



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	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	I ² C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, Power Control PWM, QEI, POR, PWM, WDT
Number of I/O	24
Program Memory Size	8KB (4K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 5x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f2331t-e-sog

PIC18F2331/2431/4331/4431

28/40/44-Pin Enhanced Flash Microcontrollers with nanoWatt Technology, High-Performance PWM and A/D

14-Bit Power Control PWM Module:

- Up to 4 Channels with Complementary Outputs
- Edge or Center-Aligned Operation
- Flexible Dead-Band Generator
- Hardware Fault Protection Inputs
- Simultaneous Update of Duty Cycle and Period:
 - Flexible Special Event Trigger output

Motion Feedback Module:

- Three Independent Input Capture Channels:
 - Flexible operating modes for period and pulse-width measurement
 - Special Hall sensor interface module
 - Special Event Trigger output to other modules
- Quadrature Encoder Interface:
 - 2-phase inputs and one index input from encoder
 - High and low position tracking with direction status and change of direction interrupt
 - Velocity measurement

High-Speed, 200 ksps 10-Bit A/D Converter:

- Up to 9 Channels
- Simultaneous, Two-Channel Sampling
- Sequential Sampling: 1, 2 or 4 Selected Channels
- Auto-Conversion Capability
- 4-Word FIFO with Selectable Interrupt Frequency
- Selectable External Conversion Triggers
- Programmable Acquisition Time

Flexible Oscillator Structure:

- Four Crystal modes up to 40 MHz
- Two External Clock modes up to 40 MHz
- Internal Oscillator Block:
 - 8 user-selectable frequencies: 31 kHz to 8 MHz
 - OSCTUNE can compensate for frequency drift
- Secondary Oscillator using Timer1 @ 32 kHz
- Fail-Safe Clock Monitor:
 - Allows for safe shutdown of device if clock fails

Power-Managed Modes:

- Run: CPU on, Peripherals on
- Idle: CPU off, Peripherals on
- Sleep: CPU off, Peripherals off
- Ultra Low, 50 nA Input Leakage
- Idle mode Currents Down to 5.8 μ A, Typical
- Sleep Current Down to 0.1 μ A, Typical
- Timer1 Oscillator, 1.8 μ A, Typical, 32 kHz, 2V
- Watchdog Timer (WDT), 2.1 μ A, typical
- Oscillator Two-Speed Start-up
 - Fast wake from Sleep and Idle, 1 μ s, typical

Peripheral Highlights:

- High-Current Sink/Source 25 mA/25 mA
- Three External Interrupts
- Two Capture/Compare/PWM (CCP) modules
- Enhanced USART module:
 - Supports RS-485, RS-232 and LIN/J2602
 - Auto-wake-up on Start bit
 - Auto-Baud Detect

Special Microcontroller Features:

- 100,000 Erase/Write Cycle Enhanced Flash Program Memory, Typical
- 1,000,000 Erase/Write Cycle Data EEPROM Memory, Typical
- Flash/Data EEPROM Retention: 100 Years
- Self-Programmable under Software Control
- Priority Levels for Interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
 - Programmable period from 41 ms to 131s
- Single-Supply In-Circuit Serial Programming™ (ICSP™) via Two Pins
- In-Circuit Debug (ICD) via Two Pins:
 - Drives PWM outputs safely when debugging

Device	Program Memory		Data Memory		I/O	10-Bit A/D (ch)	CCP	SSP		EUSART	Quadrature Encoder	14-Bit PWM (ch)	Timers 8/16-Bit
	Flash (bytes)	#Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)				SPI	Slave I ² C™				
PIC18F2331	8192	4096	768	256	24	5	2	Y	Y	Y	Y	6	1/3
PIC18F2431	16384	8192	768	256	24	5	2	Y	Y	Y	Y	6	1/3
PIC18F4331	8192	4096	768	256	36	9	2	Y	Y	Y	Y	8	1/3
PIC18F4431	16384	8192	768	256	36	9	2	Y	Y	Y	Y	8	1/3

PIC18F2331/2431/4331/4431

If the IRCF bits and the INTSRC bit are all clear, the INTOSC output is not enabled and the IOFS bit will remain clear; there will be no indication of the current clock source. The INTRC source is providing the device clocks.

If the IRCF bits are changed from all clear (thus, enabling the INTOSC output), or if INTSRC is set, the IOFS bit becomes set after the INTOSC output becomes stable. Clocks to the device continue while the INTOSC source stabilizes, after an interval of TIOBST.

If the IRCF bits were previously at a non-zero value, or if INTSRC was set before setting SCS1 and the INTOSC source was already stable, the IOFS bit will remain set.

On transitions from RC_RUN mode to PRI_RUN mode, the device continues to be clocked from the INTOSC multiplexer while the primary clock is started. When the primary clock becomes ready, a clock switch to the primary clock occurs (see Figure 4-4). When the clock switch is complete, the IOFS bit is cleared, the OST bit is set and the primary clock is providing the device clock. The IDLEN and SCS bits are not affected by the switch. The INTRC source will continue to run if either the WDT or the Fail-Safe Clock Monitor is enabled.

FIGURE 4-3: TRANSITION TIMING TO RC_RUN MODE

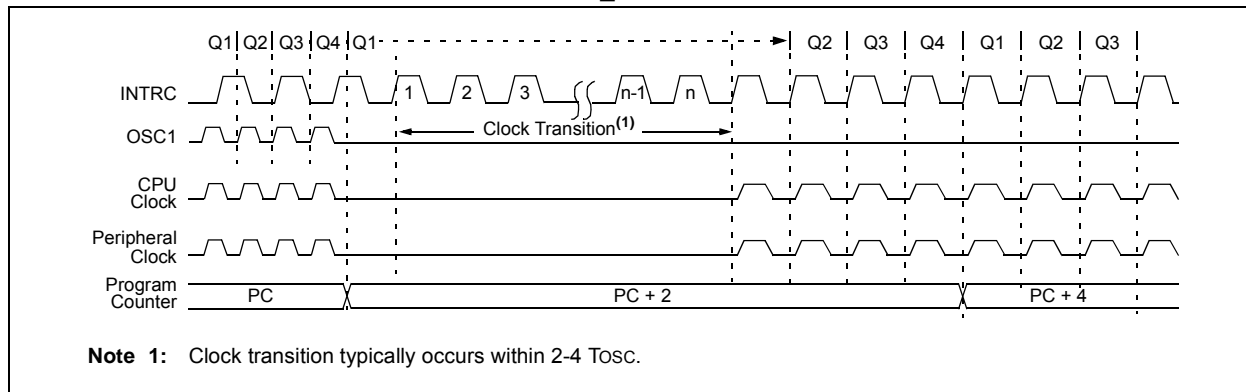
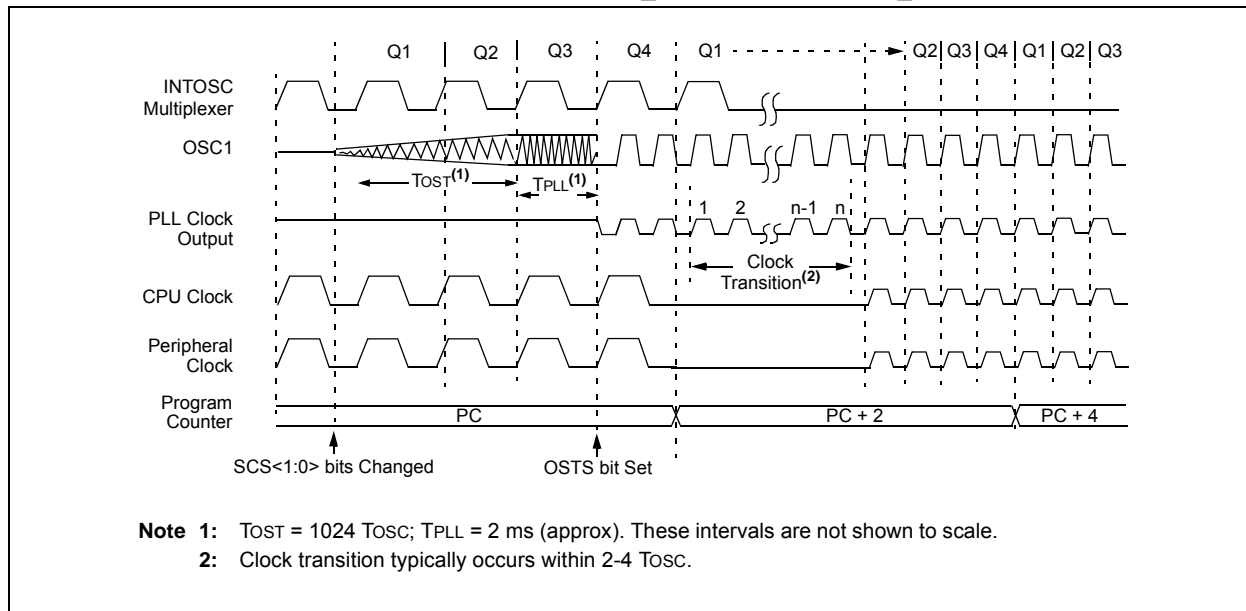


FIGURE 4-4: TRANSITION TIMING FROM RC_RUN MODE TO PRI_RUN MODE



PIC18F2331/2431/4331/4431

FIGURE 5-7: TIME-OUT SEQUENCE ON POR w/PLL ENABLED (MCLR TIED TO VDD)

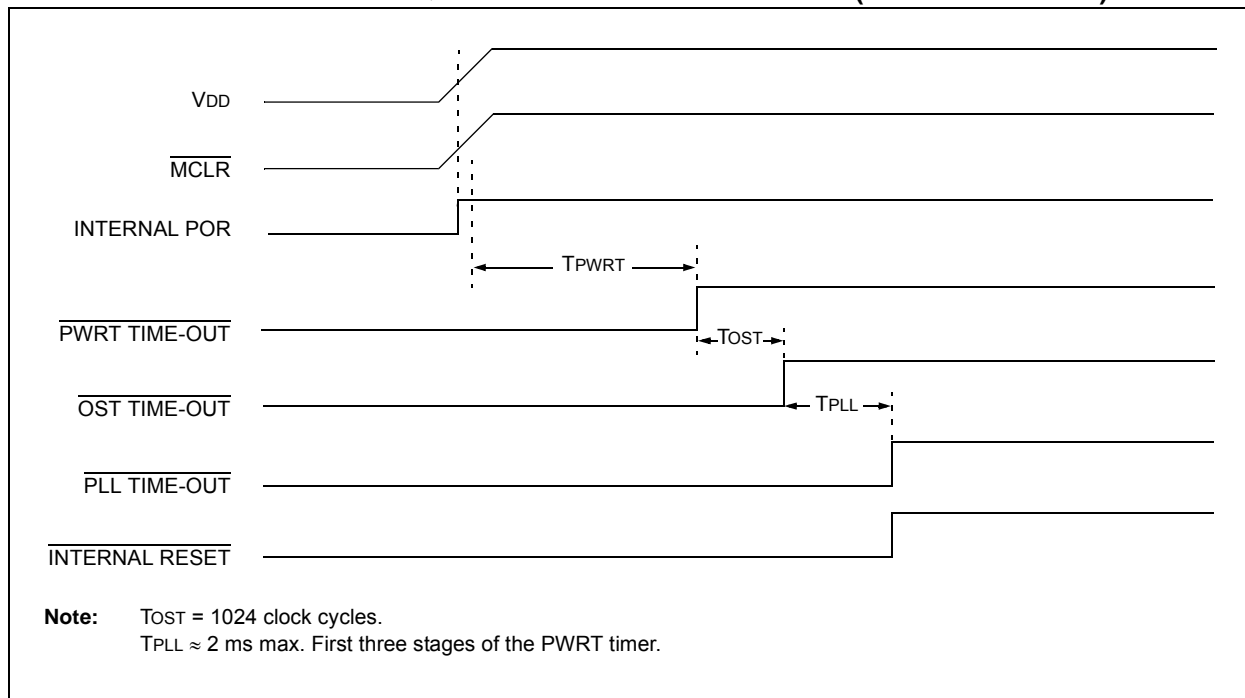


TABLE 5-2: STATUS BITS, THEIR SIGNIFICANCE AND THE INITIALIZATION CONDITION FOR RCON REGISTER

Condition	Program Counter	RCON Register	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	STKFUL	STKUNF
Power-on Reset	0000h	0--1 1100	1	1	1	0	0	0	0
RESET Instruction	0000h	0--0 uuuu	0	u	u	u	u	u	u
Brown-out	0000h	0--1 11u-	1	1	1	u	0	u	u
MCLR Reset during power-managed Run modes	0000h	0--u 1uuu	u	1	u	u	u	u	u
MCLR Reset during power-managed Idle and Sleep modes	0000h	0--u 10uu	u	1	0	u	u	u	u
WDT Time-out during full power or power-managed Run modes	0000h	0--u 0uuu	u	0	u	u	u	u	u
MCLR Reset during full-power execution	0000h	0--u uuuu	u	u	u	u	u	u	u
Stack Full Reset (STVREN = 1)								1	u
Stack Underflow Reset (STVREN = 1)								u	1
Stack Underflow Error (not an actual Reset, STVREN = 0)	0000h	u--u uuuu	u	u	u	u	u	u	1
WDT time-out during power-managed Idle or Sleep modes	PC + 2	u--u 00uu	u	0	0	u	u	u	u
Interrupt exit from power-managed modes	PC + 2 ⁽¹⁾	u--u u0uu	u	u	0	u	u	u	u

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

Note 1: When the wake-up is due to an interrupt and the GIEH or GIEL bits are set, the PC is loaded with the interrupt vector (0x000008h or 0x000018h).

PIC18F2331/2431/4331/4431

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

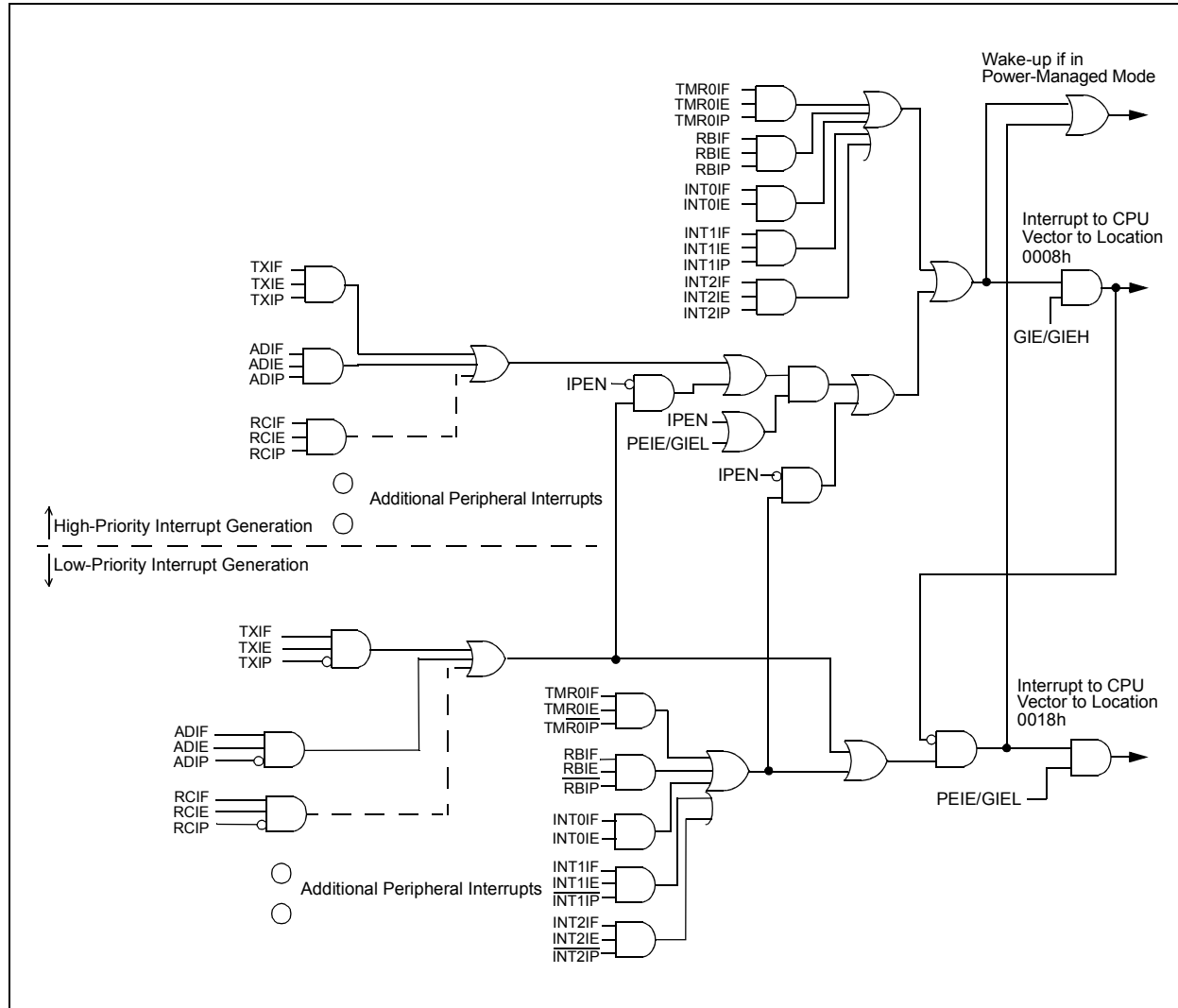
Register	Applicable Devices				Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
PTCON0	2331	2431	4331	4431	0000 0000	uuuu uuuu	uuuu uuuu
PTCON1	2331	2431	4331	4431	00-- ----	00-- ----	uu-- ----
PTMRL	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PTMRH	2331	2431	4331	4431	---- 0000	---- 0000	---- uuuu
PTPERL	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
PTPERH	2331	2431	4331	4431	---- 1111	---- 1111	---- uuuu
PDC0L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC0H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
PDC1L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC1H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
PDC2L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC2H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
PDC3L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC3H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
SEVTCMPL	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
SEVTCMPH	2331	2431	4331	4431	---- 0000	---- 0000	---- uuuu
PWMCON0	2331	2431	4331	4431	-111 0000	-111 0000	-uuu uuuu
PWMCON1	2331	2431	4331	4431	0000 0-00	0000 0-00	uuuu u-uu
DTCON	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
FLTCONFIG	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
OVDCOND	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
OVDCONS	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
CAP1BUFH/ VELRH	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP1BUFL/ VELRL	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP2BUFH/ POSCNTH	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP2BUFL/ POSCNTL	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP3BUFH/ MAXCNTH	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.
Shaded cells indicate conditions do not apply for the designated device.

- Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).
- 2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).
- 3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.
- 4:** See Table 5-2 for Reset value for specific condition.
- 5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.
- 6:** Bit 3 of PORTE and LATE are enabled if MCLR functionality is disabled. When not enabled as the PORTE pin, they are disabled and read as '0'. The 28-pin devices do not have only RE3 implemented.

PIC18F2331/2431/4331/4431

FIGURE 10-1: INTERRUPT LOGIC



PIC18F2331/2431/4331/4431

REGISTER 10-8: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

R/W-0	U-0	U-0	R/W-0	U-0	R/W-0	U-0	R/W-0
OSCFIE	—	—	EEIE	—	LVDIE	—	CCP2IE
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 **OSCFIE:** Oscillator Fail Interrupt Enable bit

1 = Enabled

0 = Disabled

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **EEIE:** Interrupt Enable bit

1 = Enabled

0 = Disabled

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

bit 2 **LVDIE:** Low-Voltage Detect Interrupt Enable bit

1 = Enabled

0 = Disabled

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

bit 0 **CCP2IE:** CCP2 Interrupt Enable bit

1 = Enabled

0 = Disabled

PIC18F2331/2431/4331/4431

TABLE 11-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	57
LATB	LATB Data Output Register								57
TRISB	PORTB Data Direction Register								57
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
INTCON2	$\overline{\text{RBP}}\text{U}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	54
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	54

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

PIC18F2331/2431/4331/4431

Table 18-1 shows the minimum PWM frequencies that can be generated with the PWM time base and the prescaler. An operating frequency of 40 MHz (F_{CYC} = 10 MHz) and PTPER = 0xFFF is assumed in the table. The PWM module must be capable of generating PWM signals at the line frequency (50 Hz or 60 Hz) for certain power control applications.

TABLE 18-1: MINIMUM PWM FREQUENCY

Minimum PWM Frequencies vs. Prescaler Value for F _{CYC} = 10 MIPS (PTPER = 0xFFFh)		
Prescale	PWM Frequency Edge-Aligned	PWM Frequency Center-Aligned
1:1	2441 Hz	1221 Hz
1:4	610 Hz	305 Hz
1:16	153 Hz	76 Hz
1:64	38 Hz	19 Hz

18.3.5 PWM TIME BASE POSTSCALER

The match output of PTMR can optionally be postscaled through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate an interrupt. The postscaler counter is cleared when any of the following occurs:

- Write to the PTMR register
- Write to the PTCN register
- Any device Reset

The PTMR register is not cleared when PTCN is written.

18.4 PWM Time Base Interrupts

The PWM timer can generate interrupts based on the modes of operation selected by the PTMOD<1:0> bits and the postscaler bits (PTOPS<3:0>).

18.4.1 INTERRUPTS IN FREE-RUNNING MODE

When the PWM time base is in the Free-Running mode (PTMOD<1:0> = 00), an interrupt event is generated each time a match with the PTPER register occurs. The PTMR register is reset to zero in the following clock edge.

Using a postscaler selection other than 1:1 will reduce the frequency of interrupt events.

FIGURE 18-5: PWM TIME BASE INTERRUPT TIMING, FREE-RUNNING MODE

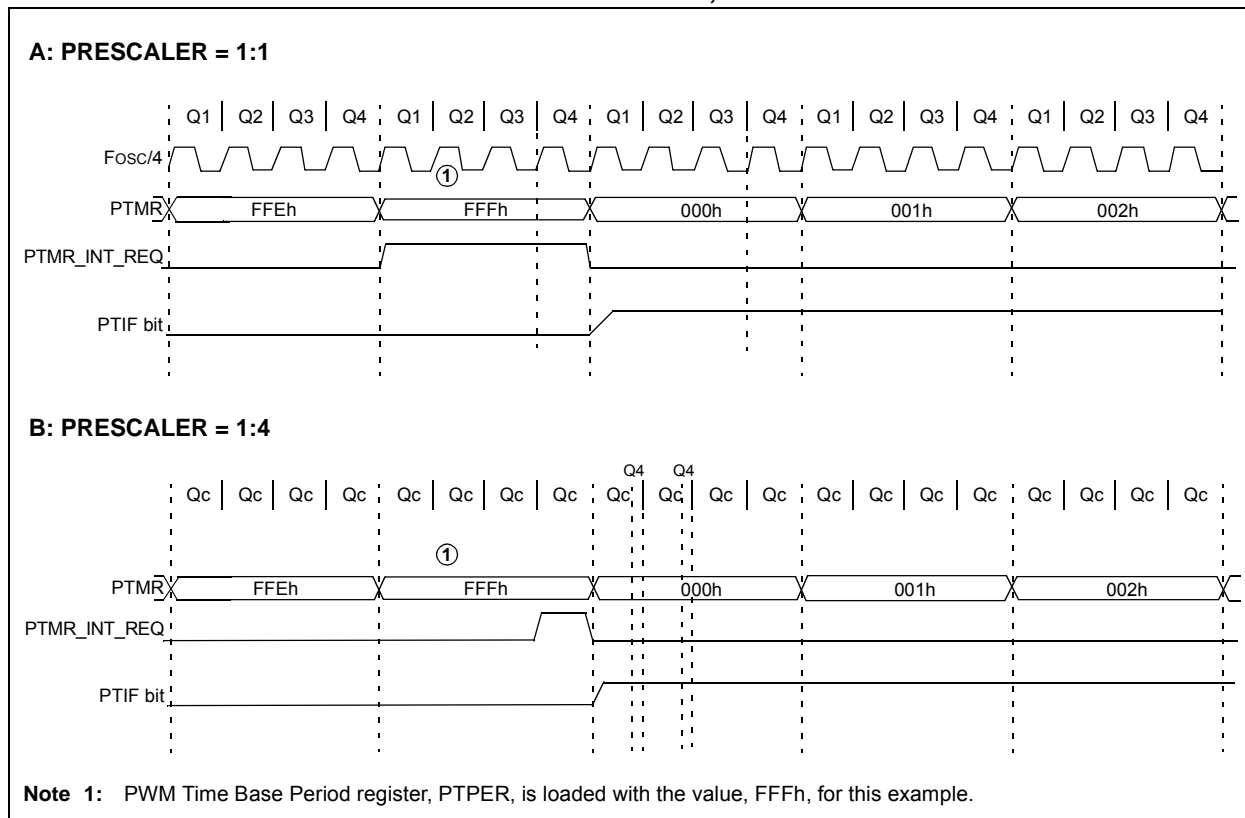
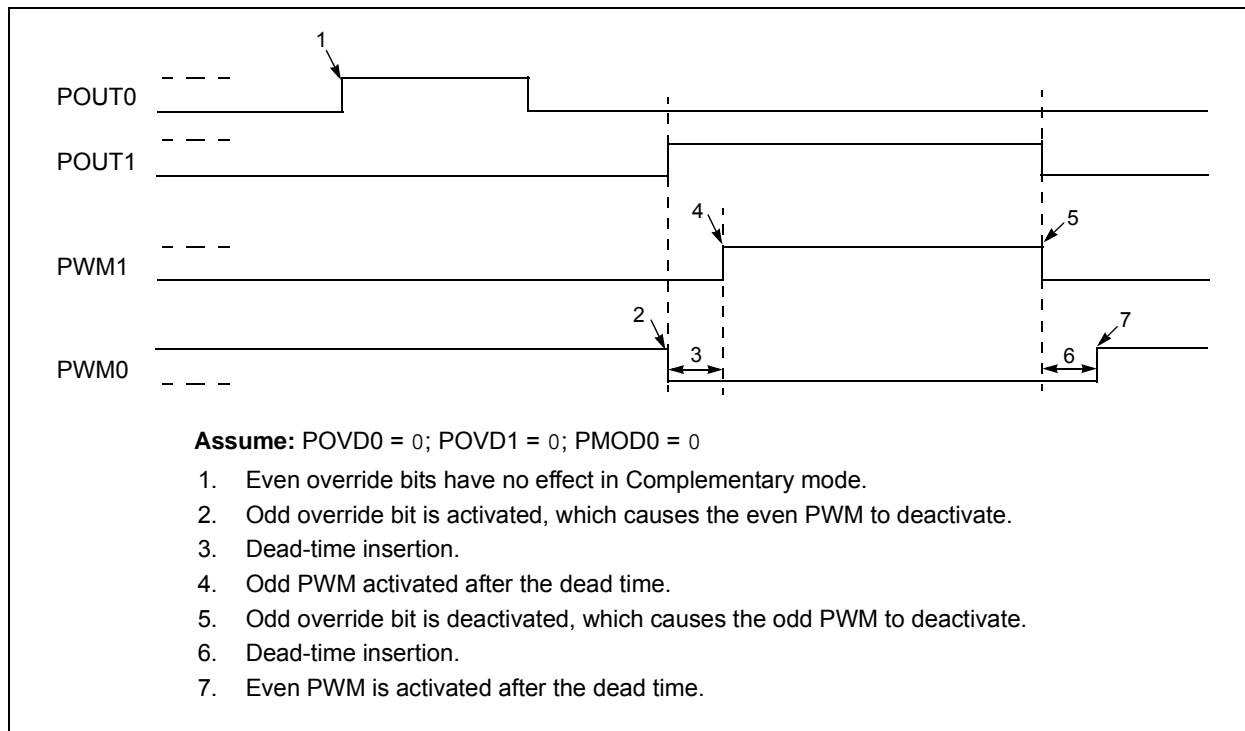


FIGURE 18-20: PWM OVERRIDE BITS IN COMPLEMENTARY MODE



PIC18F2331/2431/4331/4431

20.3.5 BREAK CHARACTER SEQUENCE

The Enhanced USART module has the capability of sending the special Break character sequences that are required by the LIN/J2602 bus standard. The Break character transmit consists of a Start bit, followed by twelve '0' bits and a Stop bit. The Frame Break character is sent whenever the SENDB and TXEN bits (TXSTA<3> and TXSTA<5>) are set while the Transmit Shift register is loaded with data. Note that the value of data written to TXREG will be ignored and all '0's will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN/J2602 specification).

Note that the data value written to the TXREG for the Break character is ignored. The write simply serves the purpose of initiating the proper sequence.

The TRMT bit indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 20-9 for the timing of the Break character sequence.

20.3.5.1 Break and Sync Transmit Sequence

The following sequence will send a message frame header made up of a Break, followed by an Auto-Baud Sync byte. This sequence is typical of a LIN/J2602 bus master.

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

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

20.3.6 RECEIVING A BREAK CHARACTER

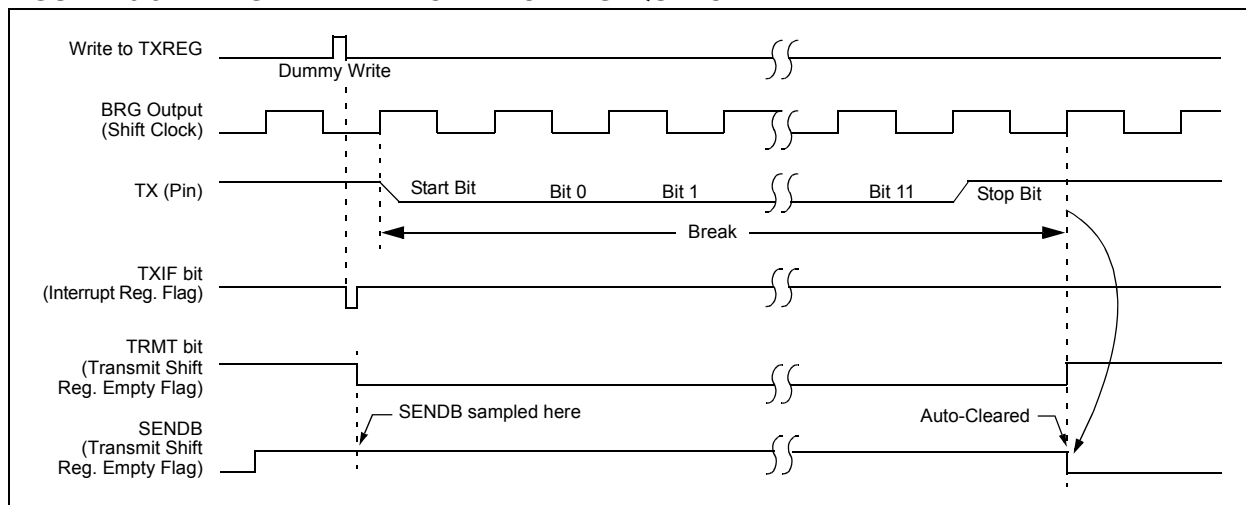
The Enhanced USART module can receive a Break character in two ways.

The first method forces configuration of the baud rate at a frequency of 9/13 of the typical speed. This allows for the Stop bit transition to be at the correct sampling location (13 bits for Break versus Start bit and 8 data bits for typical data).

The second method uses the auto-wake-up feature described in **Section 20.3.4 "Auto-Wake-up on Sync Break Character"**. By enabling this feature, the EUSART will sample the next two transitions on RX/DT, cause an RCIF interrupt and receive the next data byte followed by another interrupt.

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

FIGURE 20-9: SEND BREAK CHARACTER SEQUENCE



PIC18F2331/2431/4331/4431

REGISTER 21-2: ADCON1: A/D CONTROL REGISTER 1

R/W-0	R/W-0	U-0	R/W-0	R-0	R-0	R-0	R-0
VCFG1	VCFG0	—	FIFOEN	BFEMT	BFOVL	ADPNT1	ADPNT0
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 **VCFG<1:0>**: A/D VREF+ and A/D VREF- Source Selection bits
00 = VREF+ = AVDD, VREF- = AVSS (AN2 and AN3 are analog inputs or digital I/O)
01 = VREF+ = External VREF+, VREF- = AVSS (AN2 is an analog input or digital I/O)
10 = VREF+ = AVDD, VREF- = External VREF- (AN3 is an analog input or digital I/O)
11 = VREF+ = External VREF+, VREF- = External VREF-
- bit 5 **Unimplemented**: Read as '0'
- bit 4 **FIFOEN**: FIFO Buffer Enable bit
1 = FIFO is enabled
0 = FIFO is disabled
- bit 3 **BFEMT**: Buffer Empty bit
1 = FIFO is empty
0 = FIFO is not empty (at least one of four locations has unread A/D result data)
- bit 2 **BFOVFL**: Buffer Overflow bit
1 = A/D result has overwritten a buffer location that has unread data
0 = A/D result has not overflowed
- bit 1-0 **ADPNT<1:0>**: Buffer Read Pointer Location bits
Designates the location to be read next.
00 = Buffer Address 0
01 = Buffer Address 1
10 = Buffer Address 2
11 = Buffer Address 3

PIC18F2331/2431/4331/4431

REGISTER 21-3: ADCON2: A/D CONTROL REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ACQT3	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
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 **ADFM:** A/D Result Format Select bit

1 = Right justified

0 = Left justified

bit 6-3 **ACQT<3:0>:** A/D Acquisition Time Select bits

0000 = No delay (conversion starts immediately when $\overline{GO/DONE}$ is set)⁽¹⁾

0001 = 2 TAD

0010 = 4 TAD

0011 = 6 TAD

0100 = 8 TAD

0101 = 10 TAD

0110 = 12 TAD

0111 = 16 TAD

1000 = 20 TAD

1001 = 24 TAD

1010 = 28 TAD

1011 = 32 TAD

1100 = 36 TAD

1101 = 40 TAD

1110 = 48 TAD

1111 = 64 TAD

bit 2-0 **ADCS<2:0>:** A/D Conversion Clock Select bits

000 = FOSC/2

001 = FOSC/8

010 = FOSC/32

011 = FRC/4

100 = FOSC/4

101 = FOSC/16

110 = FOSC/64

111 = FRC (Internal A/D RC Oscillator)

Note 1: If the A/D RC clock source is selected, a delay of one T_{CY} (instruction cycle) is added before the A/D clock starts. This allows the *SLEEP* instruction to be executed before starting a conversion.

PIC18F2331/2431/4331/4431

23.5.2 DATA EEPROM CODE PROTECTION

The entire data EEPROM is protected from external reads and writes by two bits: CPD and WRTD. CPD inhibits external reads and writes of data EEPROM. WRTD inhibits external writes to data EEPROM. The CPU can continue to read and write data EEPROM regardless of the protection bit settings.

23.5.3 CONFIGURATION REGISTER PROTECTION

The Configuration registers can be write-protected. The WRTC bit controls protection of the Configuration registers. In normal execution mode, the WRTC bit is readable only. WRTC can only be written via ICSP or an external programmer.

23.6 ID Locations

Eight memory locations (200000h-200007h) are designated as ID locations, where the user can store checksum or other code identification numbers. These locations are both readable and writable during normal execution through the TBLRD and TBLWT instructions, or during program/verify. The ID locations can be read when the device is code-protected.

23.7 In-Circuit Serial Programming

PIC18F2331/2431/4331/4431 microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data, and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices, and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

23.8 In-Circuit Debugger

When the $\overline{\text{DEBUG}}$ bit in the CONFIG4L Configuration register is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® IDE. When the microcontroller has this feature enabled, some resources are not available for general use. Table 23-4 shows which resources are required by the background debugger.

TABLE 23-4: DEBUGGER RESOURCES

I/O pins:	RB6, RB7
Stack:	2 levels
Program Memory:	<1 Kbytes
Data Memory:	16 bytes

To use the In-Circuit Debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to $\overline{\text{MCLR/VPP}}$, VDD, VSS, RB7 and RB6. This will interface to the In-Circuit Debugger module available from Microchip or one of the third party development tool companies.

23.9 Single-Supply ICSP™ Programming

The LVP bit in Configuration Register 4L (CONFIG4L<2>) enables Single-Supply ICSP Programming. When LVP is enabled, the microcontroller can be programmed without requiring high voltage being applied to the $\overline{\text{MCLR/VPP}}$ pin, but the RB5/PGM pin is then dedicated to controlling Program mode entry and is not available as a general purpose I/O pin.

LVP is enabled in erased devices.

While programming, using Single-Supply Programming, VDD is applied to the $\overline{\text{MCLR/VPP}}$ pin as in normal execution mode. To enter Programming mode, VDD is applied to the PGM pin.

Note 1: High-voltage programming is always available, regardless of the state of the LVP bit or the PGM pin, by applying V_{IH} to the $\overline{\text{MCLR}}$ pin.

2: When Single-Supply Programming is enabled, the RB5 pin can no longer be used as a general purpose I/O pin.

3: When LVP is enabled externally, pull the PGM pin to VSS to allow normal program execution.

If Single-Supply ICSP Programming mode will not be used, the LVP bit can be cleared and RB5/PGM becomes available as the digital I/O pin RB5. The LVP bit may be set or cleared only when using standard high-voltage programming (V_{IH} applied to the $\overline{\text{MCLR/VPP}}$ pin). Once LVP has been disabled, only the standard high-voltage programming is available and must be used to program the device.

Memory that is not code-protected can be erased using either a block erase, or erased row by row, then written at any specified VDD. If code-protected memory is to be erased, a block erase is required. If a block erase is to be performed when using Single-Supply Programming, the device must be supplied with VDD of 4.5V to 5.5V.

PIC18F2331/2431/4331/4431

ADDWFC

ADD W and Carry bit to f

Syntax: [*label*] ADDWFC f [,d [,a]]

Operands: $0 \leq f \leq 255$

$d \in [0,1]$

$a \in [0,1]$

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

Status Affected: N, OV, C, DC, Z

Encoding:

0010	00da	ffff	ffff
------	------	------	------

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'. If 'a' is '0', the Access Bank will be selected. If 'a' is '1', the BSR will not be overridden.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example:

ADDWFC REG, W

Before Instruction

Carry bit = 1
REG = 0x02
W = 0x4D

After Instruction

Carry bit = 0
REG = 0x02
W = 0x50

ANDLW

AND Literal with W

Syntax: [*label*] ANDLW k

Operands: $0 \leq k \leq 255$

Operation: $(W) .\text{AND}. k \rightarrow W$

Status Affected: N, Z

Encoding:

0000	1011	kkkk	kkkk
------	------	------	------

Description: The contents of W are ANDed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example:

ANDLW 0x5F

Before Instruction

W = 0xA3

After Instruction

W = 0x03

PIC18F2331/2431/4331/4431

BNC

Branch if Not Carry

Syntax: [*label*] BNC n

Operands: $-128 \leq n \leq 127$

Operation: if Carry bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0011	nnnn	nnnn
------	------	------	------

Description: If the Carry bit is '0', then the program will branch.
The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: HERE BNC Jump

Before Instruction

PC = address (HERE)

After Instruction

If Carry = 0;

PC = address (Jump)

If Carry = 1;

PC = address (HERE + 2)

BNN

Branch if Not Negative

Syntax: [*label*] BNN n

Operands: $-128 \leq n \leq 127$

Operation: if Negative bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0111	nnnn	nnnn
------	------	------	------

Description: If the Negative bit is '0', then the program will branch.
The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: HERE BNN Jump

Before Instruction

PC = address (HERE)

After Instruction

If Negative = 0;

PC = address (Jump)

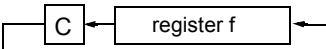
If Negative = 1;

PC = address (HERE + 2)

PIC18F2331/2431/4331/4431

RETURN		Return from Subroutine															
Syntax:	[<i>label</i>] RETURN [<i>s</i>]																
Operands:	$s \in [0,1]$																
Operation:	(TOS) → PC; if $s = 1$: (WS) → W, (STATUS) → STATUS, (BSRS) → BSR, PCLATU, PCLATH are unchanged																
Status Affected:	None																
Encoding:	<table border="1"><tr><td>0000</td><td>0000</td><td>0001</td><td>001<i>s</i></td></tr></table>					0000	0000	0001	001 <i>s</i>								
0000	0000	0001	001 <i>s</i>														
Description:	Return from subroutine. The stack is popped and the top of the stack (TOS) is loaded into the program counter. If 's' = 1, the contents of the shadow registers, WS, STATUS and BSRS, are loaded into their corresponding registers, W, STATUS and BSR. If 's' = 0, no update of these registers occurs.																
Words:	1																
Cycles:	2																
Q Cycle Activity:	<table><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>Decode</td><td>No operation</td><td>Process Data</td><td>POP PC from stack</td></tr><tr><td>No operation</td><td>No operation</td><td>No operation</td><td>No operation</td></tr></table>					Q1	Q2	Q3	Q4	Decode	No operation	Process Data	POP PC from stack	No operation	No operation	No operation	No operation
Q1	Q2	Q3	Q4														
Decode	No operation	Process Data	POP PC from stack														
No operation	No operation	No operation	No operation														


Example: RETURN
After Interrupt
PC = TOS

RLCF		Rotate Left f through Carry							
Syntax:	[<i>label</i>] RLCF f [,d [,a]]								
Operands:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$								
Operation:	$(f < n) \rightarrow \text{dest} < n + 1 > ,$ $(f < 7) \rightarrow C ,$ $(C) \rightarrow \text{dest} < 0 >$								
Status Affected:	C, N, Z								
Encoding:	<table><tr><td>0011</td><td>01<i>da</i></td><td><i>ffff</i></td><td><i>ffff</i></td></tr></table>					0011	01 <i>da</i>	<i>ffff</i>	<i>ffff</i>
0011	01 <i>da</i>	<i>ffff</i>	<i>ffff</i>						
Description:	<p>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 W. If 'd' is '1', the result is stored back in register, 'f'. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.</p> <div></div>								
Words:	1								
Cycles:	1								
Q Cycle Activity:									
Q1		Q2		Q3		Q4			
Decode		Read register 'f'		Process Data		Write to destination			

Example: RLCF REG, W
Before Instruction
REG = 1110 0110
C = 0
After Instruction
REG = 1110 0110
W = 1100 1100
C = 1

PIC18F2331/2431/4331/4431

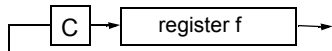
RLNCF Rotate Left f (No Carry)

Syntax:	[<i>label</i>] RLNCF f [,d [,a]]								
Operands:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$								
Operation:	$(f<n>) \rightarrow \text{dest}<n+1>$, $(f<7>) \rightarrow \text{dest}<0>$								
Status Affected:	N, Z								
Encoding:	<table><tr><td>0100</td><td>01da</td><td>ffff</td><td>ffff</td></tr></table>	0100	01da	ffff	ffff				
0100	01da	ffff	ffff						
Description:	<p>The contents of register, 'f', are rotated one bit to the left. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register, 'f'. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value.</p> 								
Words:	1								
Cycles:	1								
Q Cycle Activity:	<table><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>Decode</td><td>Read register 'f'</td><td>Process Data</td><td>Write to destination</td></tr></table>	Q1	Q2	Q3	Q4	Decode	Read register 'f'	Process Data	Write to destination
Q1	Q2	Q3	Q4						
Decode	Read register 'f'	Process Data	Write to destination						

Example: RLNCF REG

Before Instruction
 REG = 1010 1011
 After Instruction
 REG = 0101 0111

RRCF Rotate Right f through Carry

Syntax:	[<i>label</i>] RRCF f [,d [,a]]			
Operands:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$			
Operation:	$(f<n>) \rightarrow \text{dest}<n - 1>$, $(f<0>) \rightarrow C$, $(C) \rightarrow \text{dest}<7>$			
Status Affected:	C, N, Z			
Encoding:	0011	00da	ffff	ffff
Description:	<p>The contents of register, 'f', are rotated one bit to the right through the Carry Flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register, 'f'. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value.</p> 			
Words:	1			
Cycles:	1			
Q Cycle Activity:				
	Q1	Q2	Q3	Q4
	Decode	Read register 'f'	Process Data	Write to destination

Example: RRCF REG, W

Before Instruction
 REG = 1110 0110
 C = 0
 After Instruction
 REG = 1110 0110
 W = 0111 0011
 C = 0

PIC18F2331/2431/4331/4431

SUBLW Subtract W from Literal

Syntax: [label] SUBLW k

Operands: $0 \leq k \leq 255$

Operation: $k - (W) \rightarrow W$

Status Affected: N, OV, C, DC, Z

Encoding:

0000	1000	kkkk	kkkk
------	------	------	------

Description: W is subtracted from the 8-bit literal, 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example 1: SUBLW 0x02

Before Instruction

W = 1
C = ?

After Instruction

W = 1
C = 1 ; result is positive
Z = 0
N = 0

Example 2: SUBLW 0x02

Before Instruction

W = 2
C = ?

After Instruction

W = 0
C = 1 ; result is zero
Z = 1
N = 0

Example 3: SUBLW 0x02

Before Instruction

W = 3
C = ?

After Instruction

W = FF ; (2's complement)
C = 0 ; result is negative
Z = 0
N = 1

SUBWF Subtract W from f

Syntax: [label] SUBWF f[d,[a]]

Operands: $0 \leq f \leq 255$
 $d \in [0,1]$
 $a \in [0,1]$

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

Status Affected: N, OV, C, DC, Z

Encoding:

0101	11da	ffff	ffff
------	------	------	------

Description: Subtract W from register, 'f' (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register, 'f'. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: SUBWF REG

Before Instruction

REG = 3
W = 2
C = ?

After Instruction

REG = 1
W = 2
C = 1 ; result is positive
Z = 0
N = 0

Example 2: SUBWF REG, W

Before Instruction

REG = 2
W = 2
C = ?

After Instruction

REG = 2
W = 0
C = 1 ; result is zero
Z = 1
N = 0

Example 3: SUBWF REG

Before Instruction

REG = 0x01
W = 0x02
C = ?

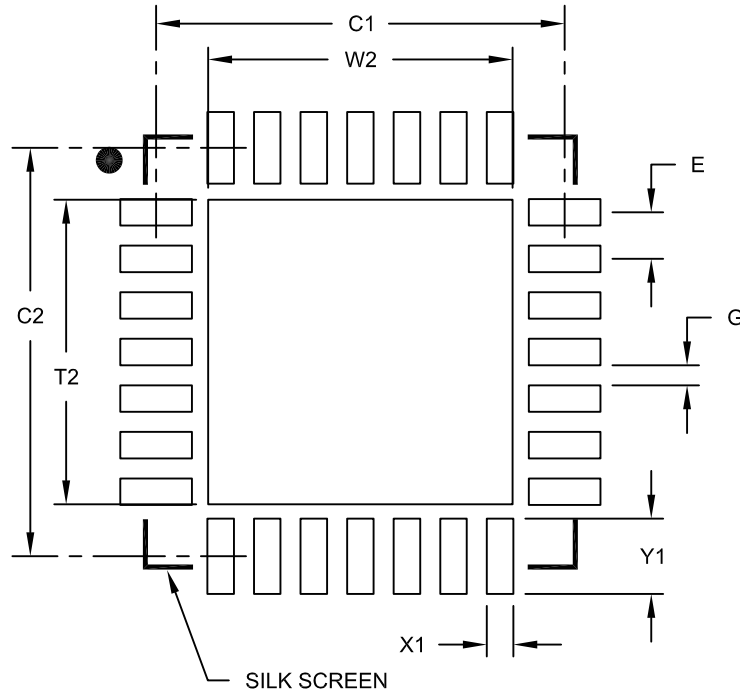
After Instruction

REG = 0xFFh ; (2's complement)
W = 0x02
C = 0x00 ; result is negative
Z = 0x00
N = 0x01

PIC18F2331/2431/4331/4431

28-Lead Plastic Quad Flat, No Lead Package (ML) – 6x6 mm Body [QFN] with 0.55 mm Contact Length

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



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	0.65 BSC		
Optional Center Pad Width	W2			4.25
Optional Center Pad Length	T2			4.25
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.37
Contact Pad Length (X28)	Y1			1.00
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2105A

PIC18F2331/2431/4331/4431

READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this document.

TO: Technical Publications Manager Total Pages Sent _____

RE: Reader Response

From: Name _____

Company _____

Address _____

City / State / ZIP / Country _____

Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional):

Would you like a reply? ___Y___N

Device: PIC18F2331/2431/4331/4431

Literature Number: DS39616D

Questions:

1. What are the best features of this document?

2. How does this document meet your hardware and software development needs?

3. Do you find the organization of this document easy to follow? If not, why?

4. What additions to the document do you think would enhance the structure and subject?

5. What deletions from the document could be made without affecting the overall usefulness?

6. Is there any incorrect or misleading information (what and where)?

7. How would you improve this document?
