty cycle describes the proportion of the on time to the off time and is expressed in percentages, where 0% is fully off and 100% is fully on. A lower duty cycle corresponds to less power applied and a higher duty cycle corresponds to more power applied.

Figure 24-3 shows a typical waveform of the PWM signal.

24.3.1 STANDARD PWM OPERATION

The standard PWM function described in this section is available and identical for CCP modules ECCP1, ECCP2, CCP3 and CCP4.

The standard PWM mode generates a Pulse-Width Modulation (PWM) signal on the CCPx pin with up to 10 bits of resolution. The period, duty cycle, and resolution are controlled by the following registers:

- PRx registers
- TxCON registers
- CCPRxL registers
- CCPxCON registers

Figure 24-4 shows a simplified block diagram of PWM operation.

Note 1: The corresponding TRIS bit must be cleared to enable the PWM output on the CCPx pin.

2: Clearing the CCPxCON register will relinquish control of the CCPx pin.

FIGURE 24-3: CCP PWM OUTPUT SIGNAL

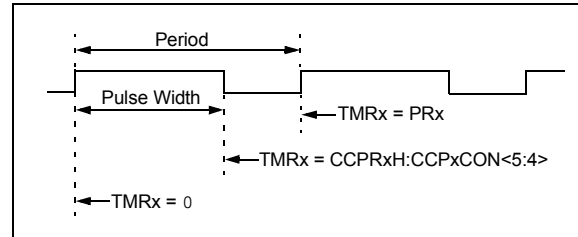
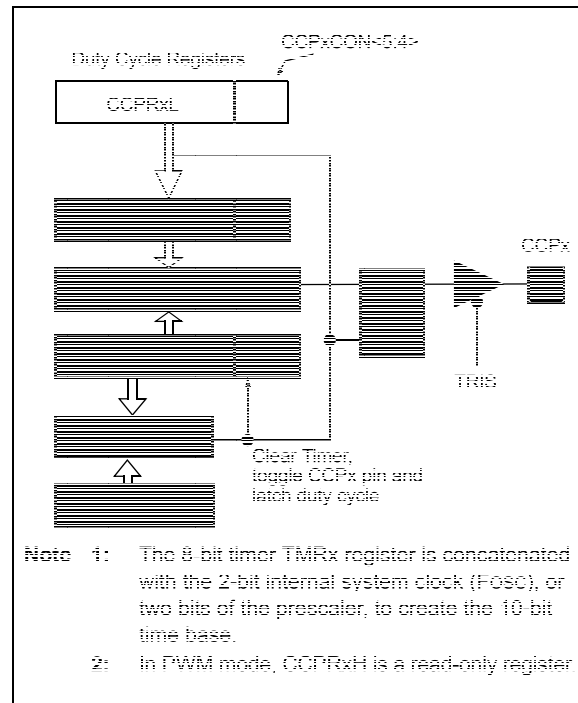


FIGURE 24-4: SIMPLIFIED PWM BLOCK DIAGRAM



PIC16(L)F1825/9

24.3.6 PWM RESOLUTION

The resolution determines the number of available duty cycles for a given period. For example, a 10-bit resolution will result in 1024 discrete duty cycles, whereas an 8-bit resolution will result in 256 discrete duty cycles.

The maximum PWM resolution is ten bits when PRx is 255. The resolution is a function of the PRx register value as shown by Equation 24-4.

EQUATION 24-4: PWM RESOLUTION

$$Resolution = \frac{\log[4(PRx + 1)]}{\log(2)} \text{ bits}$$

Note: If the pulse width value is greater than the period the assigned PWM pin(s) will remain unchanged.

TABLE 24-5: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (Fosc = 32 MHz)

PWM Frequency	1.95 kHz	7.81 kHz	31.25 kHz	125 kHz	250 kHz	333.3 kHz
Timer Prescale (1, 4, 16)	16	4	1	1	1	1
PRx Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Maximum Resolution (bits)	10	10	10	8	7	6.6

TABLE 24-6: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (Fosc = 20 MHz)

PWM Frequency	1.22 kHz	4.88 kHz	19.53 kHz	78.12 kHz	156.3 kHz	208.3 kHz
Timer Prescale (1, 4, 16)	16	4	1	1	1	1
PRx Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Maximum Resolution (bits)	10	10	10	8	7	6.6

TABLE 24-7: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (Fosc = 8 MHz)

PWM Frequency	1.22 kHz	4.90 kHz	19.61 kHz	76.92 kHz	153.85 kHz	200.0 kHz
Timer Prescale (1, 4, 16)	16	4	1	1	1	1
PRx Value	0x65	0x65	0x65	0x19	0x0C	0x09
Maximum Resolution (bits)	8	8	8	6	5	5

FIGURE 25-3: MSSPx BLOCK DIAGRAM (I²C™ SLAVE MODE)

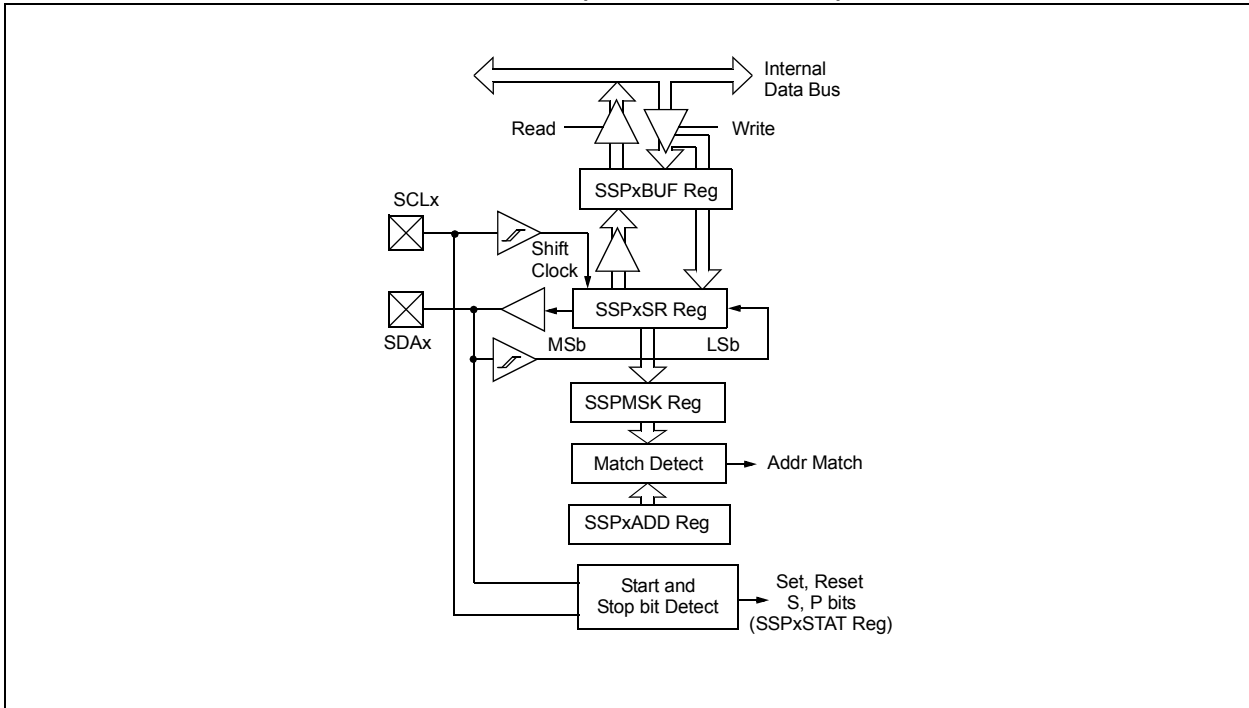


FIGURE 25-20: I²C SLAVE, 10-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 0, DHEN = 0)



26.1 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 V_{OH} mark state which represents a '1' data bit, and a V_{OL} 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 $1/(\text{Baud Rate})$. 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 26-5 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.

26.1.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in Figure 26-1. 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.

26.1.1.1 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 ANSEL bit.

Note 1: The TXIF Transmitter Interrupt flag is set when the TXEN enable bit is set.

26.1.1.2 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 T_{CY} 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.

26.1.1.3 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 26.4.1.2 "Clock Polarity"**.

26.1.1.4 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.

PIC16(L)F1825/9

REGISTER 26-3: BAUDCON: BAUD RATE CONTROL REGISTER

R-0/0	R-1/1	U-0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0
ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN
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 **ABDOVF:** Auto-Baud Detect Overflow bit
Asynchronous mode:
 1 = Auto-baud timer overflowed
 0 = Auto-baud timer did not overflow
Synchronous mode:
 Don't care
- bit 6 **RCIDL:** Receive Idle Flag bit
Asynchronous mode:
 1 = Receiver is Idle
 0 = Start bit has been received and the receiver is receiving
Synchronous mode:
 Don't care
- bit 5 **Unimplemented:** Read as '0'
- bit 4 **SCKP:** Synchronous Clock Polarity Select bit
Asynchronous mode:
 1 = Transmit inverted data to the TX/CK pin
 0 = Transmit non-inverted data to the TX/CK pin
Synchronous mode:
 1 = Data is clocked on rising edge of the clock
 0 = Data is clocked on falling edge of the clock
- bit 3 **BRG16:** 16-bit Baud Rate Generator bit
 1 = 16-bit Baud Rate Generator is used
 0 = 8-bit Baud Rate Generator is used
- bit 2 **Unimplemented:** Read as '0'
- bit 1 **WUE:** Wake-up Enable bit
Asynchronous mode:
 1 = Receiver is waiting for a falling edge. No character will be received, byte RCIF will be set. WUE will automatically clear after RCIF is set.
 0 = Receiver is operating normally
Synchronous mode:
 Don't care
- bit 0 **ABDEN:** Auto-Baud Detect Enable bit
Asynchronous mode:
 1 = Auto-Baud Detect mode is enabled (clears when auto-baud is complete)
 0 = Auto-Baud Detect mode is disabled
Synchronous mode:
 Don't care

BCF	Bit Clear f
Syntax:	[<i>label</i>] BCF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	$0 \rightarrow (f)$
Status Affected:	None
Description:	Bit 'b' in register 'f' is cleared.

BTFSC	Bit Test f, Skip if Clear
Syntax:	[<i>label</i>] BTFSC f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	skip if $(f) = 0$
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.

BRA	Relative Branch
Syntax:	[<i>label</i>] BRA label [<i>label</i>] BRA \$+k
Operands:	$-256 \leq \text{label} - \text{PC} + 1 \leq 255$ $-256 \leq k \leq 255$
Operation:	$(\text{PC}) + 1 + k \rightarrow \text{PC}$
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 $\text{PC} + 1 + k$. This instruction is a 2-cycle instruction. This branch has a limited range.

BTFSS	Bit Test f, Skip if Set
Syntax:	[<i>label</i>] BTFSS f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b < 7$
Operation:	skip if $(f) = 1$
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.

BRW	Relative Branch with W
Syntax:	[<i>label</i>] BRW
Operands:	None
Operation:	$(\text{PC}) + (W) \rightarrow \text{PC}$
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 $\text{PC} + 1 + (W)$. This instruction is a 2-cycle instruction.

BSF	Bit Set f
Syntax:	[<i>label</i>] BSF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	$1 \rightarrow (f)$
Status Affected:	None
Description:	Bit 'b' in register 'f' is set.

PIC16(L)F1825/9

LSLF Logical Left Shift

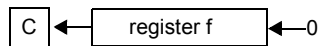
Syntax: [label] LSLF f{,d}

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(f<7>) \rightarrow C$
 $(f<6:0>) \rightarrow \text{dest}<7:1>$
 $0 \rightarrow \text{dest}<0>$

Status Affected: C, Z

Description: The contents of register 'f' are shifted one bit to the left through the Carry flag. A '0' is shifted into the LSB. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.



LSRF Logical Right Shift

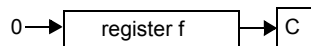
Syntax: [label] LSRF f{,d}

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $0 \rightarrow \text{dest}<7>$
 $(f<7:1>) \rightarrow \text{dest}<6:0>$,
 $(f<0>) \rightarrow C$,

Status Affected: C, Z

Description: The contents of register 'f' are shifted one bit to the right through the Carry flag. A '0' is shifted into the MSb. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.



MOVF Move f

Syntax: [label] MOVF f,d

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(f) \rightarrow (\text{dest})$

Status Affected: Z

Description: The contents of register f is moved to a destination dependent upon the status of d. If $d = 0$, destination is W register. If $d = 1$, the destination is file register f itself. $d = 1$ is useful to test a file register since status flag Z is affected.

Words: 1

Cycles: 1

Example: MOVF FSR, 0

After Instruction

W = value in FSR register

Z = 1

RETFIE **Return from Interrupt**

Syntax: `[label] RETFIE`

Operands: None

Operation: TOS → PC,
 1 → GIE

Status Affected: None

Description: Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a 2-cycle instruction.

Words: 1

Cycles: 2

Example: `RETFIE`

 After Interrupt

 PC = TOS

 GIE = 1

RETURN **Return from Subroutine**

Syntax: `[label] RETURN`

Operands: None

Operation: TOS → PC

Status Affected: None

Description: Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a 2-cycle instruction.

RETLW **Return with literal in W**

Syntax: `[label] RETLW k`

Operands: $0 \leq k \leq 255$

Operation: $k \rightarrow (W)$;
 TOS → PC

Status Affected: None

Description: The W register is loaded with the 8-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a 2-cycle instruction.

Words: 1

Cycles: 2

Example: `CALL TABLE;W contains table`
 `;offset value`
 `• ;W now has table value`
 `•`
 `•`
 `ADDWF PC ;W = offset`
 `RETLW k1 ;Begin table`
 `RETLW k2 ;`
 `•`
 `•`
 `•`
 `RETLW kn ; End of table`

TABLE

Before Instruction

 W = 0x07

After Instruction

 W = value of k8

RLF **Rotate Left f through Carry**

Syntax: `[label] RLF f,d`

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: See description below

Status Affected: C

Description: The contents of register 'f' are rotated one bit to the left through the Carry flag. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example: `RLF REG1,0`

Before Instruction

 REG1 = 1110 0110

 C = 0

After Instruction

 REG1 = 1110 0110

 W = 1100 1100

 C = 1

FIGURE 31-1: I_{DD}, LP OSCILLATOR MODE (F_{osc} = 32 kHz), PIC16LF1825/9 ONLY

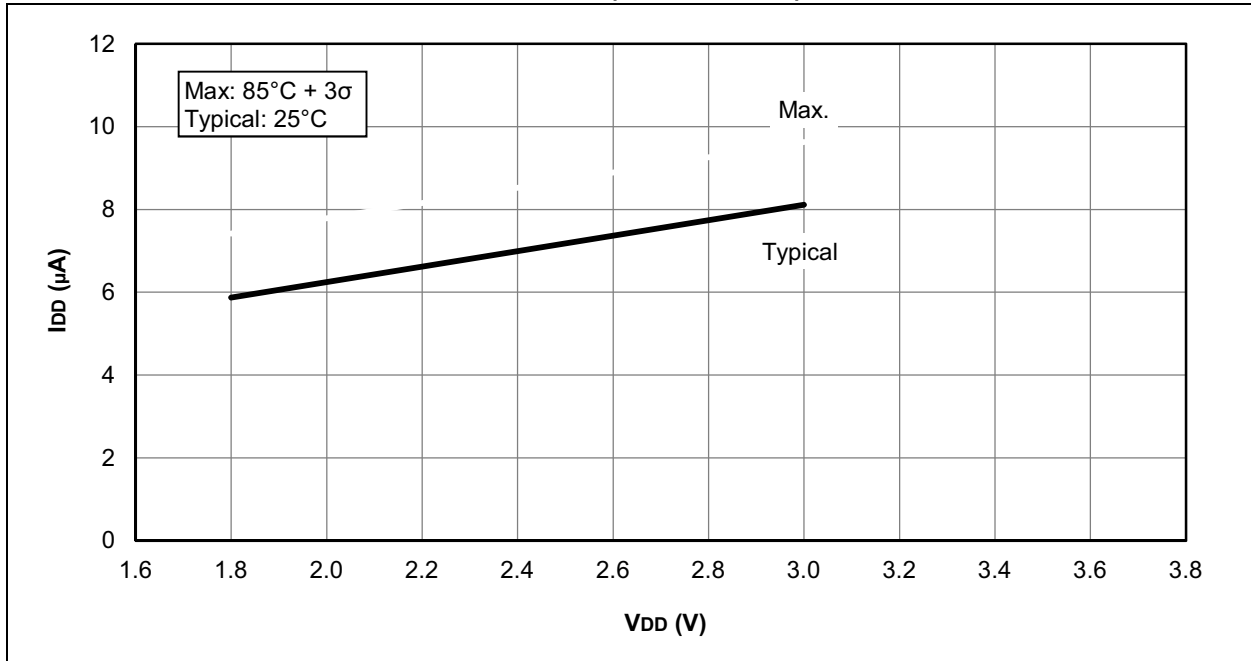


FIGURE 31-2: I_{DD}, LP OSCILLATOR MODE (F_{osc} = 32 kHz), PIC16F1825/9 ONLY

