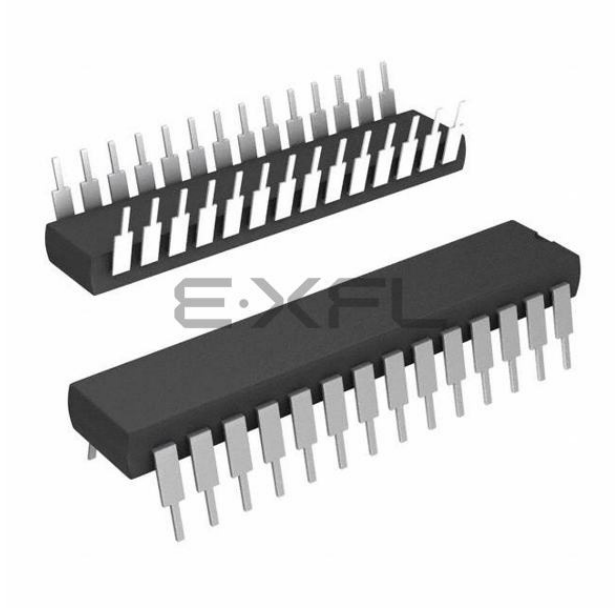


Welcome to E-XFL.COM

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	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	16KB (8K x 16)
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 24x10b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Through Hole
Package / Case	28-DIP (0.300", 7.62mm)
Supplier Device Package	28-SPDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f24k40-e-sp

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com**. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Website at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000000A is version A of document DS30000000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Website; **http://www.microchip.com**
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our website at **www.microchip.com** to receive the most current information on all of our products.

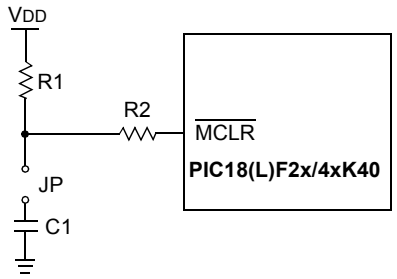
2.3 Master Clear (MCLR) Pin

The MCLR pin provides two specific device functions: Device Reset, and Device Programming and Debugging. If programming and debugging are not required in the end application, a direct connection to VDD may be all that is required. The addition of other components, to help increase the application's resistance to spurious Resets from voltage sags, may be beneficial. A typical configuration is shown in Figure 2-1. Other circuit designs may be implemented, depending on the application's requirements.

During programming and debugging, the resistance and capacitance that can be added to the pin must be considered. Device programmers and debuggers drive the MCLR pin. Consequently, specific voltage levels (VIH and VIL) and fast signal transitions must not be adversely affected. Therefore, specific values of R1 and C1 will need to be adjusted based on the application and PCB requirements. For example, it is recommended that the capacitor, C1, be isolated from the MCLR pin during programming and debugging operations by using a jumper (Figure 2-2). The jumper is replaced for normal run-time operations.

Any components associated with the MCLR pin should be placed within 0.25 inch (6 mm) of the pin.

FIGURE 2-2: EXAMPLE OF MCLR PIN CONNECTIONS



Note 1: R1 ≤ 10 kΩ is recommended. A suggested starting value is 10 kΩ. Ensure that the MCLR pin VIH and VIL specifications are met.

2: R2 ≤ 470Ω will limit any current flowing into MCLR from the external capacitor, C1, in the event of MCLR pin breakdown, due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS). Ensure that the MCLR pin VIH and VIL specifications are met.

2.4 ICSP™ Pins

The PGC and PGD pins are used for In-Circuit Serial Programming™ (ICSP™) and debugging purposes. It is recommended to keep the trace length between the ICSP connector and the ICSP pins on the device as short as possible. If the ICSP connector is expected to experience an ESD event, a series resistor is recommended, with the value in the range of a few tens of ohms, not to exceed 100Ω.

Pull-up resistors, series diodes and capacitors on the PGC and PGD pins are not recommended as they will interfere with the programmer/debugger communications to the device. If such discrete components are an application requirement, they should be removed from the circuit during programming and debugging. Alternatively, refer to the AC/DC characteristics and timing requirements information in the respective device Flash programming specification for information on capacitive loading limits, and pin input voltage high (VIH) and input low (VIL) requirements.

For device emulation, ensure that the "Communication Channel Select" (i.e., PGCx/PGDx pins), programmed into the device, matches the physical connections for the ICSP to the Microchip debugger/emulator tool.

For more information on available Microchip development tools connection requirements, refer to **Section 36.0 "Development Support"**.

REGISTER 3-6: CONFIGURATION WORD 3H (30 0005h): WINDOWED WATCHDOG TIMER

U-1	U-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	WDTCCS<2:0>			WDTCWS<2:0>		
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '1'	
-n = Value for blank device	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as '1'

bit 5-3 **WDTCCS<2:0>:** WDT Input Clock Selector bits

If WDTE<1:0> fuses = 2'b00

This bit is ignored.

Otherwise:

111 = Software Control
110 = Reserved (Default to LFINTOSC)
.
.
.
010 = Reserved (Default to LFINTOSC)
001 = WDT reference clock is the 31.25 kHz MFINTOSC
000 = WDT reference clock is the 31.0 kHz LFINTOSC (default value)

bit 2-0 **WDTCWS<2:0>:** WDT Window Select bits

WDTCWS	WINDOW at POR			Software control of WINDOW	Keyed access required?
	Value	Window delay Percent of time	Window opening Percent of time		
111	111	n/a	100	Yes	No
110	111	n/a	100	No	Yes
101	101	25	75		
100	100	37.5	62.5		
011	011	50	50		
010	010	62.5	37.5		
001	001	75	25		
000	000	87.5	12.5		

REGISTER 4-3: OSCCON3: OSCILLATOR CONTROL REGISTER 3

R/W/HC-0/0	R/W-0/0	U-0	R-0/0	R-0/0	U-0	U-0	U-0
CSWHOLD	SOSCPWR	—	ORDY	NOSCR	—	—	—
bit 7							bit 0

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

bit 7 **CSWHOLD:** Clock Switch Hold bit
- 1 = Clock switch will hold (with interrupt) when the oscillator selected by NOSC is ready
- 0 = Clock switch may proceed when the oscillator selected by NOSC is ready; NOSCR becomes '1', the switch will occur

bit 6 **SOSCPWR:** Secondary Oscillator Power Mode Select bit
- 1 = Secondary oscillator operating in High-Power mode
- 0 = Secondary oscillator operating in Low-Power mode

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

bit 4 **ORDY:** Oscillator Ready bit (read-only)
- 1 = OSCCON1 = OSCCON2; the current system clock is the clock specified by NOSC
- 0 = A clock switch is in progress

bit 3 **NOSCR:** New Oscillator is Ready bit (read-only)**(1)**
- 1 = A clock switch is in progress and the oscillator selected by NOSC indicates a "ready" condition
- 0 = A clock switch is not in progress, or the NOSC-selected oscillator is not yet ready

bit 2-0 **Unimplemented:** Read as '0'

Note 1: If CSWHOLD = 0, the user may not see this bit set because, when the oscillator becomes ready there may be a delay of one instruction clock before this bit is set. The clock switch occurs in the next instruction cycle and this bit is cleared.

8.0 RESETS

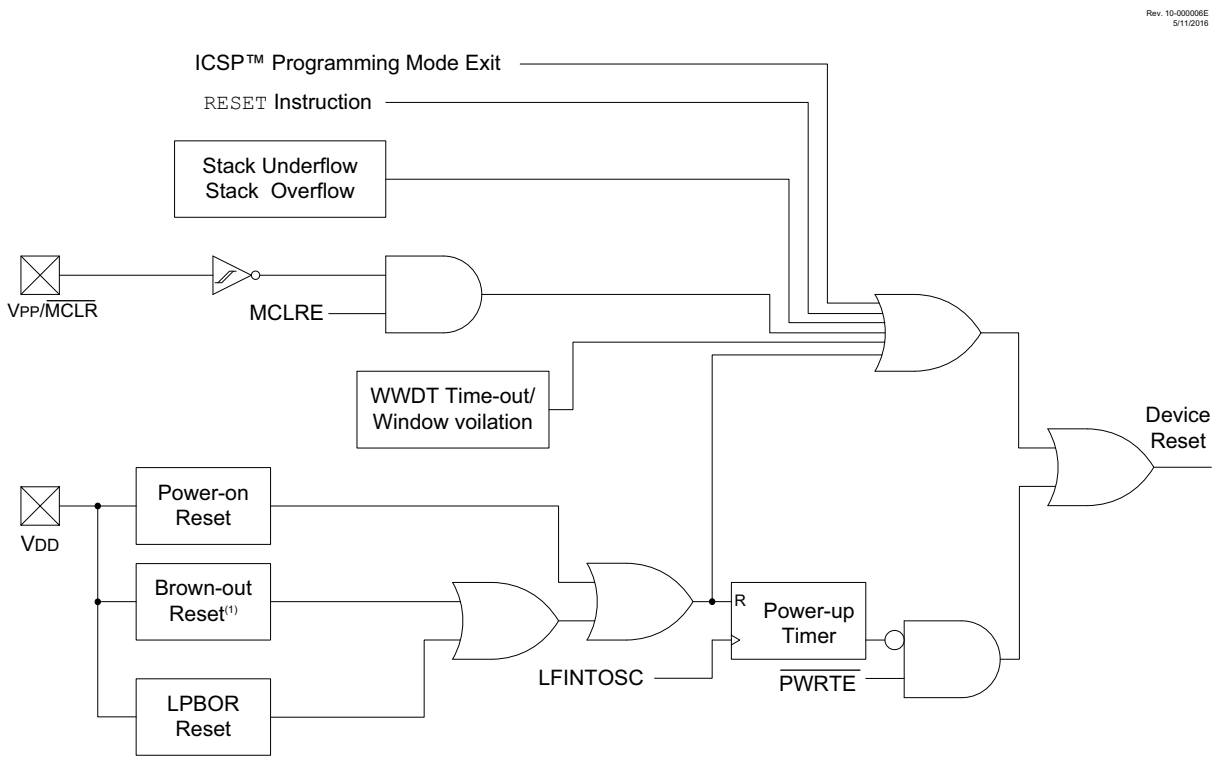
There are multiple ways to reset this device:

- Power-on Reset (POR)
- Brown-out Reset (BOR)
- Low-Power Brown-Out Reset (LPBOR)
- $\overline{\text{MCLR}}$ Reset
- WDT Reset
- RESET instruction
- Stack Overflow
- Stack Underflow
- Programming mode exit

To allow V_{DD} to stabilize, an optional Power-up Timer can be enabled to extend the Reset time after a BOR or POR event.

A simplified block diagram of the On-Chip Reset Circuit is shown in Figure 8-1.

FIGURE 8-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



Note 1: See Table 8-1 for BOR active conditions.

11.1.3 READING THE PROGRAM FLASH MEMORY

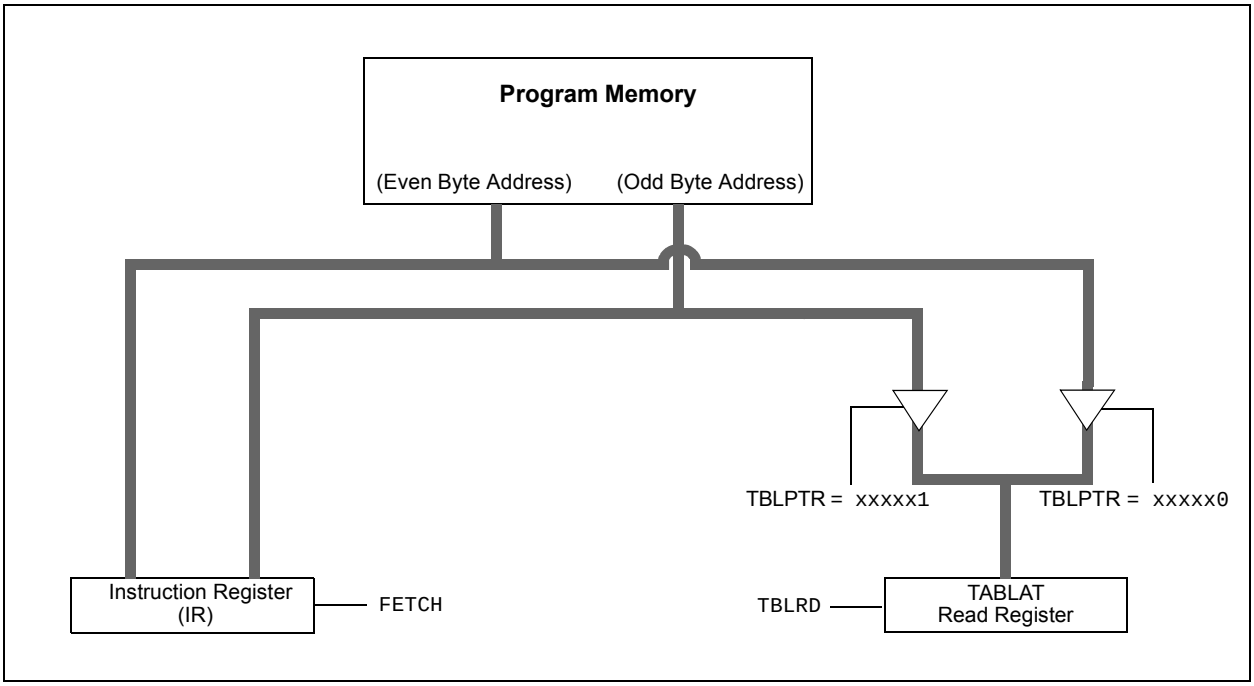
The TBLRD instruction retrieves data from program memory and places it into data RAM. Table reads from program memory are performed one byte at a time.

TBLPTR points to a byte address in program space. Executing TBLRD places the byte pointed to into TABLAT. In addition, TBLPTR can be modified automatically for the next table read operation.

The CPU operation is suspended during the read, and it resumes immediately after. From the user point of view, TABLAT is valid in the next instruction cycle.

The internal program memory is typically organized by words. The Least Significant bit of the address selects between the high and low bytes of the word. Figure 11-4 shows the interface between the internal program memory and the TABLAT.

FIGURE 11-4: READS FROM PROGRAM FLASH MEMORY



EXAMPLE 11-1: READING A PROGRAM FLASH MEMORY WORD

```
            MOVLW     CODE_ADDR_UPPER         ; Load TBLPTR with the base
            MOVWF     TBLPTRU                 ; address of the word
            MOVLW     CODE_ADDR_HIGH
            MOVWF     TBLPTRH
            MOVLW     CODE_ADDR_LOW
            MOVWF     TBLPTRL
READ_WORD
            TBLRD*+                           ; read into TABLAT and increment
            MOVF      TABLAT, W               ; get data
            MOVWF     WORD_EVEN
            TBLRD*+                           ; read into TABLAT and increment
            MOVFW     TABLAT, W               ; get data
            MOVF      WORD_ODD
```

12.0 8x8 HARDWARE MULTIPLIER

12.1 Introduction

All PIC18 devices include an 8x8 hardware multiplier as part of the ALU. The multiplier performs an unsigned operation and yields a 16-bit result that is stored in the product register pair, PRODH:PRODL. The multiplier's operation does not affect any flags in the STATUS register.

Making multiplication a hardware operation allows it to be completed in a single instruction cycle. This has the advantages of higher computational throughput and reduced code size for multiplication algorithms and allows the PIC18 devices to be used in many applications previously reserved for digital signal processors. A comparison of various hardware and software multiply operations, along with the savings in memory and execution time, is shown in Table 12-1.

12.2 Operation

Example 12-1 shows the instruction sequence for an 8x8 unsigned multiplication. Only one instruction is required when one of the arguments is already loaded in the WREG register.

Example 12-2 shows the sequence to do an 8x8 signed multiplication. To account for the sign bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EXAMPLE 12-1: 8x8 UNSIGNED MULTIPLY ROUTINE

```
MOVF   ARG1, W   ;
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
```

EXAMPLE 12-2: 8x8 SIGNED MULTIPLY ROUTINE

```
MOVF   ARG1, W
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
BTFSC  ARG2, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;          - ARG1
MOVF   ARG2, W
BTFSC  ARG1, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;          - ARG2
```

TABLE 12-1: PERFORMANCE COMPARISON FOR VARIOUS MULTIPLY OPERATIONS

Routine	Multiply Method	Program Memory (Words)	Cycles (Max)	Time			
				@ 64 MHz	@ 40 MHz	@ 10 MHz	@ 4 MHz
8x8 unsigned	Without hardware multiply	13	69	4.3 μs	6.9 μs	27.6 μs	69 μs
	Hardware multiply	1	1	62.5 ns	100 ns	400 ns	1 μs
8x8 signed	Without hardware multiply	33	91	5.7 μs	9.1 μs	36.4 μs	91 μs
	Hardware multiply	6	6	375 ns	600 ns	2.4 μs	6 μs
16x16 unsigned	Without hardware multiply	21	242	15.1 μs	24.2 μs	96.8 μs	242 μs
	Hardware multiply	28	28	1.8 μs	2.8 μs	11.2 μs	28 μs
16x16 signed	Without hardware multiply	52	254	15.9 μs	25.4 μs	102.6 μs	254 μs
	Hardware multiply	35	40	2.5 μs	4.0 μs	16.0 μs	40 μs

TABLE 13-5: SUMMARY OF REGISTERS ASSOCIATED WITH CRC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
CRCACCH	ACC<15:8>								148
CRCACCL	ACC<7:0>								149
CRCCON0	EN	GO	BUSY	ACCM	—	—	SHIFTM	FULL	147
CRCCON1	DLEN<3:0>				PLEN<3:0>				147
CRCDATH	DATA<15:8>								148
CRCDATL	DATA<7:0>								148
CRCSHIFTH	SHIFT<15:8>								149
CRCSHIFTL	SHIFT<7:0>								149
CRCXORH	X<15:8>								150
CRCXORL	X<7:1>							—	150
PMD0	SYSCMD	FVRMD	HLVDMD	CRCMD	SCANMD	NVMMD	CLKRMD	IOCMD	64
SCANCON0	SCANEN	SCANGO	BUSY	INVALID	INTM	—	MODE<1:0>		151
SCANHADRU	—	—	HADR<21:16>						153
SCANHADRH	HADR<15:8>								154
SCANHADRL	HADR<7:0>								154
SCANLADRU	—	—	LADR<21:16>						152
SCANLADRH	LADR<15:8>								152
SCANLADRL	LADR<7:0>								153
SCANTRIG	—	—	—	—	TSEL<3:0>				155
INTCON	GIE/GIEH	PEIE/GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG	166
PIR7	SCANIF	CRCIF	NVMIF	—	—	—	—	CWG1IF	174
PIE7	SCANIE	CRCIE	NVMIE	—	—	—	—	CWG1IE	182
IPR7	SCANIP	CRCIP	NVMIP	—	—	—	—	CWG1IP	190

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for the CRC module.

17.3 Bidirectional Pins

PPS selections for peripherals with bidirectional signals on a single pin must be made so that the PPS input and PPS output select the same pin. Peripherals that have bidirectional signals include:

- EUSART (synchronous operation)
- MSSP (I^2C)
- CCP module

Note:	The I^2C default input pins are I^2C and SMBus compatible. RB1 and RB2 are additional pins. RC4 and RC3 are default MMP1 pins and are SMBus compatible. Clock and data signals can be routed to any pin, however pins without I^2C compatibility will operate at standard TTL/ST logic levels as selected by the INVLV register.

17.4 PPS Lock

The PPS includes a mode in which all input and output selections can be locked to prevent inadvertent changes. PPS selections are locked by setting the PPSLOCKED bit of the PPSLOCK register. Setting and clearing this bit requires a special sequence as an extra precaution against inadvertent changes. Examples of setting and clearing the PPSLOCKED bit are shown in Example 17-1.

EXAMPLE 17-1: PPS LOCK SEQUENCE

```
; Disable interrupts:
    BCF      INTCON,GIE

; Bank to PPSLOCK register
    BANKSEL  PPSLOCK
    MOVLB    PPSLOCK
    MOVLW    55h

; Required sequence, next 4 instructions
    MOVWF    PPSLOCK
    MOVLW    AAh
    MOVWF    PPSLOCK

; Set PPSLOCKED bit to disable writes
; Only a BSF instruction will work
    BSF      PPSLOCK,0

; Enable Interrupts
    BSF      INTCON,GIE
```

EXAMPLE 17-2: PPS UNLOCK SEQUENCE

```
; Disable interrupts:
    BCF      INTCON,GIE

; Bank to PPSLOCK register
    BANKSEL  PPSLOCK
    MOVLB    PPSLOCK
    MOVLW    55h

; Required sequence, next 4 instructions
    MOVWF    PPSLOCK
    MOVLW    AAh
    MOVWF    PPSLOCK

; Clear PPSLOCKED bit to enable writes
; Only a BCF instruction will work
    BCF      PPSLOCK,0

; Enable Interrupts
    BSF      INTCON,GIE
```

17.5 PPS One-Way Lock

Using the PPS1WAY Configuration bit, the PPS settings can be locked in. When this bit is set, the PPSLOCKED bit can only be cleared and set one time after a device Reset. This allows for clearing the PPSLOCKED bit so that the input and output selections can be made during initialization. When the PPSLOCKED bit is set after all selections have been made, it will remain set and cannot be cleared until after the next device Reset event.

17.6 Operation During Sleep

PPS input and output selections are unaffected by Sleep.

17.7 Effects of a Reset

A device Power-on-Reset (POR) clears all PPS input and output selections to their default values. All other Resets leave the selections unchanged. Default input selections are shown in the **Section "Pin Allocation Tables"**.The PPS one-way is also removed.

REGISTER 19-4: TMRxGATE: TIMERx GATE ISM REGISTER

U-0	U-0	U-0	U-0	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	—	GSS<3:0>			
bit 7							bit 0

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

bit 7-4 **Unimplemented:** Read as ‘0’

bit 3-0 **GSS<3:0>:** Timerx Gate Source Selection bits

GSS	Timer1	Timer3	Timer5
	Gate Source	Gate Source	Gate Source
1111	Reserved	Reserved	Reserved
1110	ZCDOUT	ZCDOUT	ZCDOUT
1101	CMP2OUT	CMP2OUT	CMP2OUT
1100	CMP1OUT	CMP1OUT	CMP1OUT
1011	PWM4OUT	PWM4OUT	PWM4OUT
1010	PWM3OUT	PWM3OUT	PWM3OUT
1001	CCP2OUT	CCP2OUT	CCP2OUT
1000	CCP1OUT	CCP1OUT	CCP1OUT
0111	TMR6OUT (post-scaled)	TMR6OUT (post-scaled)	TMR6OUT (post-scaled)
0110	TMR5 overflow	TMR5 overflow	Reserved
0101	TMR4OUT (post-scaled)	TMR4OUT (post-scaled)	TMR4OUT (post-scaled)
0100	TMR3 overflow	Reserved	TMR3 overflow
0011	TMR2OUT (post-scaled)	TMR2OUT (post-scaled)	TMR2OUT (post-scaled)
0010	Reserved	TMR1 overflow	TMR1 overflow
0001	TMR0 overflow	TMR0 overflow	TMR0 overflow
0000	Pin selected by T1GPPS	Pin selected by T3GPPS	Pin selected by T5GPPS

20.5 Operation Examples

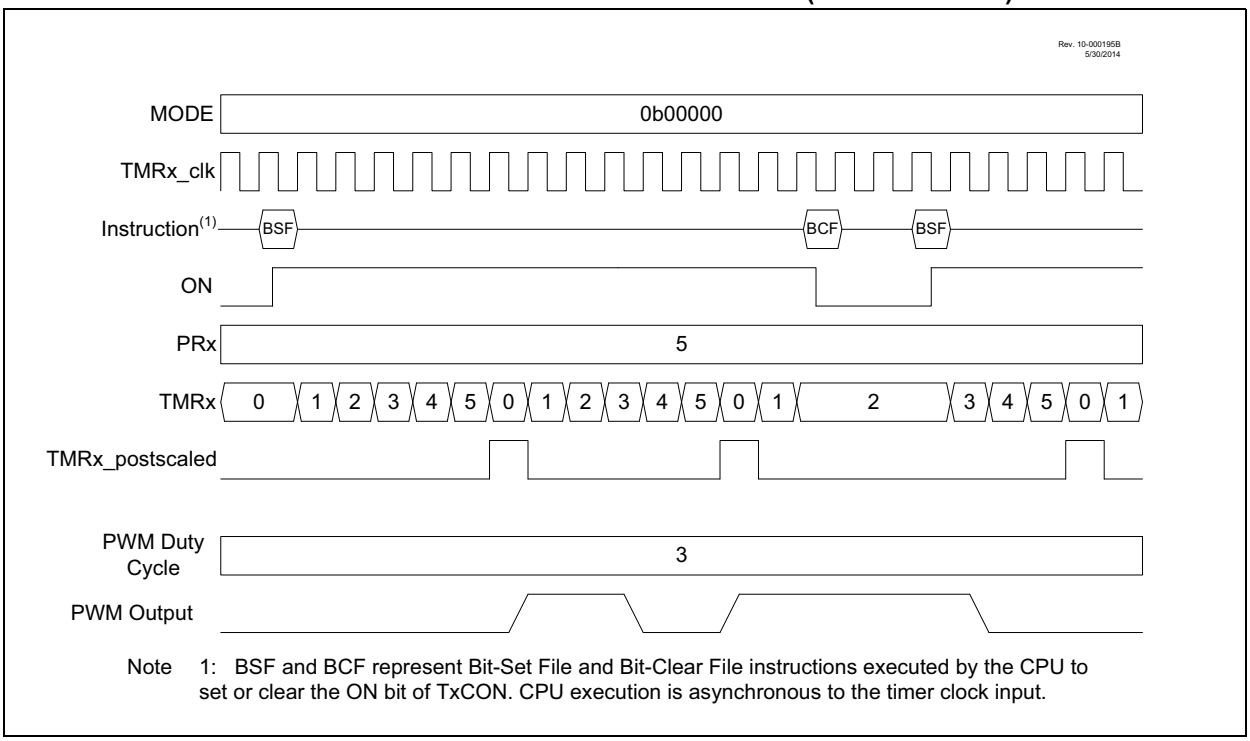
Unless otherwise specified, the following notes apply to the following timing diagrams:

- Both the prescaler and postscaler are set to 1:1 (both the CKPS and OUTPS bits in the TxCON register are cleared).
- The diagrams illustrate any clock except Fosc/4 and show clock-sync delays of at least two full cycles for both ON and Timer2_ers. When using Fosc/4, the clock-sync delay is at least one instruction period for Timer2_ers; ON applies in the next instruction period.
- The PWM Duty Cycle and PWM output are illustrated assuming that the timer is used for the PWM function of the CCP module as described in **Section 21.0 "Capture/Compare/PWM Module"**. The signals are not a part of the Timer2 module.

20.5.1 SOFTWARE GATE MODE

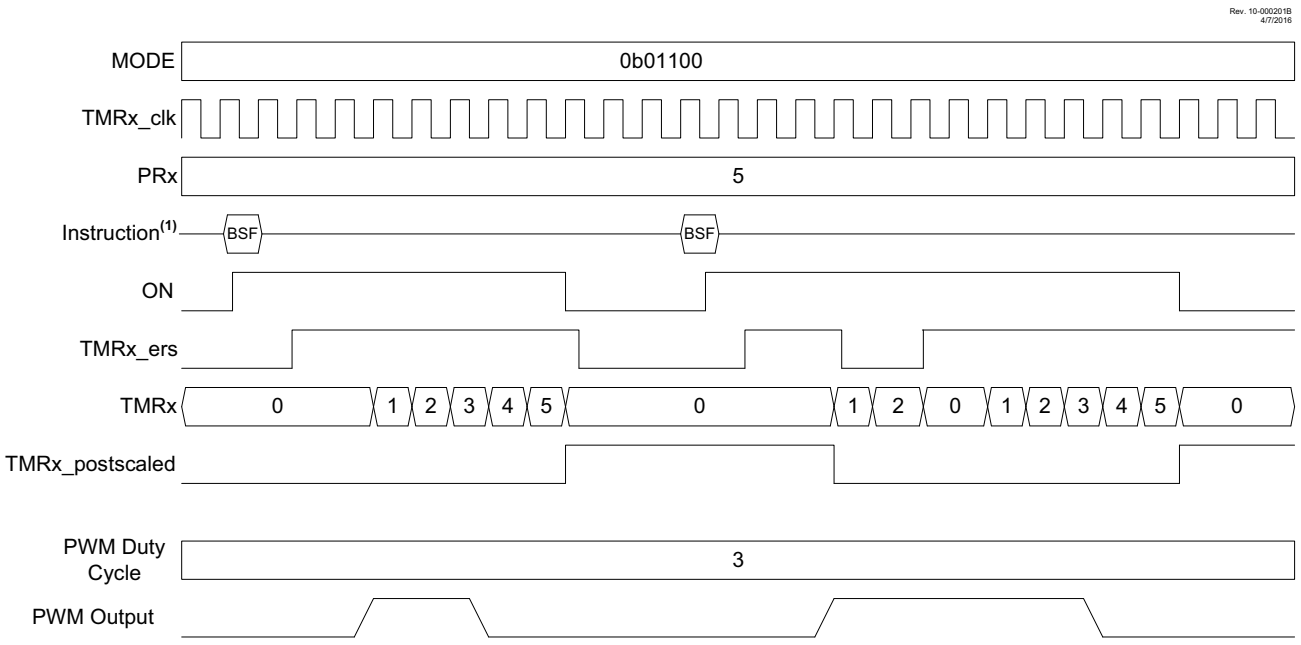
This mode corresponds to legacy Timer2 operation. The timer increments with each clock input when ON = 1 and does not increment when ON = 0. When the TMRx count equals the PRx period count the timer resets on the next clock and continues counting from 0. Operation with the ON bit software controlled is illustrated in Figure 20-4. With PRx = 5, the counter advances until TMRx = 5, and goes to zero with the next clock.

FIGURE 20-4: SOFTWARE GATE MODE TIMING DIAGRAM (MODE = 00000)



Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

FIGURE 20-10: EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = 01100)



Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

20.7 Register Definitions: Timer2/4/6 Control

Long bit name prefixes for the Timer2/4/6 peripherals are shown in Table 20-2. Refer to **Section 1.4.2.2 "Long Bit Names"** for more information.

TABLE 20-2:

Peripheral	Bit Name Prefix
Timer2	T2
Timer4	T4
Timer6	T6

REGISTER 20-3: TxCLKCON: TIMERx CLOCK SELECTION REGISTER

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	CS<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 **Unimplemented:** Read as ‘0’

bit 3-0 **CS<3:0>:** Timerx Clock Selection bits

CS<3:0>	TMR2	TMR4	TMR6
	Clock Source	Clock Source	Clock Source
1111-1001	Reserved	Reserved	Reserved
1000	ZCD_OUT	ZCD_OUT	ZCD_OUT
0111	CLKREF_OUT	CLKREF_OUT	CLKREF_OUT
0110	SOSC	SOSC	SOSC
0101	MFINTOSC (31 kHz)	MFINTOSC (31 kHz)	MFINTOSC (31 kHz)
0100	LFINTOSC	LFINTOSC	LFINTOSC
0011	HFINTOSC	HFINTOSC	HFINTOSC
0010	Fosc	Fosc	Fosc
0001	Fosc/4	Fosc/4	Fosc/4
0000	Pin selected by T2INPPS	Pin selected by T4INPPS	Pin selected by T6INPPS

REGISTER 21-2: CCPTMRS: CCP TIMERS CONTROL REGISTER

R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1
P4TSEL<1:0>		P3TSEL<1:0>		C2TSEL<1:0>		C1TSEL<1:0>	
bit 7							bit 0

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

bit 7-6 **P4TSEL<1:0>:** PWM4 Timer Selection bits
11 = PWM4 based on TMR6
10 = PWM4 based on TMR4
01 = PWM4 based on TMR2
00 = Reserved

bit 5-4 **P3TSEL<1:0>:** PWM3 Timer Selection bits
11 = PWM3 based on TMR6
10 = PWM3 based on TMR4
01 = PWM3 based on TMR2
00 = Reserved

bit 3-2 **C2TSEL<1:0>:** CCP2 Timer Selection bits
11 = CCP2 is based off Timer5 in Capture/Compare mode and Timer6 in PWM mode
10 = CCP2 is based off Timer3 in Capture/Compare mode and Timer4 in PWM mode
01 = CCP2 is based off Timer1 in Capture/Compare mode and Timer2 in PWM mode
00 = Reserved

bit 1-0 **C1TSEL<1:0>:** CCP1 Timer Selection bits
11 = CCP1 is based off Timer5 in Capture/Compare mode and Timer6 in PWM mode
10 = CCP1 is based off Timer3 in Capture/Compare mode and Timer4 in PWM mode
01 = CCP1 is based off Timer1 in Capture/Compare mode and Timer2 in PWM mode
00 = Reserved

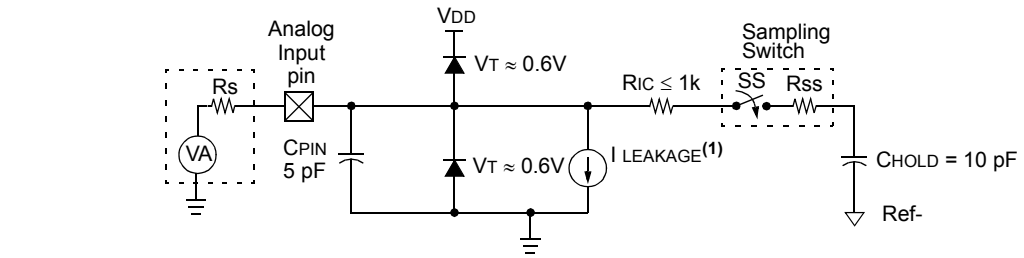
TABLE 27-5: SAMPLE BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

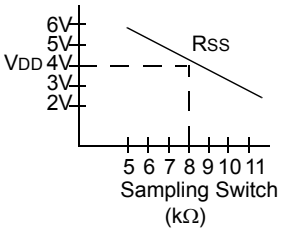
BAUD RATE	SYNC = 0, BRGH = 1, BRG16 = 0											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 3.6864 MHz			Fosc = 1.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	—	—	—	—	—	—	—	—	—	300	0.16	207
1200	—	—	—	1202	0.16	207	1200	0.00	191	1202	0.16	51
2400	2404	0.16	207	2404	0.16	103	2400	0.00	95	2404	0.16	25
9600	9615	0.16	51	9615	0.16	25	9600	0.00	23	—	—	—
10417	10417	0.00	47	10417	0.00	23	10473	0.53	21	10417	0.00	5
19.2k	19231	0.16	25	19.23k	0.16	12	19.2k	0.00	11	—	—	—
57.6k	55556	-3.55	8	—	—	—	57.60k	0.00	3	—	—	—
115.2k	—	—	—	—	—	—	115.2k	0.00	1	—	—	—

BAUD RATE	SYNC = 0, BRGH = 0, BRG16 = 1											
	Fosc = 32.000 MHz			Fosc = 20.000 MHz			Fosc = 18.432 MHz			Fosc = 11.0592 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	300.0	0.00	6666	300.0	-0.01	4166	300.0	0.00	3839	300.0	0.00	2303
1200	1200	-0.02	3332	1200	-0.03	1041	1200	0.00	959	1200	0.00	575
2400	2401	-0.04	832	2399	-0.03	520	2400	0.00	479	2400	0.00	287
9600	9615	0.16	207	9615	0.16	129	9600	0.00	119	9600	0.00	71
10417	10417	0.00	191	10417	0.00	119	10378	-0.37	110	10473	0.53	65
19.2k	19.23k	0.16	103	19.23k	0.16	64	19.20k	0.00	59	19.20k	0.00	35
57.6k	57.14k	-0.79	34	56.818	-1.36	21	57.60k	0.00	19	57.60k	0.00	11
115.2k	117.6k	2.12	16	113.636	-1.36	10	115.2k	0.00	9	115.2k	0.00	5

BAUD RATE	SYNC = 0, BRGH = 0, BRG16 = 1											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 3.6864 MHz			Fosc = 1.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	299.9	-0.02	1666	300.1	0.04	832	300.0	0.00	767	300.5	0.16	207
1200	1199	-0.08	416	1202	0.16	207	1200	0.00	191	1202	0.16	51
2400	2404	0.16	207	2404	0.16	103	2400	0.00	95	2404	0.16	25
9600	9615	0.16	51	9615	0.16	25	9600	0.00	23	—	—	—
10417	10417	0.00	47	10417	0.00	23	10473	0.53	21	10417	0.00	5
19.2k	19.23k	0.16	25	19.23k	0.16	12	19.20k	0.00	11	—	—	—
57.6k	55556	-3.55	8	—	—	—	57.60k	0.00	3	—	—	—
115.2k	—	—	—	—	—	—	115.2k	0.00	1	—	—	—

FIGURE 31-4: ANALOG INPUT MODEL



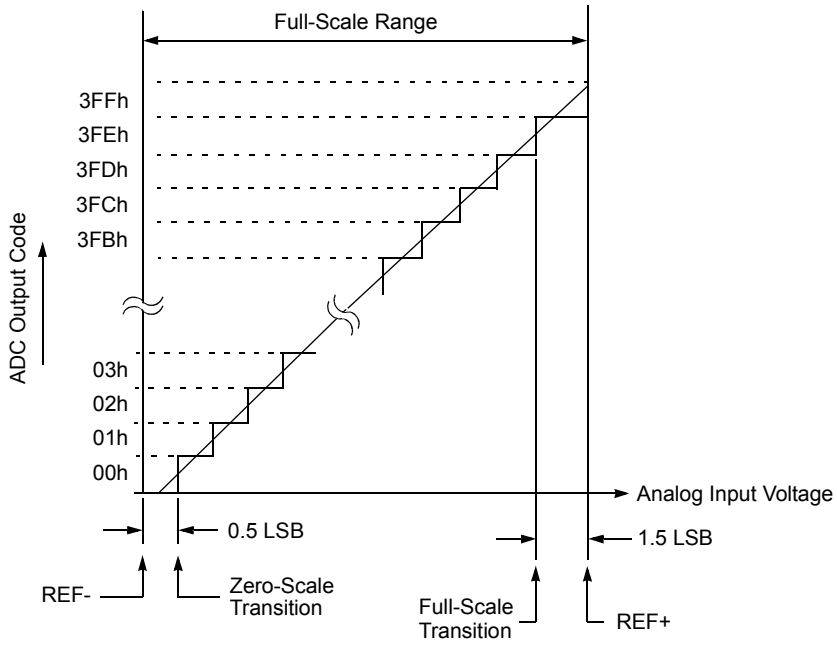


Legend:

CHOLD	=	Sample/Hold Capacitance
CPIN	=	Input Capacitance
I LEAKAGE	=	Leakage current at the pin due to various junctions
RIC	=	Interconnect Resistance
RSS	=	Resistance of Sampling Switch
SS	=	Sampling Switch
VT	=	Threshold Voltage

Note 1: Refer to Table 37-4 (parameter D340 and D341).

FIGURE 31-5: ADC TRANSFER FUNCTION



32.4 Comparator Hysteresis

A selectable amount of separation voltage can be added to the input pins of each comparator to provide a hysteresis function to the overall operation. Hysteresis is enabled by setting the CxHYS bit of the CMxCON0 register.

See Comparator Specifications in Table 37-15 for more information.

32.5 Timer1/3/5 Gate Operation

The output resulting from a comparator operation can be used as a source for gate control of Timer1/3/5. See **Section 19.8 "Timer1/3/5 Gate"** for more information. This feature is useful for timing the duration or interval of an analog event.

It is recommended that the comparator output be synchronized to Timer1. This ensures that Timer1 does not increment while a change in the comparator is occurring.

32.5.1 COMPARATOR OUTPUT SYNCHRONIZATION

The output from a comparator can be synchronized with Timer1 by setting the SYNC bit of the CMxCON0 register.

Once enabled, the comparator output is latched on the falling edge of the Timer1 source clock. If a prescaler is used with Timer1, the comparator output is latched after the prescaling function. To prevent a race condition, the comparator output is latched on the falling edge of the Timer1 clock source and Timer1 increments on the rising edge of its clock source. See the Comparator Block Diagram (Figure 32-2) and the Timer1 Block Diagram (Figure 19-1) for more information.

32.6 Comparator Interrupt

An interrupt can be generated upon a change in the output value of the comparator for each comparator, a rising edge detector and a falling edge detector are present.

When either edge detector is triggered and its associated enable bit is set (CxINTP and/or CxINTN bits of the CMxCON1 register), the Corresponding Interrupt Flag bit (CxIF bit of the PIR2 register) will be set.

To enable the interrupt, you must set the following bits:

- EN and POL bits of the CMxCON0 register
- CxIE bit of the PIE2 register
- INTP bit of the CMxCON1 register (for a rising edge detection)
- INTN bit of the CMxCON1 register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

The associated interrupt flag bit, CxIF bit of the PIR2 register, must be cleared in software. If another edge is detected while this flag is being cleared, the flag will still be set at the end of the sequence.

Note:	Although a comparator is disabled, an interrupt can be generated by changing the output polarity with the CxPOL bit of the CMxCON0 register, or by switching the comparator on or off with the CxEN bit of the CMxCON0 register.

32.7 Comparator Positive Input Selection

Configuring the PCH<2:0> bits of the CMxPCH register directs an internal voltage reference or an analog pin to the non-inverting input of the comparator:

- CxIN0+, CxIN1+ analog pin
- DAC output
- FVR (Fixed Voltage Reference)
- AVSS (Ground)

See **Section 28.0 "Fixed Voltage Reference (FVR)"** for more information on the Fixed Voltage Reference module.

See **Section 30.0 "5-Bit Digital-to-Analog Converter (DAC) Module"** for more information on the DAC input signal.

Any time the comparator is disabled (CxEN = 0), all comparator inputs are disabled.

32.8 Comparator Negative Input Selection

The NCH<2:0> bits of the CMxNCH register direct an analog input pin and internal reference voltage or analog ground to the inverting input of the comparator:

- CxIN0-, CxIN1-, CxIN2-, CxIN3- analog pin
- FVR (Fixed Voltage Reference)
- Analog Ground

Note:	To use CxINy+ and CxINy- pins as analog input, the appropriate bits must be set in the ANSEL register and the corresponding TRIS bits must also be set to disable the output drivers.

LFSR **Load FSR**

Syntax: LFSR f, k

Operands: $0 \le f \le 2$
$0 \le k \le 4095$

Operation: $k \rightarrow FSRf$

Status Affected: None

Encoding:

1110 1111	1110 0000	00ff $k_7$kkk	k_{11}kkk kkkk

Description: The 12-bit literal 'k' is loaded into the File Select Register pointed to by 'f'.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k' MSB	Process Data	Write literal 'k' MSB to FSRfH
Decode	Read literal 'k' LSB	Process Data	Write literal 'k' to FSRfL

Example: `LFSR 2, 3ABh`

After Instruction
FSR2H = 03h
FSR2L = ABh

MOVF **Move f**

Syntax: MOVF f {,d {,a}}

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

Operation: $f \rightarrow dest$

Status Affected: N, Z

Encoding:

0101	00da	ffff	ffff

Description: The contents of register 'f' are moved to a destination dependent upon the status of 'd'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default). Location 'f' can be anywhere in the 256-byte bank.
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 35.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `MOVF REG, 0, 0`

Before Instruction
REG = 22h
W = FFh
After Instruction
REG = 22h
W = 22h

RRNCF **Rotate Right f (No Carry)**

Syntax: RRNCF f {,d {,a}}

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

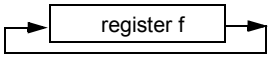
Operation: (f<n>) → dest<n – 1>,
(f<0>) → dest<7>

Status Affected: N, Z

Encoding:

0100	00da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the right. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).
If 'a' is '0', the Access Bank will be selected (default), overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 35.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1:
```
RRNCF   REG, 1, 0
Before Instruction
    REG     =   1101 0111
After Instruction
    REG     =   1110 1011
```

Example 2:
```
RRNCF   REG, 0, 0
Before Instruction
    W       =   ?
    REG     =   1101 0111
After Instruction
    W       =   1110 1011
    REG     =   1101 0111
```

SETF **Set f**

Syntax: SETF f {,a}

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

Operation: FFh → f

Status Affected: None

Encoding:

0110	100a	ffff	ffff

Description: The contents of the specified register are set to FFh.
If 'a' is '0', the Access Bank is selected.
If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 35.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example:
```
SETF    REG, 1
Before Instruction
    REG     =   5Ah
After Instruction
    REG     =   FFh
```