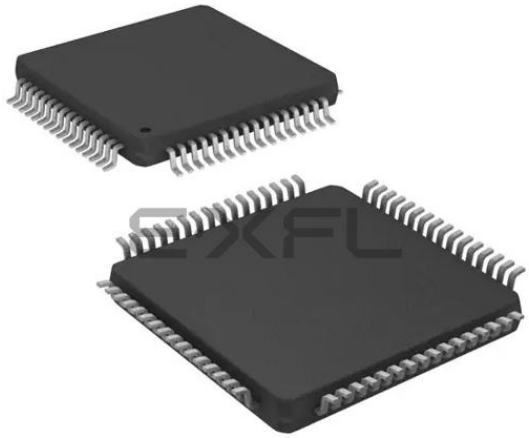


Welcome to E-XFL.COM

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	40MHz
Connectivity	CANbus, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	52
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.25K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 16x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf6680-i-pt

TABLE 1-2: PIC18F6585/8585/6680/8680 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PIC18F6X8X		PIC18F8X8X			
	TQFP	PLCC	TQFP			
						PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/INT0	48	60	58			
RB0				I/O	TTL	Digital I/O.
INT0				I	ST	External interrupt 0.
RB1/INT1	47	59	57			
RB1				I/O	TTL	Digital I/O.
INT1				I	ST	External interrupt 1.
RB2/INT2	46	58	56			
RB2				I/O	TTL	Digital I/O.
INT2				I	ST	External interrupt 2.
RB3/INT3/CCP2	45	57	55			
RB3				I/O	TTL	Digital I/O.
INT3				I/O	ST	External interrupt 3.
CCP2(1)				I/O	ST	Capture 2 input/Compare 2 output/ PWM 2 output.
RB4/KBI0	44	56	54			
RB4				I/O	TTL	Digital I/O.
KBI0				I	ST	Interrupt-on-change pin.
RB5/KBI1/PGM	43	55	53			
RB5				I/O	TTL	Digital I/O.
KBI1				I	ST	Interrupt-on-change pin.
PGM				I/O	ST	Low-Voltage ICSP Programming enable pin.
RB6/KBI2/PGC	42	54	52			
RB6				I/O	TTL	Digital I/O.
KBI2				I	ST	Interrupt-on-change pin.
PGC				I/O	ST	In-circuit debugger and ICSP programming clock.
RB7/KBI3/PGD	37	48	47			
RB7				I/O	TTL	Digital I/O.
KBI3				I/O	ST	Interrupt-on-change pin.
PGD						In-circuit debugger and ICSP programming data.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; Analog = Analog input; I = Input; O = Output; P = Power; OD = Open-Drain (no P diode to VDD)

Note 1: Alternate assignment for CCP2 in all operating modes except Microcontroller – applies to PIC18F8X8X only.

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X8X devices.

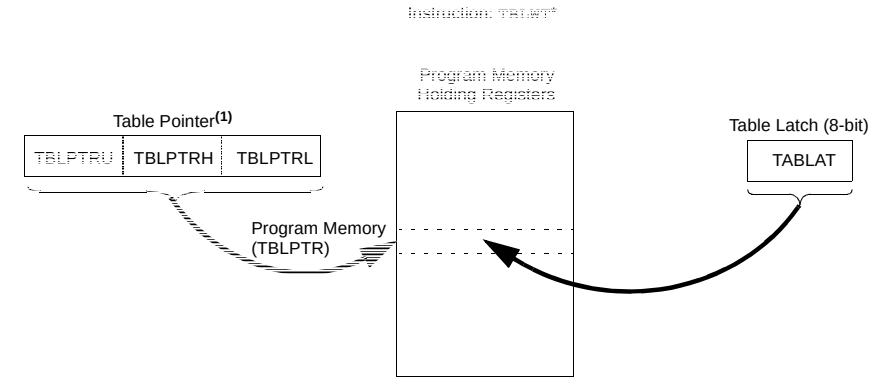
4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode; otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X8X (80-pin) devices.

6: PSP is available in Microcontroller mode only.

7: On PIC18F8X8X devices, these pins can be multiplexed with RH7/RH6 by changing the ECCPMX configuration bit.

FIGURE 5-2: TABLE WRITE OPERATION



Note 1: Table Pointer actually points to one of eight holding registers, the address of which is determined by TBLPTRL<2:0>. The process for physically writing data to the program memory array is discussed in Section 5.5 "Writing to Flash Program Memory".

5.2 Control Registers

Several control registers are used in conjunction with the TBLRD and TBLWT instructions. These include the:

- EECON1 register
- EECON2 register
- TABLAT register
- TBLPTR registers

5.2.1 EECON1 AND EECON2 REGISTERS

EECON1 is the control register for memory accesses.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the memory write and erase sequences.

Control bit EEPGD determines if the access will be a program or data EEPROM memory access. When clear, any subsequent operations will operate on the data EEPROM memory. When set, any subsequent operations will operate on the program memory.

Control bit CFGS determines if the access will be to the configuration/calibration registers or to program memory/data EEPROM memory. When set, subsequent operations will operate on configuration registers regardless of EEPGD (see **Section 24.0 "Special Features of the CPU"**). When clear, memory selection access is determined by EEPGD.

The FREE bit, when set, will allow a program memory erase operation. When the FREE bit is set, the erase operation is initiated on the next WR command. When FREE is clear, only writes are enabled.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set when a write operation is interrupted by a MCLR Reset or a WDT Time-out Reset during normal operation. In these situations, the user can check the WRERR bit and rewrite the location. It is necessary to reload the data and address registers (EEDATA and EEADR) due to Reset values of zero.

The WR control bit initiates write operations. The bit cannot be cleared, only set in software; it is cleared in hardware at the completion of the write operation. The inability to clear the WR bit in software prevents the accidental or premature termination of a write operation.

Note:	Interrupt flag bit, EEIF in the PIR2 register, is set when the write is complete. It must be cleared in software.

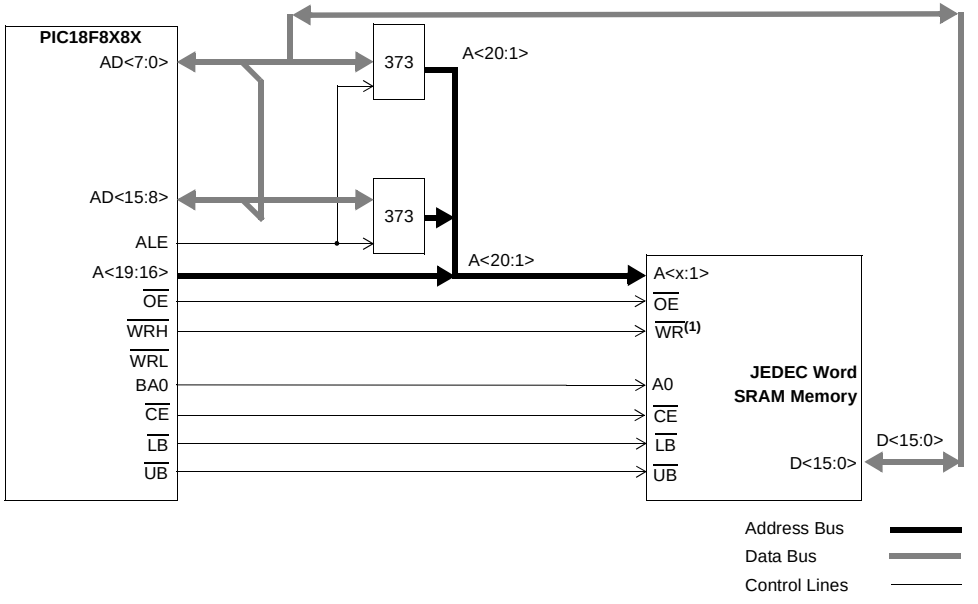
EXAMPLE 5-3: WRITING TO FLASH PROGRAM MEMORY (CONTINUED)

```
ERASE_BLOCK
                MOVLW     upper(CODE_ADDR)        ; load TBLPTR with the base
                MOVWF     TBLPTRU                 ; address of the memory block
                MOVLW     high(CODE_ADDR)
                MOVWF     TBLPTRH
                MOVLW     low(CODE_ADDR)
                MOVWF     TBLPTRL
                BSF       EECON1, EEPGD           ; point to Flash program memory
                BCF       EECON1, CFGS            ; access Flash program memory
                BSF       EECON1, WREN            ; enable write to memory
                BSF       EECON1, FREE            ; enable Row Erase operation
                BCF       INTCON, GIE             ; disable interrupts
                MOVLW     55h
                MOVWF     EECON2                  ; write 55H
   Required     MOVLW     0AAh
   Sequence     MOVWF     EECON2                  ; write AAH
                BSF       EECON1, WR              ; start erase (CPU stall)
                NOP
                BSF       INTCON, GIE             ; re-enable interrupts
                TBLRD*-                           ; dummy read decrement
WRITE_BUFFER_BACK
                MOVLW     8                       ; number of write buffer groups of 8 bytes
                MOVWF     COUNTER_HI
                MOVLW     high(BUFFER_ADDR)       ; point to buffer
                MOVWF     FSR0H
                MOVLW     low(BUFFER_ADDR)
                MOVWF     FSR0L
PROGRAM_LOOP
                MOVLW     8                       ; number of bytes in holding register
                MOVWF     COUNTER
WRITE_WORD_TO_HREGS
                MOVFW     POSTINC0, W             ; get low byte of buffer data
                MOVWF     TABLAT                  ; present data to table latch
                TBLWT+*                           ; write data, perform a short write
                                                  ; to internal TBLWT holding register.
                DECFSZ    COUNTER                 ; loop until buffers are full
                BRA       WRITE_WORD_TO_HREGS
PROGRAM_MEMORY
                BSF       EECON1, EEPGD           ; point to Flash program memory
                BCF       EECON1, CFGS            ; access Flash program memory
                BSF       EECON1, WREN            ; enable write to memory
                BCF       INTCON, GIE             ; disable interrupts
                MOVLW     55h
                MOVWF     EECON2                  ; write 55h
   Required     MOVLW     0AAh
   Sequence     MOVWF     EECON2                  ; write 0AAh
                BSF       EECON1, WR              ; start program (CPU stall)
                NOP
                BSF       INTCON, GIE             ; re-enable interrupts
                DECFSZ    COUNTER_HI              ; loop until done
                BRA       PROGRAM_LOOP
                BCF       EECON1, WREN            ; disable write to memory
```

6.2.3 16-BIT BYTE SELECT MODE

Figure 6-3 shows an example of 16-bit Byte Select mode for PIC18F8X8X devices.

FIGURE 6-3: 16-BIT BYTE SELECT MODE EXAMPLE



Note 1: This signal only applies to table writes. See **Section 5.1 "Table Reads and Table Writes"**.

10.7 PORTG, TRISG and LATG Registers

PORTG is a 6-bit wide port with 5 bidirectional pins and 1 unidirectional pin. The corresponding data direction register is TRISG. Setting a TRISG bit (= 1) will make the corresponding PORTG pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISG bit (= 0) will make the corresponding PORTG pin an output (i.e., put the contents of the output latch on the selected pin).

The Data Latch register (LATG) is also memory mapped. Read-modify-write operations on the LATG register read and write the latched output value for PORTG.

Pins RG0-RG2 on PORTG are multiplexed with the CAN peripheral. Refer to **Section 23.0 "ECAN Module"** for proper settings of TRISG when CAN is enabled. RG5 is multiplexed with MCLR/VPP. Refer to Register 24-5 for more information.

When enabling peripheral functions, care should be taken in defining TRIS bits for each PORTG pin. Some peripherals override the TRIS bit to make a pin an output, while other peripherals override the TRIS bit to make a pin an input. The user should refer to the corresponding peripheral section for the correct TRIS bit settings.

Note:	On a Power-on Reset, these pins are configured as digital inputs.

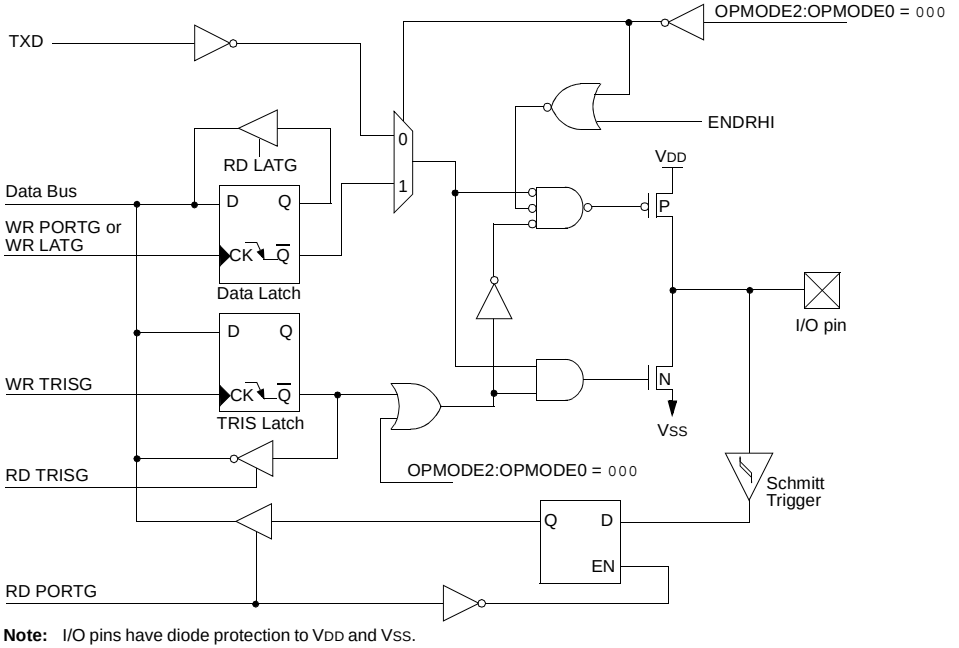
The pin override value is not loaded into the TRIS register. This allows read-modify-write of the TRIS register without concern due to peripheral overrides.

EXAMPLE 10-7: INITIALIZING PORT

```
CLRF   PORTG    ; Initialize PORTG by
                ; clearing output
                ; data latches
CLRF   LATG     ; Alternate method
                ; to clear output
                ; data latches
MOVLW  04h      ; Value used to
                ; initialize data
                ; direction
MOVWF  TRISG    ; Set RG1:RG0 as outputs
                ; RG2 as input
                ; RG4:RG3 as inputs
```

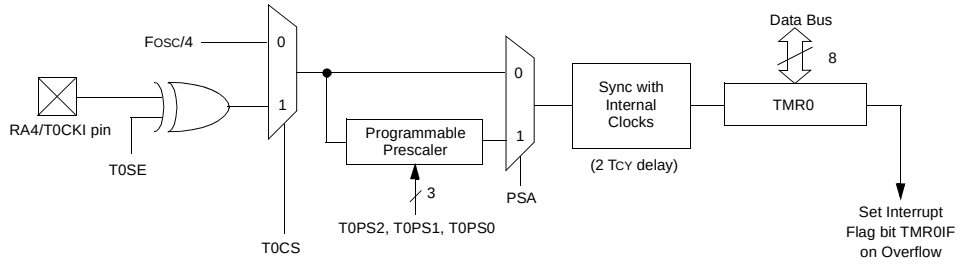
Note 1:	On a Power-on Reset, RG5 is enabled as a digital input only if Master Clear functionality is disabled (MCLRE = 0).
2:	If the device Master Clear is disabled, verify that either of the following is done to ensure proper entry into ICSP mode: a) disable Low-Voltage Programming (CONFIG4L<2> = 0); or b) make certain that RB5/KBI1/PGM is held low during entry into ICSP.

FIGURE 10-16: RG0/CANTX1 PIN BLOCK DIAGRAM



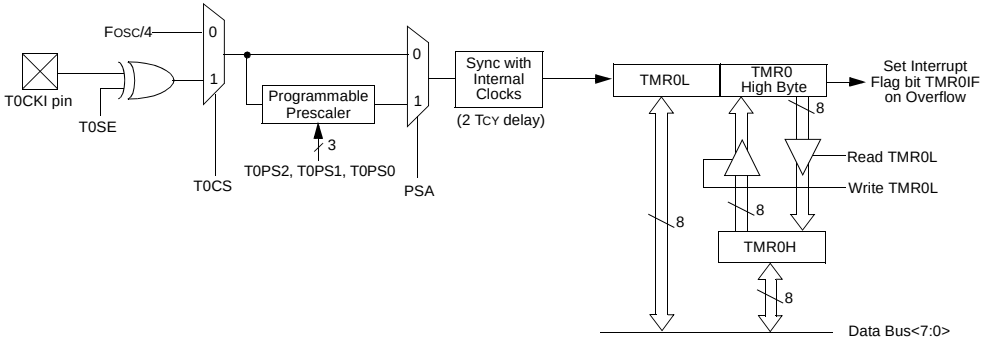
Note: I/O pins have diode protection to VDD and VSS.

FIGURE 11-1: TIMER0 BLOCK DIAGRAM IN 8-BIT MODE



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

FIGURE 11-2: TIMER0 BLOCK DIAGRAM IN 16-BIT MODE



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

17.3.6 SLAVE MODE

In Slave mode, the data is transmitted and received as the external clock pulses appear on SCK. When the last bit is latched, the SSPIF interrupt flag bit is set.

While in Slave mode, the external clock is supplied by the external clock source on the SCK pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in Sleep mode, the slave can transmit/receive data. When a byte is received, the device will wake-up from Sleep.

17.3.7 SLAVE SELECT SYNCHRONIZATION

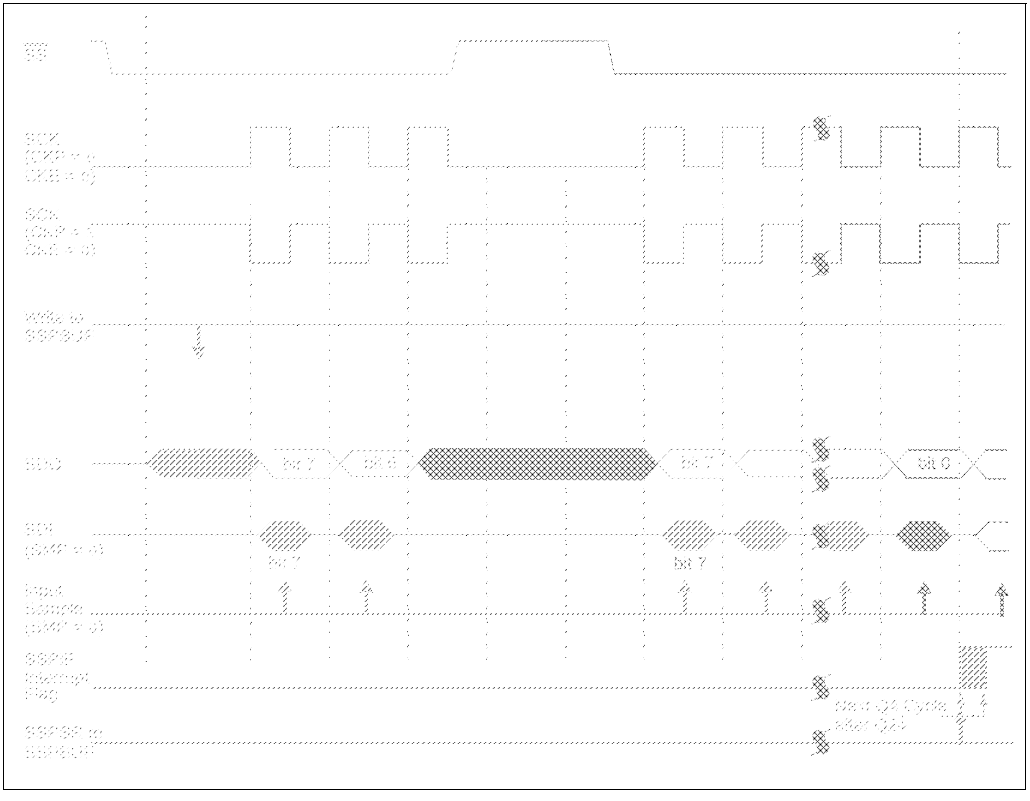
The $\overline{SS}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with $\overline{SS}$ pin control enabled (SSPCON1<3:0> = 04h). The pin must not be driven low for the $\overline{SS}$ pin to function as an input. The data latch must be high. When the $\overline{SS}$ pin is low, transmission and reception are enabled and the SDO pin is driven. When the $\overline{SS}$ pin goes high, the SDO pin is no longer driven even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

Note 1:	When the SPI is in Slave mode with $\overline{SS}$ pin control enabled (SSPCON<3:0> = 0100), the SPI module will reset if the $\overline{SS}$ pin is set to V_{DD}.
2:	If the SPI is used in Slave mode with CKE set, then the $\overline{SS}$ pin control must be enabled.

When the SPI module resets, the bit counter is forced to '0'. This can be done by either forcing the $\overline{SS}$ pin to a high level or clearing the SSPEN bit.

To emulate two-wire communication, the SDO pin can be connected to the SDI pin. When the SPI needs to operate as a receiver, the SDO pin can be configured as an input. This disables transmissions from the SDO. The SDI can always be left as an input (SDI function) since it cannot create a bus conflict.

FIGURE 17-4: SLAVE SYNCHRONIZATION WAVEFORM



18.2.4 AUTO-WAKE-UP ON SYNC BREAK CHARACTER

During Sleep mode, all clocks to the USART are suspended. Because of this, the Baud Rate Generator is inactive and a proper byte reception cannot be performed. The auto-wake-up feature allows the controller to wake-up due to activity on the RX/DT line while the USART is operating in Asynchronous mode.

The auto-wake-up feature is enabled by setting the WUE bit (BAUDCON<1>). Once set, the typical receive sequence on RX/DT is disabled and the USART remains in an Idle state monitoring for a wake-up event independent of the CPU mode. A wake-up event consists of a high-to-low transition on the RX/DT line. (This coincides with the start of a sync break or a wake-up signal character for the LIN protocol.)

Following a wake-up event, the module generates an RCIF interrupt. The interrupt is generated synchronously to the Q clocks in normal operating modes (Figure 18-7) and asynchronously, if the device is in Sleep mode (Figure 18-8). The interrupt condition is cleared by reading the RCREG register.

The WUE bit is automatically cleared once a low-to-high transition is observed on the RX line following the wake-up event. At this point, the USART module is in Idle mode and returns to normal operation. This signals to the user that the sync break event is over.

18.2.4.1 Special Considerations Using Auto-Wake-up

Since auto-wake-up functions by sensing rising edge transitions on RX/DT, information with any state changes before the Stop bit may signal a false end-of-character and cause data or framing errors. To work properly, therefore, the initial character in the transmission must be all '0's. This can be 00h (8 bytes) for standard RS-232 devices or 000h (12 bits) for LIN bus.

Oscillator start-up time must also be considered, especially in applications using oscillators with longer start-up intervals (i.e., XT or HS mode). The sync break (or wake-up signal) character must be of sufficient length and be followed by a sufficient interval to allow enough time for the selected oscillator to start and provide proper initialization of the USART.

18.2.4.2 Special Considerations Using the WUE Bit

The timing of WUE and RCIF events may cause some confusion when it comes to determining the validity of received data. As noted, setting the WUE bit places the USART in an Idle mode. The wake-up event causes a receive interrupt by setting the RCIF bit. The WUE bit is cleared after this when a rising edge is seen on RX/DT. The interrupt condition is then cleared by reading the RCREG register. Ordinarily, the data in RCREG will be dummy data and should be discarded.

The fact that the WUE bit has been cleared (or is still set) and the RCIF flag is set should not be used as an indicator of the integrity of the data in RCREG. Users should consider implementing a parallel method in firmware to verify received data integrity.

To assure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

FIGURE 18-7: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING NORMAL OPERATION

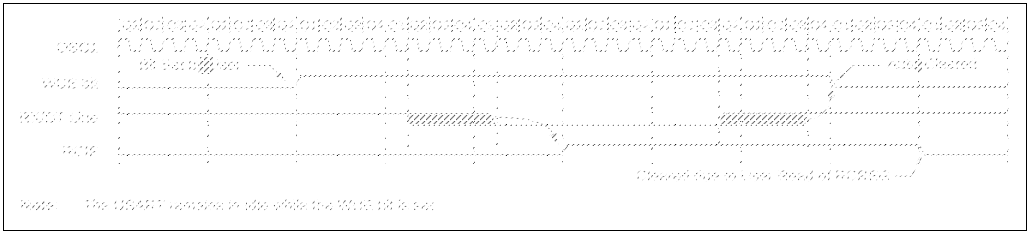
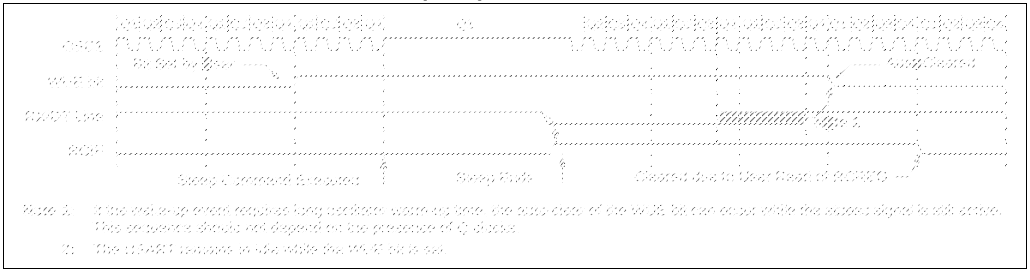


FIGURE 18-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



20.7 Comparator Operation During Sleep

When a comparator is active and the device is placed in Sleep mode, the comparator remains active and the interrupt is functional if enabled. This interrupt will wake-up the device from Sleep mode when enabled. While the comparator is powered up, higher Sleep currents than shown in the power-down current specification will occur. Each operational comparator will consume additional current as shown in the comparator specifications. To minimize power consumption while in Sleep mode, turn off the comparators (CM<2:0> = 111) before entering Sleep. If the device wakes up from Sleep, the contents of the CMCON register are not affected.

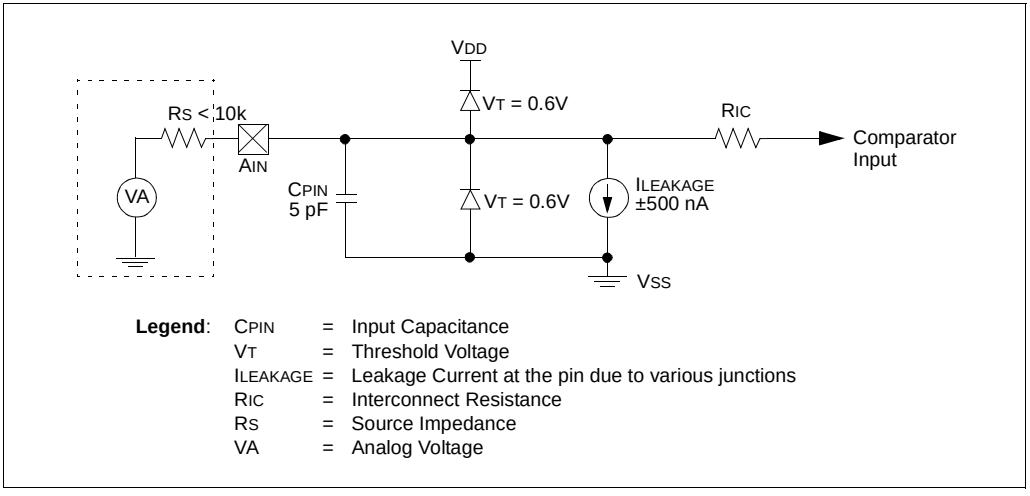
20.8 Effects of a Reset

A device Reset forces the CMCON register to its Reset state, causing the comparator module to be in the Comparator Reset mode (CM<2:0> = 000). This ensures that all potential inputs are analog inputs. Device current is minimized when analog inputs are present at Reset time. The comparators will be powered down during the Reset interval.

20.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 20-4. Since the analog pins are connected to a digital output, they have reverse biased diodes to VDD and VSS. The analog input, therefore, must be between VSS and VDD. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up condition may occur. A maximum source impedance of 10 kΩ is recommended for the analog sources. Any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current.

FIGURE 20-4: COMPARATOR ANALOG INPUT MODEL



Legend:

CPIN	=	Input Capacitance
VT	=	Threshold Voltage
ILEAKAGE	=	Leakage Current at the pin due to various junctions
RIC	=	Interconnect Resistance
RS	=	Source Impedance
VA	=	Analog Voltage

REGISTER 23-53: BRGCON2: BAUD RATE 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
SEG2PHTS	SAM	SEG1PH2	SEG1PH1	SEG1PH0	PRSEG2	PRSEG1	PRSEG0
bit 7							bit 0

bit 7 **SEG2PHTS:** Phase Segment 2 Time Select bit
1 = Freely programmable
0 = Maximum of PHEG1 or Information Processing Time (IPT), whichever is greater

bit 6 **SAM:** Sample of the CAN bus Line bit
1 = Bus line is sampled three times prior to the sample point
0 = Bus line is sampled once at the sample point

bit 5-3 **SEG1PH2:SEG1PH0:** Phase Segment 1 bits
111 = Phase Segment 1 time = 8 x TQ
110 = Phase Segment 1 time = 7 x TQ
101 = Phase Segment 1 time = 6 x TQ
100 = Phase Segment 1 time = 5 x TQ
011 = Phase Segment 1 time = 4 x TQ
010 = Phase Segment 1 time = 3 x TQ
001 = Phase Segment 1 time = 2 x TQ
000 = Phase Segment 1 time = 1 x TQ

bit 2-0 **PRSEG2:PRSEG0:** Propagation Time Select bits
111 = Propagation time = 8 x TQ
110 = Propagation time = 7 x TQ
101 = Propagation time = 6 x TQ
100 = Propagation time = 5 x TQ
011 = Propagation time = 4 x TQ
010 = Propagation time = 3 x TQ
001 = Propagation time = 2 x TQ
000 = Propagation time = 1 x TQ

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

23.9 Baud Rate Setting

All nodes on a given CAN bus must have the same nominal bit rate. The CAN protocol uses Non-Return-to-Zero (NRZ) coding which does not encode a clock within the data stream. Therefore, the receive clock must be recovered by the receiving nodes and synchronized to the transmitter's clock.

As oscillators and transmission time may vary from node to node, the receiver must have some type of Phase Lock Loop (PLL) synchronized to data transmission edges to synchronize and maintain the receiver clock. Since the data is NRZ coded, it is necessary to include bit stuffing to ensure that an edge occurs at least every six bit times to maintain the Digital Phase Lock Loop (DPLL) synchronization.

The bit timing of the PIC18F6585/8585/6680/8680 is implemented using a DPLL that is configured to synchronize to the incoming data and provides the nominal timing for the transmitted data. The DPLL breaks each bit time into multiple segments made up of minimal periods of time called the Time Quanta (T_Q).

Bus timing functions executed within the bit time frame, such as synchronization to the local oscillator, network transmission delay compensation, and sample point positioning, are defined by the programmable bit timing logic of the DPLL.

All devices on the CAN bus must use the same bit rate. However, all devices are not required to have the same master oscillator clock frequency. For the different clock frequencies of the individual devices, the bit rate has to be adjusted by appropriately setting the baud rate prescaler and number of time quanta in each segment.

The Nominal Bit Rate is the number of bits transmitted per second, assuming an ideal transmitter with an ideal oscillator, in the absence of resynchronization. The nominal bit rate is defined to be a maximum of 1 Mb/s.

The Nominal Bit Time is defined as:

EQUATION 23-1:

$$T_{BIT} = 1/\text{Nominal Bit Rate}$$

The Nominal Bit Time can be thought of as being divided into separate, non-overlapping time segments. These segments (Figure 23-4) include:

- Synchronization Segment (Sync_Seg)
- Propagation Time Segment (Prop_Seg)
- Phase Buffer Segment 1 (Phase_Seg1)
- Phase Buffer Segment 2 (Phase_Seg2)

The time segments (and thus the Nominal Bit Time) are in turn made up of integer units of time called Time Quanta or T_Q (see Figure 23-4). By definition, the Nominal Bit Time is programmable from a minimum of 8 T_Q to a maximum of 25 T_Q. Also by definition, the minimum Nominal Bit Time is 1 μs, corresponding to a maximum 1 Mb/s rate. The actual duration is given by the relationship:

EQUATION 23-2:

$$\text{Nominal Bit Time} = T_Q * (\text{Sync\_Seg} + \text{Prop\_Seg} + \text{Phase\_Seg1} + \text{Phase\_Seg2})$$

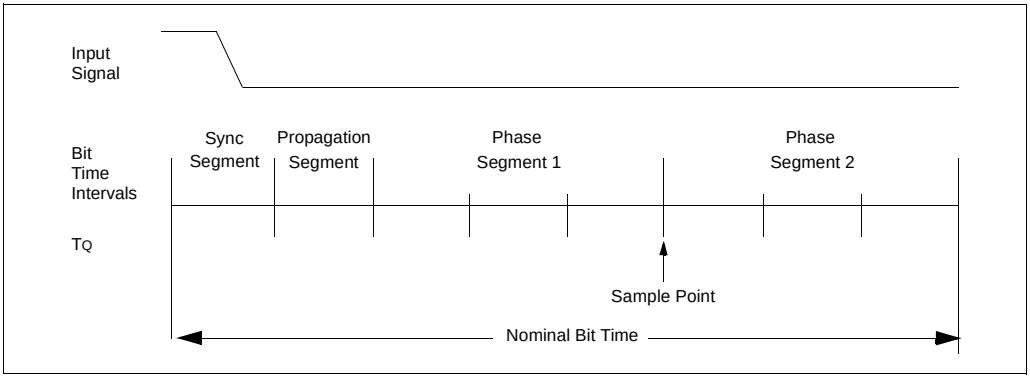
The Time Quantum is a fixed unit derived from the oscillator period. It is also defined by the programmable baud rate prescaler with integer values from 1 to 64 in addition to a fixed divide-by-two for clock generation. Mathematically, this is:

EQUATION 23-3:

$$T_Q\ (\mu s) = (2 * (BRP+1))/F_{OSC}\ (MHz)$$
or
$$T_Q\ (\mu s) = (2 * (BRP+1)) * T_{OSC}\ (\mu s)$$

where F_{OSC} is the clock frequency, T_{OSC} is the corresponding oscillator period, and BRP is an integer (0 through 63) represented by the binary values of BRGCON1<5:0>.

FIGURE 23-4: BIT TIME PARTITIONING



REGISTER 24-5: CONFIG3H: CONFIGURATION REGISTER 3 HIGH (BYTE ADDRESS 300005h)

R/P-1	U-0	U-0	U-0	U-0	U-0	R/P-1	R/P-1
MCLRE	—	—	—	—	—	ECCPMX	CCP2MX
bit 7							bit 0

bit 7 **MCLRE:** $\overline{\text{MCLR}}$ Enable bit(1)
1 = $\overline{\text{MCLR}}$ pin enabled, RG5 input pin disabled
0 = RG5 input enabled, $\overline{\text{MCLR}}$ disabled

bit 6-2 **Unimplemented:** Read as '0'

bit 1 **ECCPMX:** CCP1 PWM outputs P1B, P1C mux bit (PIC18F8X8X devices only)(2)
1 = P1B, P1C are multiplexed with RE6, RE5
0 = P1B, P1C are multiplexed with RH7, RH6

bit 0 **CCP2MX:** CCP2 Mux bit
In Microcontroller mode:
1 = CCP2 input/output is multiplexed with RC1
0 = CCP2 input/output is multiplexed with RE7
In Microprocessor, Microprocessor with Boot Block and Extended Microcontroller modes (PIC18F8X8X devices only):
1 = CCP2 input/output is multiplexed with RC1
0 = CCP2 input/output is multiplexed with RB3

Note 1: If $\overline{\text{MCLR}}$ is disabled, either disable low-voltage ICSP or hold RB5/PGM low to ensure proper entry into ICSP mode.
2: Reserved for PIC18F6X8X devices; maintain this bit set.

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-6: CONFIG4L: CONFIGURATION REGISTER 4 LOW (BYTE ADDRESS 300006h)

R/P-1	U-0	U-0	U-0	U-0	R/P-1	U-0	R/P-1
DEBUG	—	—	—	—	LVP	—	STVREN
bit 7							bit 0

bit 7 $\overline{\textbf{DEBUG}}$**:** Background Debugger Enable bit
1 = Background debugger disabled. RB6 and RB7 configured as general purpose I/O pins.
0 = Background debugger enabled. RB6 and RB7 are dedicated to in-circuit debug.

bit 6-3 **Unimplemented:** Read as '0'

bit 2 **LVP:** Low-Voltage ICSP Enable bit
1 = Low-voltage ICSP enabled
0 = Low-voltage ICSP disabled

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

bit 0 **STVREN:** Stack Full/Underflow Reset Enable bit
1 = Stack full/underflow will cause Reset
0 = Stack full/underflow will not cause Reset

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-13: DEVICE ID REGISTER 1 FOR PIC18FXX8X DEVICES (ADDRESS 3FFFFEh)

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							bit 0

bit 7-5 **DEV2:DEV0:** Device ID bits
000 = PIC18F8680
001 = PIC18F6680
010 = PIC18F8585
011 = PIC18F6585

bit 4-0 **REV4:REV0:** Revision ID bits
These bits are used to indicate the device revision.

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-14: DEVICE ID REGISTER 2 FOR PIC18FXX8X DEVICES (ADDRESS 3FFFFFh)

R-0	R-0	R-0	R-0	R-1	R-0	R-1	R-0
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
bit 7							bit 0

bit 7-0 **DEV10:DEV3:** Device ID bits
These bits are used with the DEV2:DEV0 bits in the Device ID Register 1 to identify the part number.
0000 1010 = PIC18F6585/8585/6680/8680

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
- n = Value when device is unprogrammed		u = Unchanged from programmed state

25.1 Instruction Set

ADDLW **ADD literal to W**

Syntax: [*label*] ADDLW k

Operands: $0 \le k \le 255$

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

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

Encoding:

0000	1111	kkkk	kkkk

Description: The contents of W are added to the 8-bit literal 'k' and 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: `ADDLW 0x15`

Before Instruction
W = 0x10

After Instruction
W = 0x25

ADDWF **ADD W to f**

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

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

Operation: $(W) + (f) \rightarrow$ dest

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

Encoding:

0010	01da	ffff	ffff

Description: Add W to register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'd' (default). If 'a' is '0', the Access Bank will be selected. If 'a' is '1', the BSR is used.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `ADDWF REG, 0, 0`

Before Instruction
W = 0x17
REG = 0xC2

After Instruction
W = 0xD9
REG = 0xC2

IORLW **Inclusive OR literal with W**

Syntax: [*label*] IORLW k

Operands: $0 \le k \le 255$

Operation: (W) .OR. k → W

Status Affected: N, Z

Encoding:

0000	1001	kkkk	kkkk

Description: The contents of W are OR'ed with the eight-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: `IORLW 0x35`

Before Instruction
W = 0x9A

After Instruction
W = 0xBF

IORWF **Inclusive OR W with f**

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

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

Operation: (W) .OR. (f) → dest

Status Affected: N, Z

Encoding:

0001	00da	ffff	ffff

Description: Inclusive OR W with register 'f'. 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, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `IORWF RESULT, 0, 1`

Before Instruction
RESULT = 0x13
W = 0x91

After Instruction
RESULT = 0x13
W = 0x93

POP **Pop Top of Return Stack**

Syntax: [*label*] POP

Operands: None

Operation: (TOS) → bit bucket

Status Affected: None

Encoding:

0000	0000	0000	0110

Description: The TOS value is pulled off the return stack and is discarded. The TOS value then becomes the previous value that was pushed onto the return stack.
This instruction is provided to enable the user to properly manage the return stack to incorporate a software stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	POP TOS value	No operation

Example:
```
POP
GOTO    NEW
```
Before Instruction
TOS = 0031A2h
Stack (1 level down) = 014332h

After Instruction
TOS = 014332h
PC = NEW

PUSH **Push Top of Return Stack**

Syntax: [*label*] PUSH

Operands: None

Operation: (PC+2) → TOS

Status Affected: None

Encoding:

0000	0000	0000	0101

Description: The PC+2 is pushed onto the top of the return stack. The previous TOS value is pushed down on the stack. This instruction allows implementing a software stack by modifying TOS, and then pushing it onto the return stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	PUSH PC+2 onto return stack	No operation	No operation

Example:
```
PUSH
```
Before Instruction
TOS = 00345Ah
PC = 000124h

After Instruction
PC = 000126h
TOS = 000126h
Stack (1 level down) = 00345Ah

TSTFSZ **Test f, skip if 0**

Syntax: [*label*] TSTFSZ f [,a]

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

Operation: skip if f = 0

Status Affected: None

Encoding:

0110	011a	ffff	ffff

Description: If 'f' = 0, the next instruction, fetched during the current instruction execution is discarded and a NOP is executed, making this a two-cycle instruction. 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 (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    TSTFSZ  CNT, 1
NZERO   :
ZERO    :
```

Before Instruction
PC = Address (HERE)

After Instruction
If CNT = 0x00,
PC = Address (ZERO)
If CNT ≠ 0x00,
PC = Address (NZERO)

XORLW **Exclusive OR literal with W**

Syntax: [*label*] XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → W

Status Affected: N, Z

Encoding:

0000	1010	kkkk	kkkk

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

Words: 1

Cycles: 1

Q Cycle Activity:

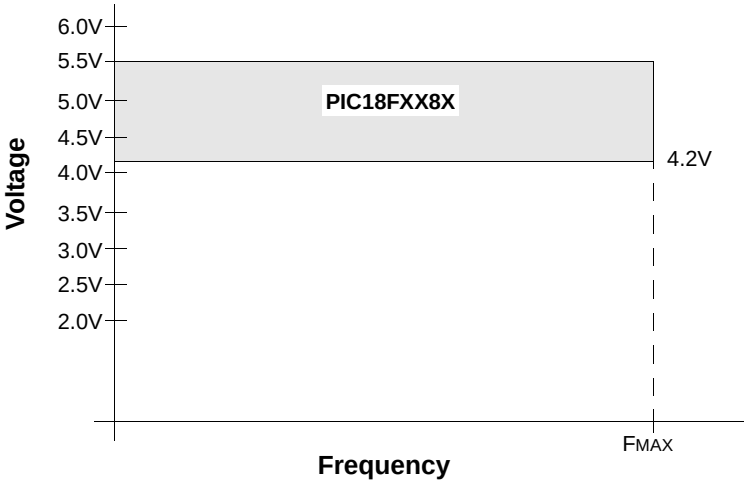
Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example:
```
XORLW 0xAF
```

Before Instruction
W = 0xB5

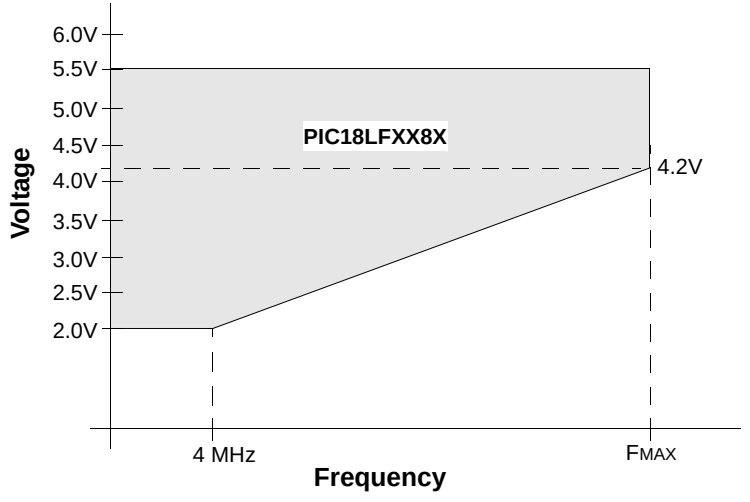
After Instruction
W = 0x1A

FIGURE 27-1: PIC18F6585/8585/6680/8680 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)



FMAX = 40 MHz for PIC18F6X8X and PIC18F8X8X in Microcontroller mode.
FMAX = 25 MHz for PIC18F8X8X in modes other than Microcontroller mode.

FIGURE 27-2: PIC18LF6585/8585/6680/8680 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)



For PIC18F6X8X and PIC18F8X8X in Microcontroller mode:
FMAX = (16.36 MHz/V) (VDDAPPMIN – 2.0V) + 4 MHz, if VDDAPPMIN ≤ 4.2V;
FMAX = 40 MHz, if VDDAPPMIN > 4.2V.

For PIC18F8X8X in modes other than Microcontroller mode:
FMAX = (9.55 MHz/V) (VDDAPPMIN – 2.0V) + 4 MHz, if VDDAPPMIN ≤ 4.2V;
FMAX = 25 MHz, if VDDAPPMIN > 4.2V.

Note: VDDAPPMIN is the minimum voltage of the PIC® device in the application.

28.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

Note: The graphs and tables provided following this note are a statistical summary based on a limited number of samples and are provided for informational purposes only. The performance characteristics listed herein are not tested or guaranteed. In some graphs or tables, the data presented may be outside the specified operating range (e.g., outside specified power supply range) and therefore, outside the warranted range.

"Typical" represents the mean of the distribution at 25°C. "Maximum" or "minimum" represents (mean + 3σ) or (mean – 3σ) respectively, where σ is a standard deviation, over the whole temperature range.

FIGURE 28-1: TYPICAL IDD vs. FOSC OVER VDD (HS MODE)

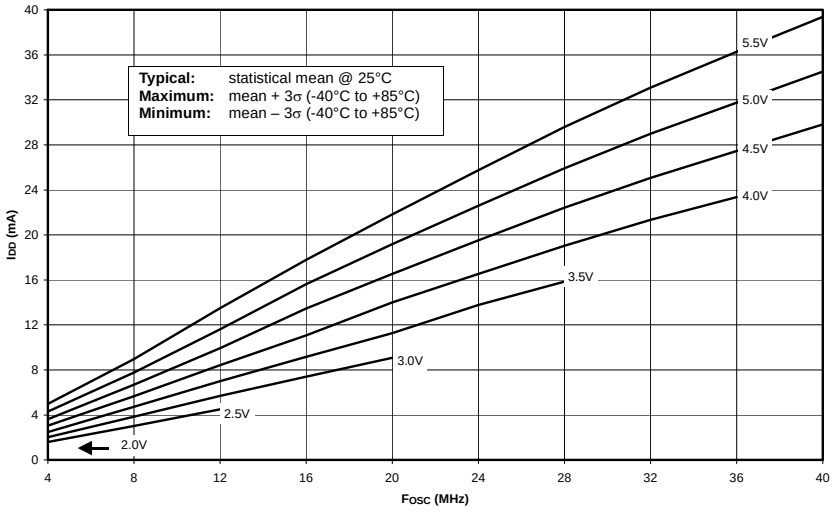


FIGURE 28-2: MAXIMUM IDD vs. FOSC OVER VDD (HS MODE)

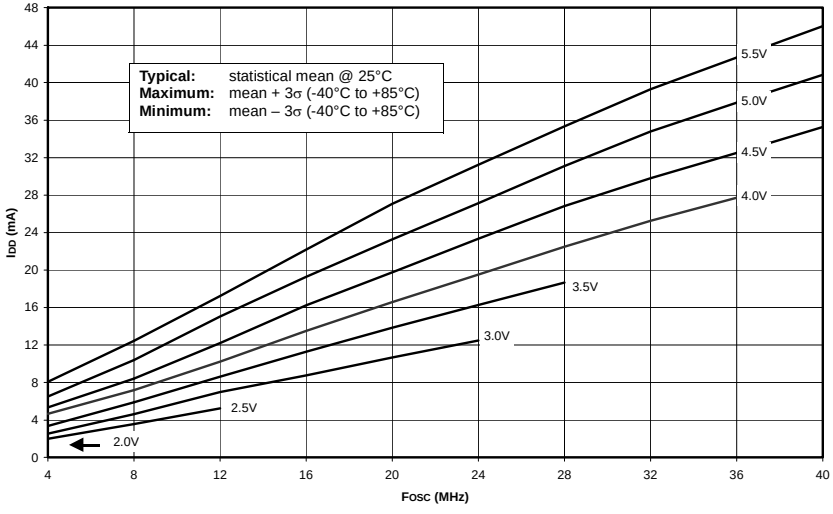


FIGURE 28-29: A/D NON-LINEARITY vs. VREFH (VDD = 5V, -40°C TO +125°C)