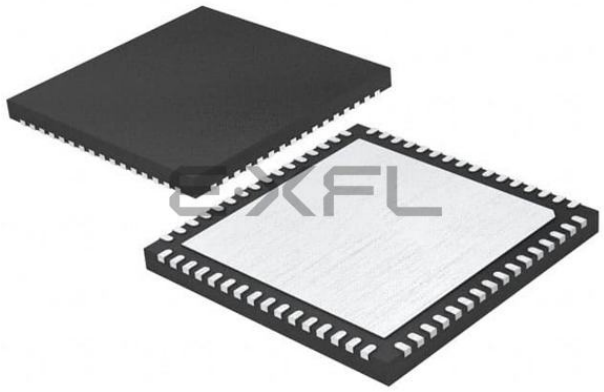


Welcome to E-XFL.COM

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	53
Program Memory Size	128KB (64K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 16x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-VFQFN Exposed Pad
Supplier Device Package	64-VQFN (9x9)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f67k22-i-mrrsl

REGISTER 5-1: RCON: RESET CONTROL REGISTER

R/W-0	R/W-1	R/W-1	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	SBOREN	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
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 **IPEN:** Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (PIC16CXXX Compatibility mode)

bit 6 **SBOREN:** BOR Software Enable bit
If BOREN<1:0> = 01:
1 = BOR is enabled
0 = BOR is disabled
If BOREN<1:0> = 00, 10 or 11:
Bit is disabled and read as '0'.

bit 5 **$\overline{CM}$:** Configuration Mismatch Flag bit
1 = A Configuration Mismatch Reset has not occurred
0 = A Configuration Mismatch Reset has occurred (must be set in software after a Configuration Mismatch Reset occurs)

bit 4 **$\overline{RI}$:** RESET Instruction Flag bit
1 = The RESET instruction was not executed (set by firmware only)
0 = The RESET instruction was executed causing a device Reset (must be set in software after a Brown-out Reset occurs)

bit 3 **$\overline{TO}$:** Watchdog Time-out Flag bit
1 = Set by power-up, CLRWDT instruction or SLEEP instruction
0 = A WDT time-out occurred

bit 2 **$\overline{PD}$:** Power-Down Detection Flag bit
1 = Set by power-up or by the CLRWDT instruction
0 = Set by execution of the SLEEP instruction

bit 1 **$\overline{POR}$:** Power-on Reset Status bit
1 = A Power-on Reset has not occurred (set by firmware only)
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit
1 = A Brown-out Reset has not occurred (set by firmware only)
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

Note 1: It is recommended that the $\overline{POR}$ bit be set after a Power-on Reset has been detected, so that subsequent Power-on Resets may be detected.

2: Brown-out Reset is said to have occurred when $\overline{BOR}$ is '0' and $\overline{POR}$ is '1' (assuming that $\overline{POR}$ was set to '1' by software immediately after a Power-on Reset).

TABLE 5-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets, WDT Reset, RESET Instruction, Stack Resets, CM Resets	Wake-up via WDT or Interrupt
ODCON2	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	uuuu uuuu	uuuu uuuu
ODCON2	PIC18F65K22	PIC18F85K22	--00 0000	--uu uuuu	--uu uuuu
ODCON3	PIC18F6XK22	PIC18F8XK22	00-- ---0	uu-- ---u	uu-- ---u
MEMCON	PIC18F6XK22	PIC18F8XK22	0-00 --00	0-00 --00	u-uu --uu
ANCON0	PIC18F6XK22	PIC18F8XK22	1111 1111	uuuu uuuu	uuuu uuuu
ANCON1	PIC18F6XK22	PIC18F8XK22	1111 1111	uuuu uuuu	uuuu uuuu
ANCON2	PIC18F6XK22	PIC18F8XK22	1111 1111	uuuu uuuu	uuuu uuuu
RCSTA2	PIC18F6XK22	PIC18F8XK22	0000 000x	0000 000x	uuuu uuuu
TXSTA2	PIC18F6XK22	PIC18F8XK22	0000 0010	0000 0010	uuuu uuuu
BAUDCON2	PIC18F6XK22	PIC18F8XK22	0100 0-00	0100 0-00	uuuu u-uu
SPBRGH2	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
SPBRG2	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
RCREG2	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
TXREG2	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	xxxx xxxx	uuuu uuuu
PSTR2CON	PIC18F6XK22	PIC18F8XK22	00-0 0001	00-0 0001	uu-u uuuu
PSTR3CON	PIC18F6XK22	PIC18F8XK22	00-0 0001	00-0 0001	uu-u uuuu
PMD0	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PMD1	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PMD2	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	0000 0000	uuuu uuuu
PMD2	PIC18F65K22	PIC18F85K22	-0-0 0000	-0-0 0000	-u-u uuuu
PMD3	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	0000 0000	uuuu uuuu
PMD3	PIC18F65K22	PIC18F85K22	--00 000-	--00 000-	--uu uuu-

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: 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.

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: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-1 for Reset value for specific condition.

6.4.3.1 FSR Registers and the INDF Operand

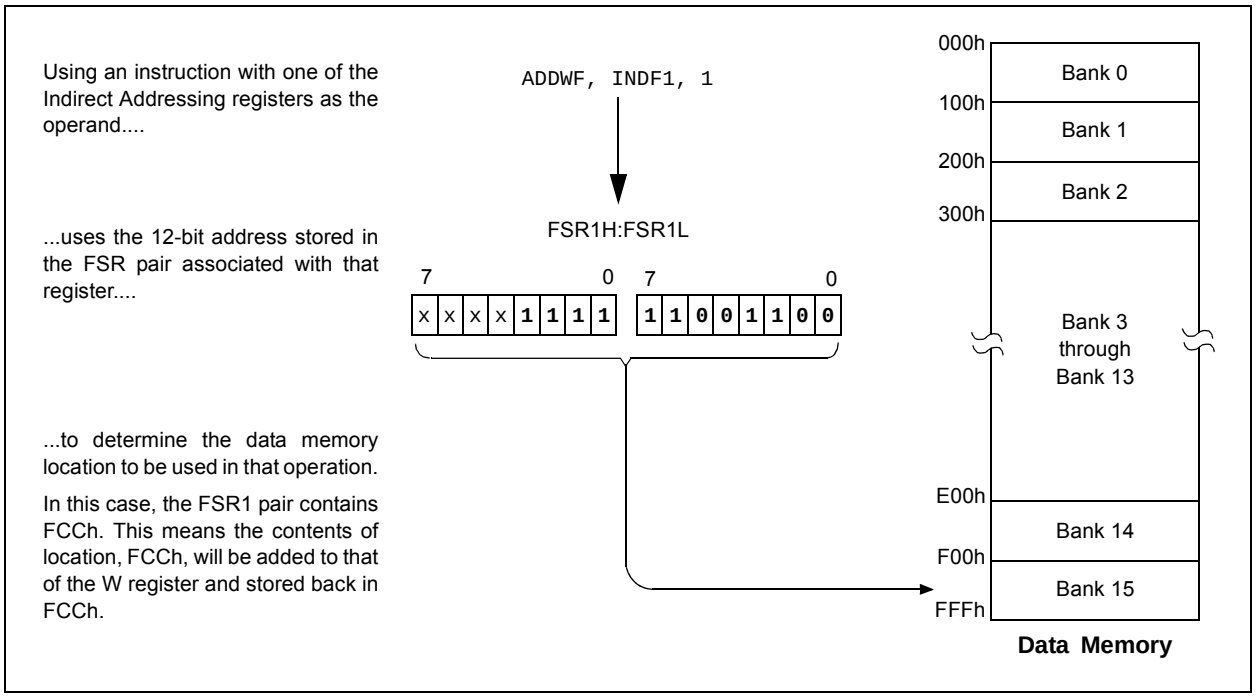
At the core of Indirect Addressing are three sets of registers: FSR0, FSR1 and FSR2. Each represents a pair of 8-bit registers: FSRnH and FSRnL. The four upper bits of the FSRnH register are not used, so each FSR pair holds a 12-bit value. This represents a value that can address the entire range of the data memory in a linear fashion. The FSR register pairs, then, serve as pointers to data memory locations.

Indirect Addressing is accomplished with a set of Indirect File Operands, INDF0 through INDF2. These can be thought of as “virtual” registers. The operands are mapped in the SFR space, but are not physically implemented. Reading or writing to a particular INDF register actually accesses its corresponding FSR register pair. A read from INDF1, for example, reads the data at the address indicated by FSR1H:FSR1L.

Instructions that use the INDF registers as operands actually use the contents of their corresponding FSR as a pointer to the instruction’s target. The INDF operand is just a convenient way of using the pointer.

Because Indirect Addressing uses a full 12-bit address, data RAM banking is not necessary. Thus, the current contents of the BSR and the Access RAM bit have no effect on determining the target address.

FIGURE 6-8: INDIRECT ADDRESSING



8.6.2 16-BIT WORD WRITE MODE

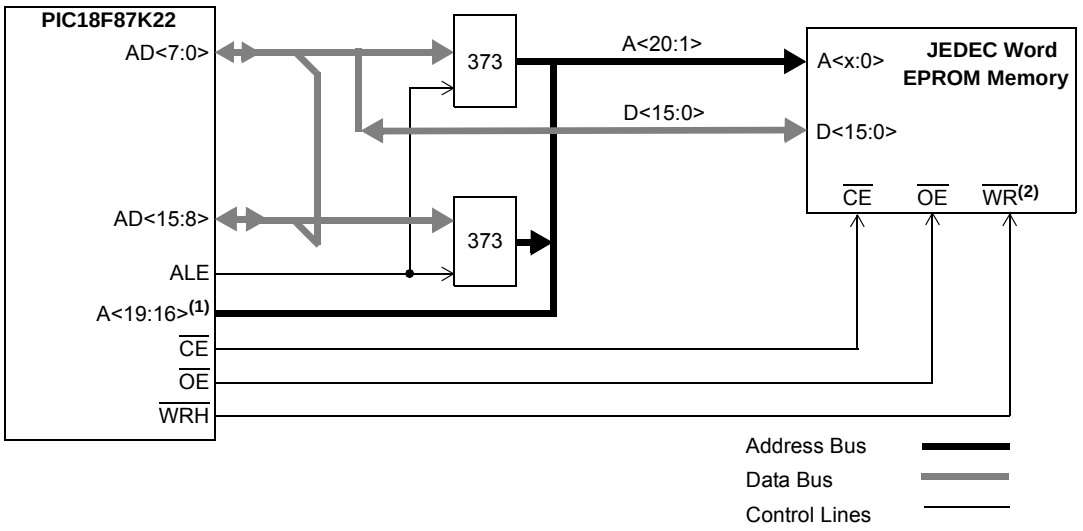
Figure 8-2 shows an example of 16-Bit Word Write mode for PIC18F87K22 family devices. This mode is used for word-wide memories, which includes some of the EPROM and Flash type memories. This mode allows opcode fetches and table reads from all forms of 16-bit memory, and table writes to any type of word-wide external memories. This method makes a distinction between TBLWT cycles to even or odd addresses.

During a TBLWT cycle to an even address (TBLPTR<0> = 0), the TABLAT data is transferred to a holding latch and the external address data bus is tri-stated for the data portion of the bus cycle. No write signals are activated.

During a TBLWT cycle to an odd address (TBLPTR<0> = 1), the TABLAT data is presented on the upper byte of the AD<15:0> bus. The contents of the holding latch are presented on the lower byte of the AD<15:0> bus.

The $\overline{WRH}$ signal is strobed for each write cycle; the $\overline{WRL}$ pin is unused. The signal on the BA0 pin indicates the LSb of the TBLPTR, but it is left unconnected. Instead, the $\overline{UB}$ and $\overline{LB}$ signals are active to select both bytes. The obvious limitation to this method is that the table write must be done in pairs on a specific word boundary to correctly write a word location.

FIGURE 8-2: 16-BIT WORD WRITE MODE EXAMPLE



Note 1: Upper order address lines are used only for 20-bit address widths.

2: This signal only applies to table writes. See **Section 7.1 "Table Reads and Table Writes"**.

Example 10-3 shows the sequence to do a 16 x 16 unsigned multiplication. Equation 10-1 shows the algorithm that is used. The 32-bit result is stored in four registers (RES3:RES0).

EQUATION 10-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L)
\end{aligned}
$$

EXAMPLE 10-3: 16 x 16 UNSIGNED MULTIPLY ROUTINE

```
    MOVF    ARG1L, W
    MULWF   ARG2L           ; ARG1L * ARG2L->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES1     ;
    MOVFF   PRODL, RES0     ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H           ; ARG1H * ARG2H->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES3     ;
    MOVFF   PRODL, RES2     ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H           ; ARG1L * ARG2H->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    MOVF    ARG1H, W        ;
    MULWF   ARG2L           ; ARG1H * ARG2L->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
```

Example 10-4 shows the sequence to do a 16 x 16 signed multiply. Equation 10-2 shows the algorithm used. The 32-bit result is stored in four registers (RES3:RES0). To account for the sign bits of the arguments, the MSb for each argument pair is tested and the appropriate subtractions are done.

EQUATION 10-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L) + \\
&\quad (-1 \bullet ARG2H\langle 7\rangle \bullet ARG1H:ARG1L \bullet 2^{16}) + \\
&\quad (-1 \bullet ARG1H\langle 7\rangle \bullet ARG2H:ARG2L \bullet 2^{16})
\end{aligned}
$$

EXAMPLE 10-4: 16 x 16 SIGNED MULTIPLY ROUTINE

```
   MOVF    ARG1L, W
   MULWF   ARG2L        ; ARG1L * ARG2L ->
                        ; PRODH:PRODL
   MOVFF   PRODH, RES1 ;
   MOVFF   PRODL, RES0 ;
;
   MOVF    ARG1H, W
   MULWF   ARG2H        ; ARG1H * ARG2H ->
                        ; PRODH:PRODL
   MOVFF   PRODH, RES3 ;
   MOVFF   PRODL, RES2 ;
;
   MOVF    ARG1L, W
   MULWF   ARG2H        ; ARG1L * ARG2H ->
                        ; PRODH:PRODL
   MOVF    PRODL, W     ;
   ADDWF   RES1, F      ; Add cross
   MOVF    PRODH, W     ; products
   ADDWFC  RES2, F      ;
   CLRF    WREG         ;
   ADDWFC  RES3, F      ;
;
   MOVF    ARG1H, W     ;
   MULWF   ARG2L        ; ARG1H * ARG2L ->
                        ; PRODH:PRODL
   MOVF    PRODL, W     ;
   ADDWF   RES1, F      ; Add cross
   MOVF    PRODH, W     ; products
   ADDWFC  RES2, F      ;
   CLRF    WREG         ;
   ADDWFC  RES3, F      ;
;
   BTFSS   ARG2H, 7     ; ARG2H:ARG2L neg?
   BRA     SIGN_ARG1    ; no, check ARG1
   MOVF    ARG1L, W     ;
   SUBWF   RES2         ;
   MOVF    ARG1H, W     ;
   SUBWFB  RES3         ; SIGN_ARG1
   BTFSS   ARG1H, 7     ; ARG1H:ARG1L neg?
   BRA     CONT_CODE    ; no, done
   MOVF    ARG2L, W     ;
   SUBWF   RES2         ;
   MOVF    ARG2H, W     ;
   SUBWFB  RES3
;
CONT_CODE
   :
```

REGISTER 11-15: PIE6: PERIPHERAL INTERRUPT ENABLE REGISTER 6

U-0	U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	EEIE	—	CMP3IE	CMP2IE	CMP1IE
bit 7							bit 0

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

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **EEIE:** Data EEDATA/Flash Write Operation Enable bit
1 = Interrupt is enabled
0 = interrupt is disabled

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

bit 2 **CMP3IE:** CMP3 Enable bit
1 = Interrupt is enabled
0 = interrupt is disabled

bit 1 **CMP2E:** CMP2 Enable bit
1 = Interrupt is enabled
0 = interrupt is disabled

bit 0 **CMP1IE:** CMP1 Enable bit
1 = Interrupt is enabled
0 = interrupt is disabled

TABLE 12-15: PORTH FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RH4/CCP9/ P3C/AN12/ C2INC	RH4	0	O	DIG	LATH<4> data output.
		1	I	ST	PORTH<4> data input.
	CCP9	0	O	DIG	CCP9 compare/PWM output; takes priority over port data.
		1	I	ST	CCP9 capture input.
	P3C	0	O	—	ECCP3 PWM Output C. May be configured for tri-state during Enhanced PWM.
	AN12	1	I	ANA	A/D Input Channel 12. Default input configuration on POR; does not affect digital input.
	C2INC	x	I	ANA	Comparator 2 Input C.
RH5/CCP8/ P3B/AN13/ C2IND	RH5	0	O	DIG	LATH<5> data output.
		1	I	ST	PORTH<5> data input.
	CCP8	0	O	DIG	CCP8 compare/PWM output; takes priority over port data.
		1	I	ST	CCP8 capture input.
	P3B	0	O	—	ECCP3 PWM Output B. May be configured for tri-state during Enhanced PWM.
	AN13	1	I	ANA	A/D Input Channel 13. Default input configuration on POR; does not affect digital input.
	C2IND	x	I	ANA	Comparator 2 Input D.
RH6/CCP7/ P1C/AN14/ C1INC	RH6	0	O	DIG	LATH<6> data output.
		1	I	ST	PORTH<6> data input.
	CCP7	0	O	DIG	CCP7 compare/PWM output; takes priority over port data.
		1	I	ST	CCP7 capture input.
	P1C	0	O	—	ECCP1 PWM Output C. May be configured for tri-state during Enhanced PWM.
	AN14	1	I	ANA	A/D Input Channel 14. Default input configuration on POR; does not affect digital input.
	C1INC	x	I	ANA	Comparator 1 Input C.
RH7/CCP6/ P1B/AN15	RH7	0	O	DIG	LATH<7> data output.
		1	I	ST	PORTH<7> data input.
	CCP6	0	O	DIG	CCP6 compare/PWM output; takes priority over port data.
		1	I	ST	CCP6 capture input.
	P1B	0	O	—	ECCP1 PWM Output B. May be configured for tri-state during Enhanced PWM.
	AN15	1	I	ANA	A/D Input Channel 15. Default input configuration on POR; does not affect digital input.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input,
x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

18.1 RTCC MODULE REGISTERS

The RTCC module registers are divided into the following categories:

RTCC Control Registers

- RTCCFG
- RTCCAL
- PADCFG1
- ALRMCFG
- ALRMRPT

RTCC Value Registers

- RTCVALH
- RTCVALL

  Both registers access the following registers:
  - YEAR
  - MONTH
  - DAY
  - WEEKDAY
  - HOUR
  - MINUTE
  - SECOND

Alarm Value Registers

- ALRMVALH
- ALRMVALL

  Both registers access the following registers:
  - ALRMMNTH
  - ALRMDAY
  - ALRMWD
  - ALRMHR
  - ALRMMIN
  - ALRMSEC

Note:	The RTCVALH and RTCVALL registers can be accessed through RTCRPT<1:0> (RTCCFG<1:0>). ALRMVALH and ALRMVALL can be accessed through ALRMPTR<1:0> (ALRMCFG<1:0>).

18.1.4 RTCEN BIT WRITE

RTCWREN (RTCCFG<5>) must be set before a write to RTCEN can take place. Any write to the RTCEN bit, while RTCWREN = 0, will be ignored.

Like the RTCEN bit, the RTCVALH and RTCVALL registers can only be written to when RTCWREN = 1. A write to these registers, while RTCWREN = 0, will be ignored.

18.2 Operation

18.2.1 REGISTER INTERFACE

The register interface for the RTCC and alarm values is implemented using the Binary Coded Decimal (BCD) format. This simplifies the firmware when using the module, as each of the digits is contained within its own 4-bit value (see Figure 18-2 and Figure 18-3).

FIGURE 18-2: TIMER DIGIT FORMAT

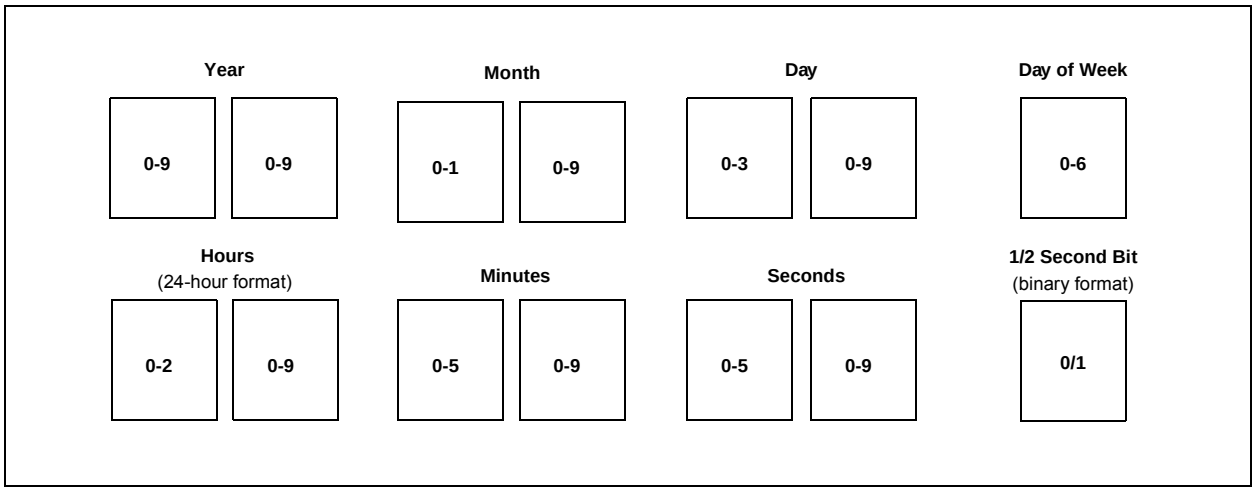
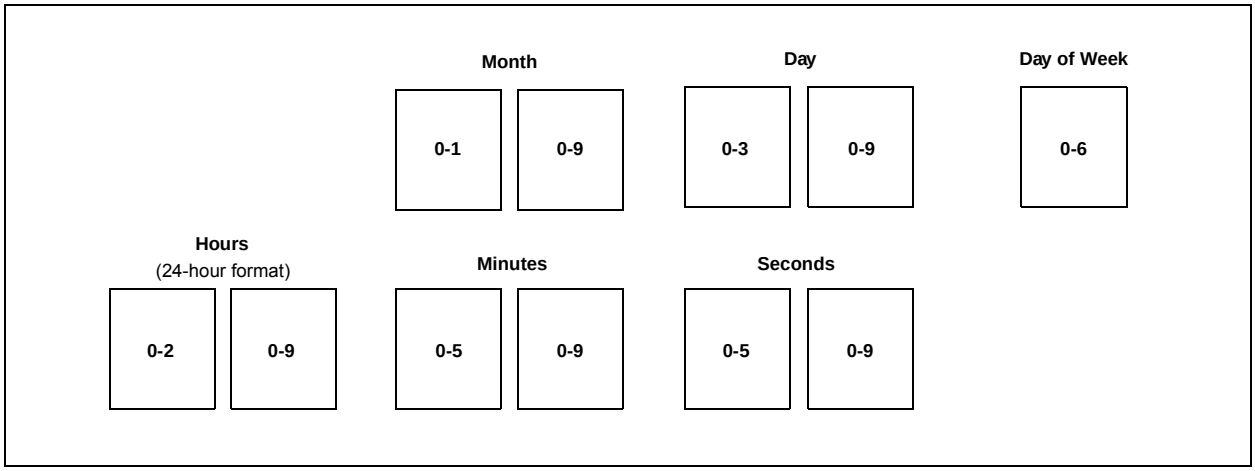


FIGURE 18-3: ALARM DIGIT FORMAT



19.3 Compare Mode

In Compare mode, the 16-bit CCPR4 register value is constantly compared against the Timer register pair value selected in the CCPTMR1 register. When a match occurs, the CCP4 pin can be:

- Driven high
- Driven low
- Toggled (high-to-low or low-to-high)
- Unchanged (that is, reflecting the state of the I/O latch)

The action on the pin is based on the value of the mode select bits (CCP4M<3:0>). At the same time, the interrupt flag bit, CCP4IF, is set.

Figure 19-2 gives the Compare mode block diagram

19.3.1 CCP PIN CONFIGURATION

The user must configure the CCPx pin as an output by clearing the appropriate TRIS bit.

Note:	Clearing the CCP4CON register will force the RC1 or RE7 compare output latch (depending on device configuration) to the default low level. This is not the PORTC or PORTE I/O data latch.

19.3.2 TIMER1/3/5/7 MODE SELECTION

If the CCP module is using the compare feature in conjunction with any of the Timer1/3/5/7 timers, the timers must be running in Timer mode or Synchronized Counter mode. In Asynchronous Counter mode, the compare operation may not work.

Note:	Details of the timer assignments for the CCP modules are given in Table 19-2 and Table 19-3.

19.3.3 SOFTWARE INTERRUPT MODE

When the Generate Software Interrupt mode is chosen (CCP4M<3:0> = 1010), the CCP4 pin is not affected. Only a CCP interrupt is generated, if enabled, and the CCP4IE bit is set.

19.3.4 SPECIAL EVENT TRIGGER

Both CCP modules are equipped with a Special Event Trigger. This is an internal hardware signal generated in Compare mode to trigger actions by other modules. The Special Event Trigger is enabled by selecting the Compare Special Event Trigger mode (CCP4M<3:0> = 1011).

For either CCP module, the Special Event Trigger resets the Timer register pair for whichever timer resource is currently assigned as the module's time base. This allows the CCPRx registers to serve as a programmable Period register for either timer.

The Special Event Trigger for CCP4 cannot start an A/D conversion.

Note:	The Special Event Trigger of ECCP2 can start an A/D conversion, but the A/D Converter must be enabled. For more information, see **Section 19.0 "Capture/Compare/PWM (CCP) Modules"**.

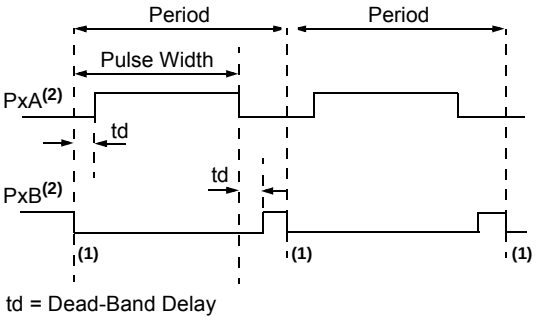
20.4.1 HALF-BRIDGE MODE

In Half-Bridge mode, two pins are used as outputs to drive push-pull loads. The PWM output signal is output on the PxA pin, while the complementary PWM output signal is output on the PxB pin (see Figure 20-6). This mode can be used for half-bridge applications, as shown in Figure 20-7, or for full-bridge applications, where four power switches are being modulated with two PWM signals.

In Half-Bridge mode, the programmable dead-band delay can be used to prevent shoot-through current in half-bridge power devices. The value of the PxDC<6:0> bits of the ECCPxDEL register sets the number of instruction cycles before the output is driven active. If the value is greater than the duty cycle, the corresponding output remains inactive during the entire cycle. For more details on the dead-band delay operations, see **Section 20.4.6 "Programmable Dead-Band Delay Mode"**.

Since the PxA and PxB outputs are multiplexed with the port data latches, the associated TRIS bits must be cleared to configure PxA and PxB as outputs.

FIGURE 20-6: EXAMPLE OF HALF-BRIDGE PWM OUTPUT



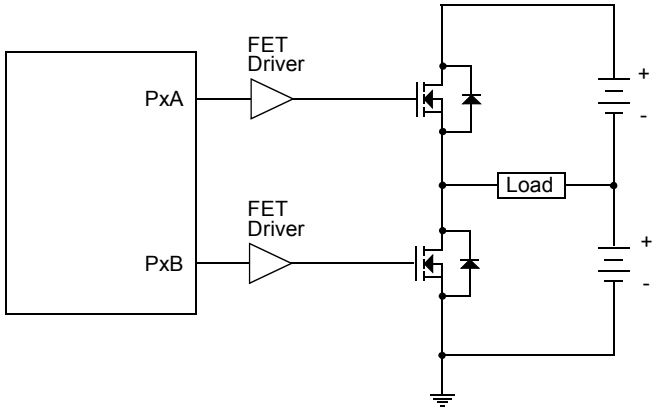
td = Dead-Band Delay

Note 1: At this time, the TMR2 register is equal to the PR2 register.

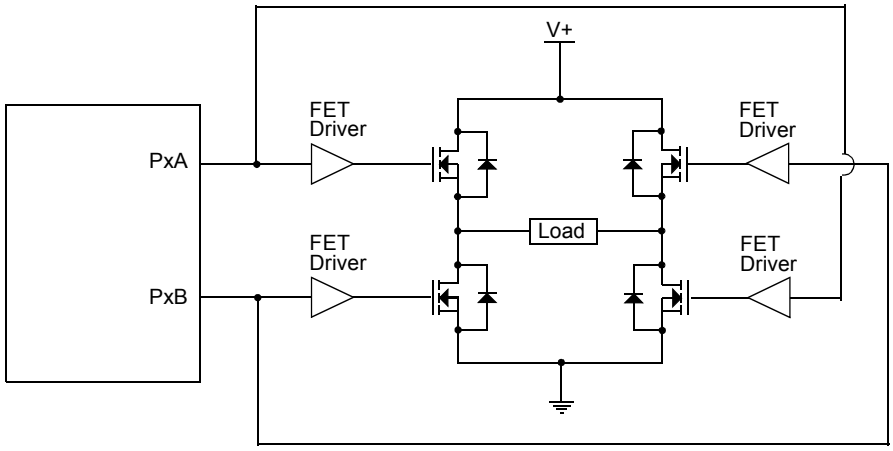
2: Output signals are shown as active-high.

FIGURE 20-7: EXAMPLE OF HALF-BRIDGE APPLICATIONS

Standard Half-Bridge Circuit ("Push-Pull")



Half-Bridge Output Driving a Full-Bridge Circuit



21.3.2 OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPxCON1<5:0> and SSPxSTAT<7:6>). These control bits allow the following to be specified:

- Master mode (SCKx is the clock output)
- Slave mode (SCKx is the clock input)
- Clock Polarity (Idle state of SCKx)
- Data Input Sample Phase (middle or end of data output time)
- Clock Edge (output data on rising/falling edge of SCKx)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

Each MSSP module consists of a Transmit/Receive Shift register (SSPxSR) and a Serial Input Buffer register (SSPxBUF). The SSPxSR shifts the data in and out of the device, MSb first. The SSPxBUF holds the data that was written to the SSPxSR until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPxBUF register. Then, the Buffer Full detect bit, BF (SSPxSTAT<0>), and the interrupt flag bit, SSPxIF, are set. This double-buffering of the received data (SSPxBUF) allows the next byte to start reception before reading the data that was just received. Any write to the SSPxBUF register during transmission/reception of data will be ignored and the Write Collision Detect bit, WCOL (SSPxCON1<7>), will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPxBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPxBUF should be read before the next byte of data to transfer is written to the SSPxBUF. The Buffer Full bit, BF (SSPxSTAT<0>), indicates when SSPxBUF has been loaded with the received data (transmission is complete). When the SSPxBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the MSSP interrupt is used to determine when the transmission/reception has completed. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur. Example 21-1 shows the loading of the SSPxBUF (SSPxSR) for data transmission.

The SSPxSR is not directly readable or writable and can only be accessed by addressing the SSPxBUF register. Additionally, the SSPxSTAT register indicates the various status conditions.

21.3.3 OPEN-DRAIN OUTPUT OPTION

The drivers for the SDOx output and SCKx clock pins can be optionally configured as open-drain outputs. This feature allows the voltage level on the pin to be pulled to a higher level through an external pull-up resistor, and allows the output to communicate with external circuits without the need for additional level shifters. For more information, see **Section 12.1.3 "Open-Drain Outputs"**.

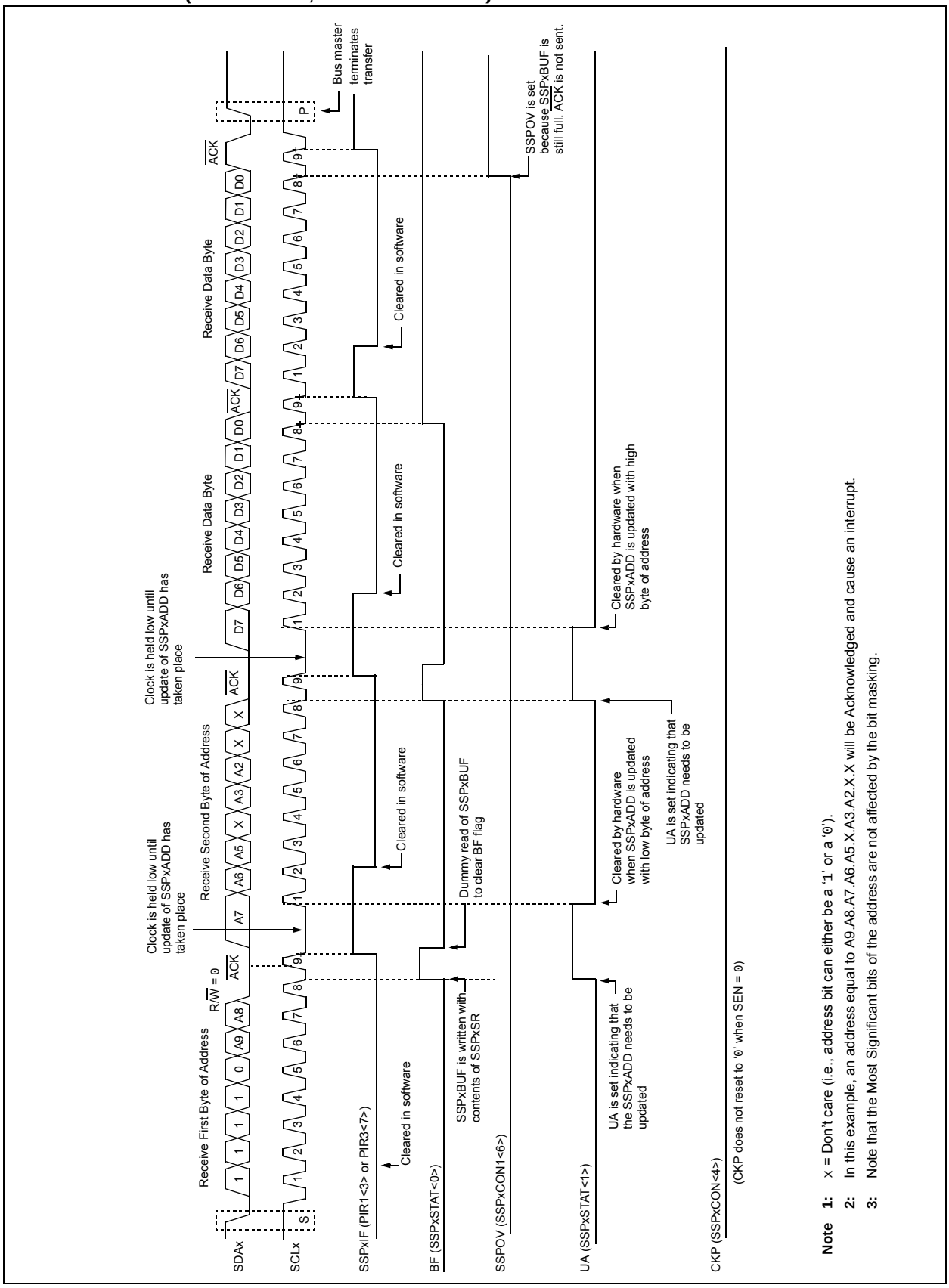
The open-drain output option is controlled by the SSP2OD and SSP1OD bits (ODCON3<1:0>). Setting an SSPxOD bit configures the SDOx and SCKx pins for the corresponding module for open-drain operation.

Note:	To avoid lost data in Master mode, a read of the SSPxBUF must be performed to clear the Buffer Full (BF) detect bit (SSPxSTAT<0>) between each transmission.

EXAMPLE 21-1: LOADING THE SSP1BUF (SSP1SR) REGISTER

```
LOOP    BTFSS   SSP1STAT, BF    ;Has data been received (transmit complete)?
        BRA     LOOP            ;No
        MOVF    SSP1BUF, W      ;WREG reg = contents of SSP1BUF
        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful
        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSP1BUF         ;New data to xmit
```

FIGURE 21-11: I²C™ SLAVE MODE TIMING WITH SEN = 0 AND ADMSK<5:1> = 01001 (RECEPTION, 10-BIT ADDRESS)



Note 1: x = Don't care (i.e., address bit can either be a '1' or a '0').

2: In this example, an address equal to A9.A8.A7.A6.A5.X.A3.A2.X.X will be Acknowledged and cause an interrupt.

3: Note that the Most Significant bits of the address are not affected by the bit masking.

22.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module is one of two serial I/O modules. (Generically, the EUSART is also known as a Serial Communications Interface or SCI.) The EUSART can be configured as a full-duplex, asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers. It can also be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The Enhanced USART module implements additional features, including automatic baud rate detection and calibration, automatic wake-up on Sync Break reception and 12-bit Break character transmit. These make it ideally suited for use in Local Interconnect Network bus (LIN/J2602 bus) systems.

All members of the PIC18F87K22 family are equipped with two independent EUSART modules, referred to as EUSART1 and EUSART2. They can be configured in the following modes:

- Asynchronous (full duplex) with:
  - Auto-wake-up on character reception
  - Auto-baud calibration
  - 12-bit Break character transmission
- Synchronous – Master (half duplex) with selectable clock polarity
- Synchronous – Slave (half duplex) with selectable clock polarity

The pins of EUSART1 and EUSART2 are multiplexed with the functions of PORTC (RC6/TX1/CK1 and RC7/RX1/DT1) and PORTG (RG1/TX2/CK2/AN19/C3OUT and RG2/RX2/DT2/AN18/C3INA), respectively. In order to configure these pins as an EUSART:

- For EUSART1:
  - Bit, SPEN (RCSTA1<7>), must be set (= 1)
  - Bit, TRISC<7>, must be set (= 1)
  - Bit, TRISC<6>, must be cleared (= 0) for Asynchronous and Synchronous Master modes
  - Bit, TRISC<6>, must be set (= 1) for Synchronous Slave mode
- For EUSART2:
  - Bit, SPEN (RCSTA2<7>), must be set (= 1)
  - Bit, TRISG<2>, must be set (= 1)
  - Bit TRISG<1> must be cleared (= 0) for Asynchronous and Synchronous Master modes
  - Bit, TRISC<6>, must be set (= 1) for Synchronous Slave mode

Note:	The EUSART control will automatically reconfigure the pin from input to output as needed.

The operation of each Enhanced USART module is controlled through three registers:

- Transmit Status and Control (TXSTAx)
- Receive Status and Control (RCSTAx)
- Baud Rate Control (BAUDCONx)

These are detailed on the following pages in Register 22-1, Register 22-2 and Register 22-3, respectively.

Note:	Throughout this section, references to register and bit names that may be associated with a specific EUSART module are referred to generically by the use of ‘x’ in place of the specific module number. Thus, “RCSTAx” might refer to the Receive Status register for either EUSART1 or EUSART2.

22.1.3 AUTO-BAUD RATE DETECT

The Enhanced USART module supports the automatic detection and calibration of baud rate. This feature is active only in Asynchronous mode and while the WUE bit is clear.

The automatic baud rate measurement sequence (Figure 22-1) begins whenever a Start bit is received and the ABDEN bit is set. The calculation is self-averaging.

In the Auto-Baud Rate Detect (ABD) mode, the clock to the BRG is reversed. Rather than the BRG clocking the incoming RXx signal, the RXx signal is timing the BRG. In ABD mode, the internal Baud Rate Generator is used as a counter to time the bit period of the incoming serial byte stream.

Once the ABDEN bit is set, the state machine will clear the BRG and look for a Start bit. The Auto-Baud Rate Detect must receive a byte with the value, 55h (ASCII “U”, which is also the LIN/J2602 bus Sync character), in order to calculate the proper bit rate. The measurement is taken over both a low and a high bit time in order to minimize any effects caused by asymmetry of the incoming signal. After a Start bit, the SPBRGx begins counting up, using the preselected clock source on the first rising edge of RXx. After eight bits on the RXx pin or the fifth rising edge, an accumulated value totalling the proper BRG period is left in the SPBRGHx:SPBRGx register pair. Once the 5th edge is seen (this should correspond to the Stop bit), the ABDEN bit is automatically cleared.

If a rollover of the BRG occurs (an overflow from FFFFh to 0000h), the event is trapped by the ABDOVF status bit (BAUDCONx<7>). It is set in hardware by BRG rollovers and can be set or cleared by the user in software. ABD mode remains active after rollover events and the ABDEN bit remains set (Figure 22-2).

While calibrating the baud rate period, the BRG registers are clocked at 1/8th the preconfigured clock rate. The BRG clock will be configured by the BRG16 and BRGH bits. The BRG16 bit must be set to use both SPBRG1 and SPBRGH1 as a 16-bit counter. This allows the user to verify that no carry occurred for 8-bit modes by checking for 00h in the SPBRGHx register. Refer to Table 22-4 for counter clock rates to the BRG.

While the ABD sequence takes place, the EUSART state machine is held in Idle. The RCxIF interrupt is set once the fifth rising edge on RXx is detected. The value in the RCREGx needs to be read to clear the RCxIF interrupt. The contents of RCREGx should be discarded.

Note 1: If the WUE bit is set with the ABDEN bit, Auto-Baud Rate Detection will occur on the byte *following* the Break character.

2: It is up to the user to determine that the incoming character baud rate is within the range of the selected BRG clock source. Some combinations of oscillator frequency and EUSART baud rates are not possible due to bit error rates. Overall system timing and communication baud rates must be taken into consideration when using the Auto-Baud Rate Detection feature.

3: To maximize baud rate range, if that feature is used, it is recommended that the BRG16 bit (BAUDCONx<3>) be set.

TABLE 22-4: BRG COUNTER CLOCK RATES

BRG16	BRGH	BRG Counter Clock
0	0	FOSC/512
0	1	FOSC/128
1	0	FOSC/128
1	1	FOSC/32

22.1.3.1 ABD and EUSART Transmission

Since the BRG clock is reversed during ABD acquisition, the EUSART transmitter cannot be used during ABD. This means that whenever the ABDEN bit is set, TXREGx cannot be written to. Users should also ensure that ABDEN does not become set during a transmit sequence. Failing to do this may result in unpredictable EUSART operation.

24.5 Comparator Control and Configuration

Each comparator has up to eight possible combinations of inputs: up to four external analog inputs and one of two Internal Reference Voltages.

All of the comparators allow a selection of the signal from pin, CxINA, or the voltage from the comparator reference (CV_{REF}) on the non-inverting channel. This is compared to either CxINB, CxINC, C2INB/C2IND or the microcontroller's fixed Internal Reference Voltage (V_{BG}, 1.024V nominal) on the inverting channel. The comparator inputs and outputs are tied to fixed I/O pins, defined in Table 24-1. The available comparator configurations and their corresponding bit settings are shown in Figure 24-4.

TABLE 24-1: COMPARATOR INPUTS AND OUTPUTS

Comparator	Input or Output	I/O Pin
1	C1INA (V_{IN}+)	RF6
	C1INB (V_{IN}-)	RF5
	C1INC(1) (V_{IN}-)	RH6
	C2INB (V_{IN}-)	RF3
	C1OUT	RF2
2	C2INA (V_{IN}+)	RF4
	C2INB (V_{IN}-)	RF3
	C2INC(1) (V_{IN}-)	RH4
	C2IND(1) (V_{IN}-)	RH5
	C2OUT	RF1
3	C3INA (V_{IN}+)	RG2
	C3INB (V_{IN}-)	RG3
	C3INC (V_{IN}-)	RG4
	C2INB (V_{IN}-)	RF3
	C3OUT	RG1

Note 1: C1INC, C2INC and C2IND are all unavailable for 64-pin devices (PIC18F6XK22).

24.5.1 COMPARATOR ENABLE AND INPUT SELECTION

Setting the CON bit of the CMxCON register (CMxCON<7>) enables the comparator for operation. Clearing the CON bit disables the comparator, resulting in minimum current consumption.

The CCH<1:0> bits in the CMxCON register (CMxCON<1:0>) direct either one of three analog input pins, or the Internal Reference Voltage (VBG), to the comparator, V_{IN}-. Depending on the Comparator operating mode, either an external or Internal Reference Voltage may be used. For external analog pins that are unavailable in 64-pin devices (C1INC, C2INC and C2IND), the corresponding configurations that use them as inputs are unavailable.

The analog signal present at V_{IN}- is compared to the signal at V_{IN}+ and the digital output of the comparator is adjusted accordingly.

The external reference is used when CREF = 0 (CMxCON<2>) and V_{IN}+ is connected to the CxINA pin. When external reference voltages are used, the comparator module can be configured to have the reference sources externally. The reference signal must be between V_{SS} and V_{DD}, and can be applied to either pin of the comparator.

The comparator module also allows the selection of an internally generated reference voltage from the Comparator Voltage Reference (CV_{REF}) module. This module is described in more detail in **Section 25.0 "Comparator Voltage Reference Module"**. The reference from the comparator reference voltage module is only available when CREF = 1. In this mode, the Internal Reference Voltage is applied to the comparator's V_{IN}+ pin.

Note:	The comparator input pin, selected by CCH<1:0>, must be configured as an input by setting both the corresponding TRISF, TRISG or TRISH bit and the corresponding ANSELx bit in the ANCONx register.

24.5.2 COMPARATOR ENABLE AND OUTPUT SELECTION

The comparator outputs are read through the CMSTAT register. The CMSTAT<5> bit reads the Comparator 1 output, CMSTAT<6> reads Comparator 2 output and CMSTAT<7> reads Comparator 3 output. These bits are read-only.

The comparator outputs may also be directly output to the RF2, RF1 and RG1 I/O pins by setting the COE bit (CMxCON<6>). When enabled, multiplexers in the output path of the pins switch to the output of the comparator. While in this mode, the TRISF<2:1> and TRISG<1> bits still function as the digital output enable bits for the RF2, RF1 and RG1 pins.

By default, the comparator's output is at logic high whenever the voltage on V_{IN}+ is greater than on V_{IN}-. The polarity of the comparator outputs can be inverted using the CPOL bit (CMxCON<5>).

The uncertainty of each of the comparators is related to the input offset voltage and the response time given in the specifications, as discussed in **Section 24.2 "Comparator Operation"**.

REGISTER 28-11: CONFIG6H: CONFIGURATION REGISTER 6 HIGH (BYTE ADDRESS 30000Bh)

R/C-1	R/C-1	R-1	U-0	U-0	U-0	U-0	U-0
WRTD	WRTB	WRTC(1)	—	—	—	—	—
bit 7							bit 0

Legend:	C = Clearable bit		
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 **WRTD:** Data EEPROM Write Protection bit
1 = Data EEPROM is not write-protected
0 = Data EEPROM is write-protected

bit 6 **WRTB:** Boot Block Write Protection bit
1 = Boot block is not write-protected(2)
0 = Boot block is write-protected(2)

bit 5 **WRTC:** Configuration Register Write Protection bit(1)
1 = Configuration registers are not write-protected(2)
0 = Configuration registers are write-protected(2)

bit 4-0 **Unimplemented:** Read as '0'

Note 1: This bit is read-only in normal Execution mode; it can be written only in Program mode.
2: For the memory size of the blocks, see Figure 28-6.

BRA **Unconditional Branch**

Syntax: BRA n

Operands: $-1024 \le n \le 1023$

Operation: $(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1101	0nnn	nnnn	nnnn

Description: Add the 2's complement number '2n' 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 a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

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

Example: HERE BRA Jump

Before Instruction
PC = address (HERE)
After Instruction
PC = address (Jump)

BSF **Bit Set f**

Syntax: BSF f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: $1 \rightarrow f<b>$

Status Affected: None

Encoding:

1000	bbba	ffff	ffff

Description: Bit 'b' in register 'f' is set.

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 29.2.3 "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: BSF FLAG_REG, 7, 1

Before Instruction
FLAG_REG = 0Ah
After Instruction
FLAG_REG = 8Ah

DAW **Decimal Adjust W Register**

Syntax: DAW

Operands: None

Operation: If [W<3:0> > 9] or [DC = 1], then
(W<3:0>) + 6 → W<3:0>;
else
(W<3:0>) → W<3:0>

If [W<7:4> > 9] or [C = 1], then
(W<7:4>) + 6 → W<7:4>;
C = 1;
else
(W<7:4>) → W<7:4>

Status Affected: C

Encoding:

0000	0000	0000	0111

Description: DAW adjusts the 8-bit value in W, resulting from the earlier addition of two variables (each in packed BCD format) and produces a correct packed BCD result.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1: DAW

Before Instruction
W = A5h
C = 0
DC = 0
After Instruction
W = 05h
C = 1
DC = 0

Example 2:

Before Instruction
W = CEh
C = 0
DC = 0
After Instruction
W = 34h
C = 1
DC = 0

DECF **Decrement f**

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

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (f) – 1 → dest

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

Encoding:

0000	01da	ffff	ffff

Description: Decrement register, 'f'. 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 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 29.2.3 "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: DECF CNT, 1, 0

Before Instruction
CNT = 01h
Z = 0
After Instruction
CNT = 00h
Z = 1

ADDWF **ADD W to Indexed (Indexed Literal Offset mode)**

Syntax: ADDWF [k] {,d}

Operands: $0 \le k \le 95$
$d \in [0,1]$

Operation: (W) + ((FSR2) + k) → dest

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

Encoding:

0010	01d0	kkkk	kkkk

Description: The contents of W are added to the contents of the register indicated by FSR2, offset by the value 'k'.

If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: ADDWF [OFST],0

Before Instruction
W = 17h
OFST = 2Ch
FSR2 = 0A00h
Contents of 0A2Ch = 20h
After Instruction
W = 37h
Contents of 0A2Ch = 20h

BSF **Bit Set Indexed (Indexed Literal Offset mode)**

Syntax: BSF [k], b

Operands: $0 \le f \le 95$
$0 \le b \le 7$

Operation: 1 → ((FSR2) + k)<b>

Status Affected: None

Encoding:

1000	bbb0	kkkk	kkkk

Description: Bit 'b' of the register indicated by FSR2, offset by the value 'k', is set.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: BSF [FLAG_OFST], 7

Before Instruction
FLAG_OFST = 0Ah
FSR2 = 0A00h
Contents of 0A0Ah = 55h
After Instruction
Contents of 0A0Ah = D5h

SETF **Set Indexed (Indexed Literal Offset mode)**

Syntax: SETF [k]

Operands: $0 \le k \le 95$

Operation: FFh → ((FSR2) + k)

Status Affected: None

Encoding:

0110	1000	kkkk	kkkk

Description: The contents of the register indicated by FSR2, offset by 'k', are set to FFh.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: SETF [OFST]

Before Instruction
OFST = 2Ch
FSR2 = 0A00h
Contents of 0A2Ch = 00h
After Instruction
Contents of 0A2Ch = FFh