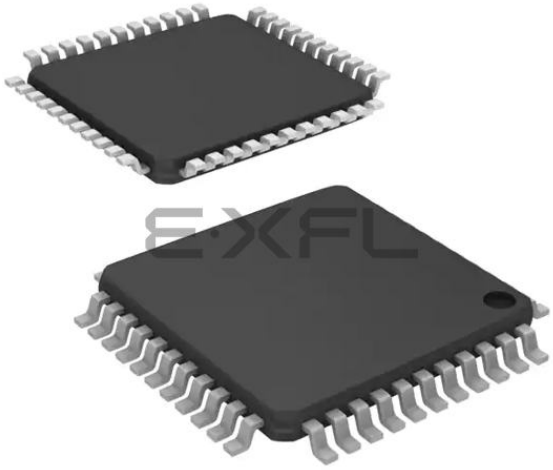


**Welcome to E-XFL.COM**

## 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 | 48MHz |
| Connectivity | I²C, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, HLVD, POR, PWM, WDT |
| Number of I/O | 35 |
| Program Memory Size | 32KB (16K x 16) |
| Program Memory Type | FLASH |
| EEPROM Size | 256 x 8 |
| RAM Size | 1.5K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.3V ~ 5.5V |
| Data Converters | A/D 30x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 125°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-TQFP |
| Supplier Device Package | 44-TQFP (10x10) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic18f45k22-e-pt |

**FIGURE 2-11: FSCM TIMING DIAGRAM**

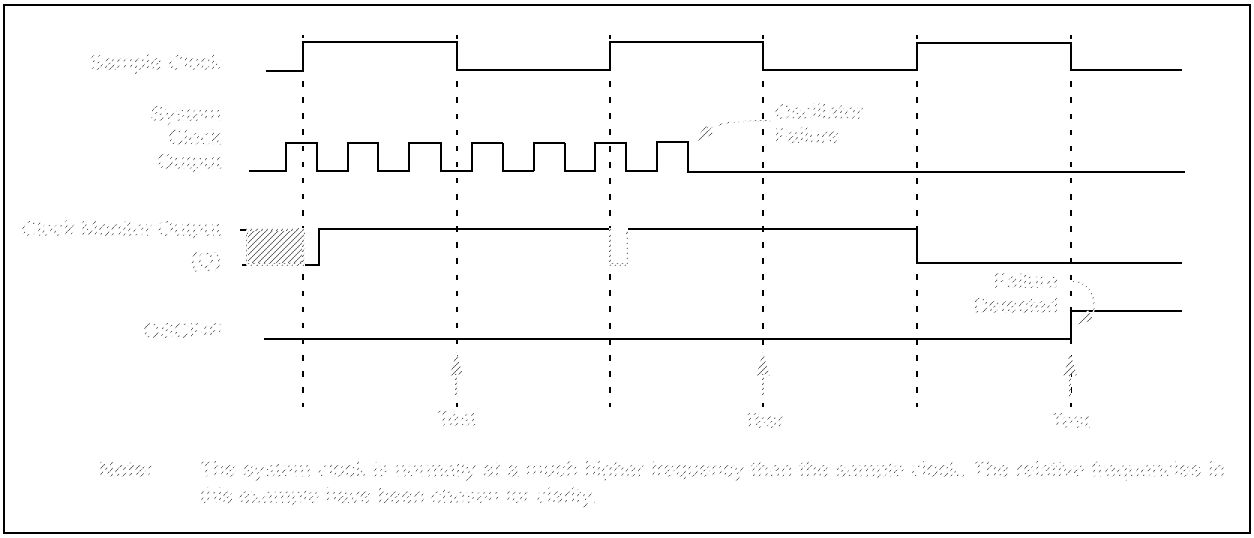



**TABLE 2-4: REGISTERS ASSOCIATED WITH CLOCK SOURCES**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 109 |
| IPR2 | OSCFIP | C1IP | C2IP | EEIP | BCL1IP | HLVDIP | TMR3IP | CCP2IP | 122 |
| OSCCON | IDLEN | IRCF<2:0> | | | OSTS | HFIOFS | SCS<1:0> | | 30 |
| OSCCON2 | PLLRDY | SOSCRUN | — | MFIOSEL | SOSCGO | PRISD | MFIOFS | LFIOFS | 31 |
| OSCTUNE | INTSRC | PLLEN | TUN<5:0> | | | | | | 35 |
| PIE2 | OSCFIE | C1IE | C2IE | EEIE | BCL1IE | HLVDIE | TMR3IE | CCP2IE | 118 |
| PIR2 | OSCFIF | C1IF | C2IF | EEIF | BCL1IF | HLVDIF | TMR3IF | CCP2IF | 113 |

**Legend:** — = unimplemented locations, read as '0'. Shaded bits are not used by clock sources.

**TABLE 2-5: CONFIGURATION REGISTERS ASSOCIATED WITH CLOCK SOURCES**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| CONFIG1H | IESO | FCMEN | PRICLKEN | PLLCFG | FOSC<3:0> | | | | 345 |
| CONFIG2L | — | — | — | BORV<1:0> | | BOREN<1:0> | | $\overline{\text{PWRTEN}}$ | 346 |
| CONFIG3H | MCLRE | — | P2BMX | T3CMX | HFOFST | CCP3MX | PBADEN | CCP2MX | 348 |

**Legend:** — = unimplemented locations, read as '0'. Shaded bits are not used for clock sources.

**REGISTER 9-16: IPR3: PERIPHERAL INTERRUPT PRIORITY REGISTER 3**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| SSP2IP | BCL2IP | RC2IP | TX2IP | CTMUIP | TMR5GIP | TMR3GIP | TMR1GIP |
| 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 **SSP2IP:** Synchronous Serial Port 2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **BCL2IP:** Bus Collision 2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **RC2IP:** EUSART2 Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TX2IP:** EUSART2 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **CTMUIP:** CTMU Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **TMR5GIP:** TMR5 Gate Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR3GIP:** TMR3 Gate Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR1GIP:** TMR1 Gate Interrupt Priority bit
1 = High priority
0 = Low priority

A mismatch condition will continue to set the RBIF flag bit. Reading or writing PORTB will end the mismatch condition and allow the RBIF bit to be cleared. The latch holding the last read value is not affected by a MCLR nor Brown-out Reset. After either one of these Resets, the RBIF flag will continue to be set if a mismatch is present.

| | |
|---|---|
| **Note:** | If a change on the I/O pin should occur when the read operation is being executed (start of the Q2 cycle), then the RBIF interrupt flag may not get set. Furthermore, since a read or write on a port affects all bits of that port, care must be taken when using multiple pins in Interrupt-on-change mode. Changes on one pin may not be seen while servicing changes on another pin. |

The interrupt-on-change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt-on-change feature. Polling of PORTB is not recommended while using the interrupt-on-change feature.

### 10.3.3 ALTERNATE FUNCTIONS

PORTB is multiplexed with several peripheral functions (Table 10-5). The pins have TTL input buffers. Some of these pin functions can be relocated to alternate pins using the Control fuse bits in CONFIG3H. RB5 is the default pin for P2B (28-pin devices). Clearing the P2BMX bit moves the pin function to RC0. RB5 is also the default pin for the CCP3/P3A peripheral pin. Clearing the CCP3MX bit moves the pin function to the RC6 pin (28-pin devices) or RE0 (40/44-pin devices).

Two other pin functions, T3CKI and CCP2/P2A, can be relocated from their default pins to PORTB pins by clearing the control fuses in CONFIG3H. Clearing T3CMX and CCP2MX moves the pin functions to RB5 and RB3, respectively.

**TABLE 10-5: PORTB I/O SUMMARY**

| Pin | Function | TRIS Setting | ANSEL Setting | Pin Type | Buffer Type | Description |
|---|---|---|---|---|---|---|
| RB0/INT0/CCP4/ FLT0/SRI/SS2/ AN12 | RB0 | 0 | 0 | O | DIG | LATB<0> data output; not affected by analog input. |
| | | 1 | 0 | I | TTL | PORTB<0> data input; disabled when analog input enabled. |
| | INT0 | 1 | 0 | I | ST | External interrupt 0. |
| | CCP4(3) | 0 | 0 | O | DIG | Compare 4 output/PWM 4 output. |
| | | 1 | 0 | I | ST | Capture 4 input. |
| | FLT0 | 1 | 0 | I | ST | PWM Fault input for ECCP auto-shutdown. |
| | SRI | 1 | 0 | I | ST | SR latch input. |
| | SS2(3) | 1 | 0 | I | TTL | SPI slave select input (MSSP2). |
| | AN12 | 1 | 1 | I | AN | Analog input 12. |
| RB1/INT1/P1C/ SCK2/SCL2/ C12IN3-/AN10 | RB1 | 0 | 0 | O | DIG | LATB<1> data output; not affected by analog input. |
| | | 1 | 0 | I | TTL | PORTB<1> data input; disabled when analog input enabled. |
| | INT1 | 1 | 0 | I | ST | External Interrupt 1. |
| | P1C(3) | 0 | 0 | O | DIG | Enhanced CCP1 PWM output 3. |
| | SCK2(3) | 0 | 0 | O | DIG | MSSP2 SPI Clock output. |
| | | 1 | 0 | I | ST | MSSP2 SPI Clock input. |
| | SCL2(3) | 0 | 0 | O | DIG | MSSP2 $I^2C$ Clock output. |
| | | 1 | 0 | I | $I^2C$ | MSSP2 $I^2C$ Clock input. |
| | C12IN3- | 1 | 1 | I | AN | Comparators C1 and C2 inverting input. |
| | AN10 | 1 | 1 | I | AN | Analog input 10. |

**Legend:** AN = Analog input or output; TTL = TTL compatible input; HV = High Voltage; OD = Open Drain; XTAL = Crystal; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; $I^2C$ = Schmitt Trigger input with $I^2C$.

**Note 1:** Default pin assignment for P2B, T3CKI, CCP3 and CCP2 when Configuration bits PB2MX, T3CMX, CCP3MX and CCP2MX are set.

**2:** Alternate pin assignment for P2B, T3CKI, CCP3 and CCP2 when Configuration bits PB2MX, T3CMX, CCP3MX and CCP2MX are clear.

**3:** Function on PORTD and PORTE for PIC18(L)F4XK22 devices.

## 10.7 Port Analog Control

Most port pins are multiplexed with analog functions such as the Analog-to-Digital Converter and comparators. When these I/O pins are to be used as analog inputs it is necessary to disable the digital input buffer to avoid excessive current caused by improper biasing of the digital input. Individual control of the digital input buffers on pins which share analog functions is provided by the ANSELA, ANSELB, ANSELC, ANSELD and ANSELE registers. Setting an ANSx bit high will disable the associated digital input buffer and cause all reads of that pin to return '0' while allowing analog functions of that pin to operate correctly.

The state of the ANSx bits has no affect on digital output functions. A pin with the associated TRISx bit clear and ANSx bit set will still operate as a digital output but the input mode will be analog. This can cause unexpected behavior when performing read-modify-write operations on the affected port.

All ANSEL register bits default to '1' upon POR and BOR, disabling digital inputs for their associated port pins. All TRIS register bits default to '1' upon POR or BOR, disabling digital outputs for their associated port pins. As a result, all port pins that have an ANSEL register will default to analog inputs upon POR or BOR.

## 10.8 Port Slew Rate Control

The output slew rate of each port is programmable to select either the standard transition rate or a reduced transition rate of approximately 0.1 times the standard to minimize EMI. The reduced transition time is the default slew rate for all ports.

## 10.9 Register Definitions – Port Control

**REGISTER 10-1: PORTX(1): PORTx REGISTER**

| R/W-u/x | R/W-u/x | R/W-u/x | R/W-u/x | R/W-u/x | R/W-u/x | R/W-u/x | R/W-u/x |
|---|---|---|---|---|---|---|---|
| Rx7 | Rx6 | Rx5 | Rx4 | Rx3 | Rx2 | Rx1 | Rx0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **Rx<7:0>:** PORTx I/O bit values(2)

**Note 1:** Register Description for PORTA, PORTB, PORTC and PORTD.

**2:** Writes to PORTx are written to corresponding LATx register. Reads from PORTx register is return of I/O pin values.

**REGISTER 10-2: PORTE: PORTE REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-u/x | R/W-u/x | R/W-u/x | R/W-u/x |
|---|---|---|---|---|---|---|---|
| — | — | — | — | RE3(1) | RE2(2), (3) | RE1(2), (3) | RE0(2), (3) |
| bit 7 | | | | | | | bit 0 |

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

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **RE3:** PORTE Input bit value(1)

bit 2-0 **RE<2:0>:** PORTE I/O bit values(2), (3)

**Note 1:** Port is available as input only when MCLRE = 0.

**2:** Writes to PORTx are written to corresponding LATx register. Reads from PORTx register is return of I/O pin values.

**3:** Available on PIC18(L)F4XK22 devices.

**REGISTER 10-3: ANSELA – PORTA ANALOG SELECT REGISTER**

| U-0 | U-0 | R/W-1 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | ANSA5 | — | ANSA3 | ANSA2 | ANSA1 | ANSA0 |
| 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 **Unimplemented:** Read as '0'

bit 5 **ANSA5:** RA5 Analog Select bit
1 = Digital input buffer disabled
0 = Digital input buffer enabled

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

bit 3-0 **ANSA<3:0>:** RA<3:0> Analog Select bit
1 = Digital input buffer disabled
0 = Digital input buffer enabled

**REGISTER 10-7: ANSELE – PORTE ANALOG SELECT REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | ANSE2(1) | ANSE1(1) | ANSE0(1) |
| 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 7-3 **Unimplemented:** Read as '0'

bit 2-0 **ANSE<2:0>:** RE<2:0> Analog Select bit(1)

1 = Digital input buffer disabled
0 = Digital input buffer enabled

**Note 1:** Available on PIC18(L)F4XK22 devices only.

**REGISTER 10-8: TRISx: PORTx TRI-STATE REGISTER(1)**

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| TRISx7 | TRISx6 | TRISx5 | TRISx4 | TRISx3 | TRISx2 | TRISx1 | TRISx0 |
| 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 7-0 **TRISx<7:0>:** PORTx Tri-State Control bit

1 = PORTx pin configured as an input (tri-stated)
0 = PORTx pin configured as an output

**Note 1:** Register description for TRISA, TRISB, TRISC and TRISD.

**REGISTER 10-9: TRISE: PORTE TRI-STATE REGISTER**

| R/W-1 | U-0 | U-0 | U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| WPUE3 | — | — | — | — | TRISE2(1) | TRISE1(1) | TRISE0(1) |
| 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 **WPUE3:** Weak Pull-up Register bits

1 = Pull-up enabled on PORT pin
0 = Pull-up disabled on PORT pin

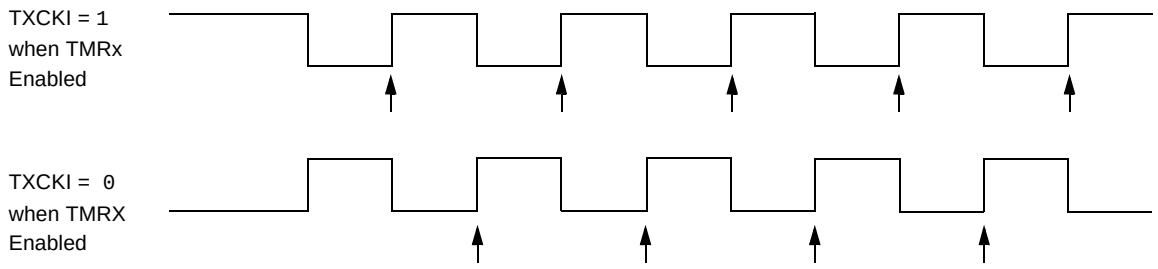
bit 6-3 **Unimplemented:** Read as '0'

bit 2-0 **TRISE<7:0>:** PORTE Tri-State Control bit(1)

1 = PORTE pin configured as an input (tri-stated)
0 = PORTE pin configured as an output

**Note 1:** Available on PIC18(L)F4XK22 devices only.

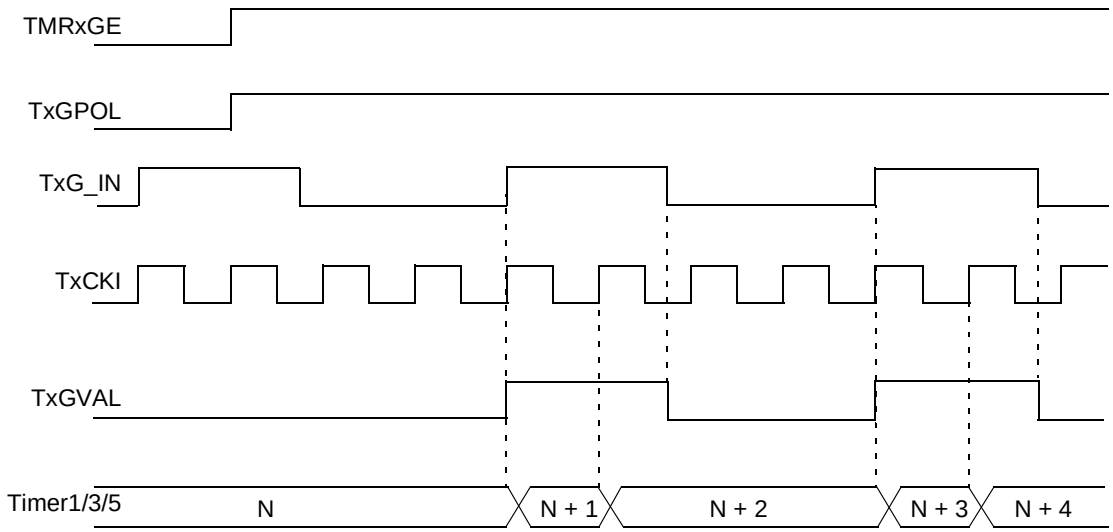
**FIGURE 12-3: TIMER1/3/5 INCREMENTING EDGE**




**Note 1:** Arrows indicate counter increments.

**2:** In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge of the clock.

**FIGURE 12-4: TIMER1/3/5 GATE ENABLE MODE**



**REGISTER 14-6: PWMxCON: ENHANCED PWM CONTROL REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| PxRSEN | PxDC<6:0> | | | | | | |
| 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 **PxRSEN:** PWM Restart Enable bit
- 1 = Upon auto-shutdown, the CCPxASE bit clears automatically once the shutdown event goes away; the PWM restarts automatically
- 0 = Upon auto-shutdown, CCPxASE must be cleared in software to restart the PWM

bit 6-0 **PxDC<6:0>:** PWM Delay Count bits

PxDCx = Number of FOSC/4 (4 * TOSC) cycles between the scheduled time when a PWM signal **should** transition active and the **actual** time it transitions active

**REGISTER 14-7: PSTRxCON: PWM STEERING CONTROL REGISTER(1)**

| U-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | — | STRxSYNC | STRxD | STRxC | STRxB | STRxA |
| bit 7 | | | | | | | bit 0 |

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

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

bit 4 **STRxSYNC:** Steering Sync bit
- 1 = Output steering update occurs on next PWM period
- 0 = Output steering update occurs at the beginning of the instruction cycle boundary

bit 3 **STRxD:** Steering Enable bit D
- 1 = PxD pin has the PWM waveform with polarity control from CCPxM<1:0>
- 0 = PxD pin is assigned to port pin

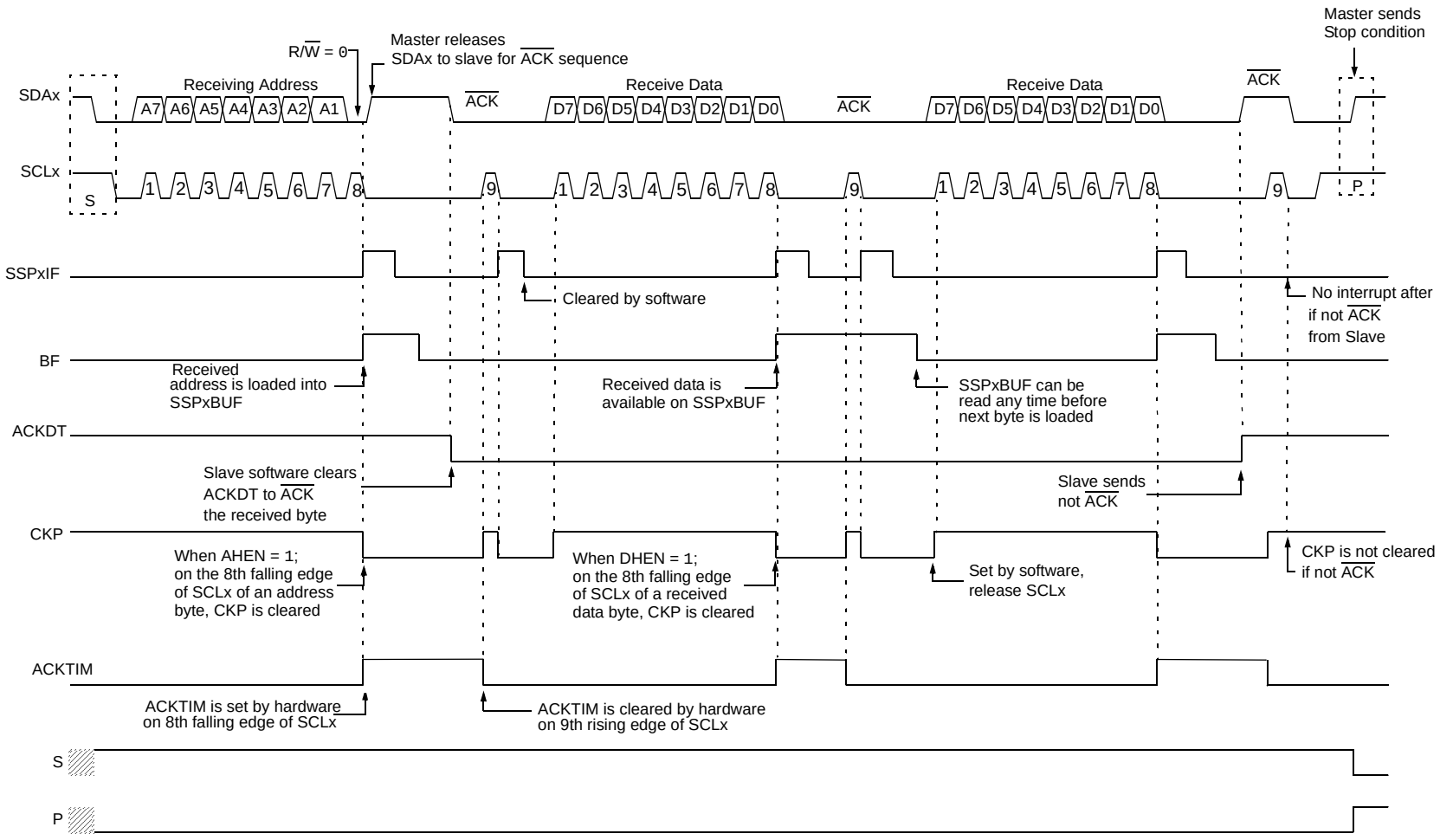
bit 2 **STRxC:** Steering Enable bit C
- 1 = PxC pin has the PWM waveform with polarity control from CCPxM<1:0>
- 0 = PxC pin is assigned to port pin

bit 1 **STRxB:** Steering Enable bit B
- 1 = PxB pin has the PWM waveform with polarity control from CCPxM<1:0>
- 0 = PxB pin is assigned to port pin

bit 0 **STRxA:** Steering Enable bit A
- 1 = PxA pin has the PWM waveform with polarity control from CCPxM<1:0>
- 0 = PxA pin is assigned to port pin

**Note 1:** The PWM Steering mode is available only when the CCPxCON register bits CCPxM<3:2> = 11 and PxM<1:0> = 00.

**FIGURE 15-17: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 1, DHEN = 1)**



**REGISTER 24-11: CONFIG7H: CONFIGURATION REGISTER 7 HIGH**

| U-0 | R/C-1 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | EBTRB | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | |
|---|---|
| R = Readable bit | U = Unimplemented bit, read as ‘0’ |
| -n = Value when device is unprogrammed | C = Clearable only bit |

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **EBTRB:** Boot Block Table Read Protection bit
1 = Boot Block not protected from table reads executed in other blocks
0 = Boot Block protected from table reads executed in other blocks

bit 5-0 **Unimplemented:** Read as ‘0’

**REGISTER 24-12: DEVID1: DEVICE ID REGISTER 1**

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

| **Legend:** | |
|---|---|
| R = Readable bit | U = Unimplemented bit, read as ‘0’ |
| -n = Value when device is unprogrammed | C = Clearable only bit |

bit 7-5 **DEV<2:0>:** Device ID bits
These bits, together with DEV<10:3> in DEVID2, determine the device ID.
See Table 24-2 for complete Device ID list.

bit 4-0 **REV<4:0>:** Revision ID bits
These bits indicate the device revision.

**REGISTER 24-13: DEVID2: DEVICE ID REGISTER 2**

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

| **Legend:** | |
|---|---|
| R = Readable bit | U = Unimplemented bit, read as ‘0’ |
| -n = Value when device is unprogrammed | C = Clearable only bit |

bit 7-0 **DEV<10:3>:** Device ID bits
These bits, together with DEV<2:0> in DEVID1, determine the device ID.
See Table 24-2 for complete Device ID list.

## 25.0 INSTRUCTION SET SUMMARY

PIC18(L)F2X/4XK22 devices incorporate the standard set of 75 PIC18 core instructions, as well as an extended set of eight new instructions, for the optimization of code that is recursive or that utilizes a software stack. The extended set is discussed later in this section.

### 25.1 Standard Instruction Set

The standard PIC18 instruction set adds many enhancements to the previous PIC® MCU instruction sets, while maintaining an easy migration from these PIC® MCU instruction sets. Most instructions are a single program memory word (16 bits), but there are four instructions that require two program memory locations.

Each single-word instruction is a 16-bit word divided into an opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into four basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal** operations
- **Control** operations

The PIC18 instruction set summary in Table 25-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 25-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The destination of the result (specified by ‘d’)
3. The accessed memory (specified by ‘a’)

The file register designator ‘f’ specifies which file register is to be used by the instruction. The destination designator ‘d’ specifies where the result of the operation is to be placed. If ‘d’ is zero, the result is placed in the WREG register. If ‘d’ is one, the result is placed in the file register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The bit in the file register (specified by ‘b’)
3. The accessed memory (specified by ‘a’)

The bit field designator ‘b’ selects the number of the bit affected by the operation, while the file register designator ‘f’ represents the number of the file in which the bit is located.

The **literal** instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by ‘k’)
- The desired FSR register to load the literal value into (specified by ‘f’)
- No operand required (specified by ‘—’)

The **control** instructions may use some of the following operands:

- A program memory address (specified by ‘n’)
- The mode of the CALL or RETURN instructions (specified by ‘s’)
- The mode of the table read and table write instructions (specified by ‘m’)
- No operand required (specified by ‘—’)

All instructions are a single word, except for four double-word instructions. These instructions were made double-word to contain the required information in 32 bits. In the second word, the four MSbs are ‘1’s. If this second word is executed as an instruction (by itself), it will execute as a NOP.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the program counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles, with the additional instruction cycle(s) executed as a NOP.

The double-word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 μs. If a conditional test is true, or the program counter is changed as a result of an instruction, the instruction execution time is 2 μs. Two-word branch instructions (if true) would take 3 μs.

Figure 25-1 shows the general formats that the instructions can have. All examples use the convention ‘nnh’ to represent a hexadecimal number.

The Instruction Set Summary, shown in Table 25-2, lists the standard instructions recognized by the Microchip Assembler (MPASM™).

**Section 25.1.1 “Standard Instruction Set”** provides a description of each instruction.

**BCF** **Bit Clear f**

Syntax: BCF f, b {,a}

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

Operation: $0 \rightarrow f<b>$

Status Affected: None

Encoding:

| 1001 | bbba | ffff | ffff |
|---|---|---|---|

Description: Bit 'b' in register 'f' is cleared.
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 25.2.3 "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: `BCF FLAG_REG, 7, 0`

Before Instruction
FLAG_REG = C7h
After Instruction
FLAG_REG = 47h

**BN** **Branch if Negative**

Syntax: BN n

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

Operation: if NEGATIVE bit is '1'
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

| 1110 | 0110 | nnnn | nnnn |
|---|---|---|---|

Description: If the NEGATIVE bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example:
```
HERE    BN   Jump
```
Before Instruction
PC = address (HERE)
After Instruction
If NEGATIVE = 1;
PC = address (Jump)
If NEGATIVE = 0;
PC = address (HERE + 2)

**BZ** **Branch if Zero**

Syntax: BZ n

Operands: -128 ≤ n ≤ 127

Operation: if ZERO bit is '1'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

| 1110 | 0000 | nnnn | nnnn |
|---|---|---|---|

Description: If the ZERO bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example: HERE BZ Jump

Before Instruction
PC = address (HERE)
After Instruction
If ZERO = 1;
PC = address (Jump)
If ZERO = 0;
PC = address (HERE + 2)

**CALL** **Subroutine Call**

Syntax: CALL k {,s}

Operands: 0 ≤ k ≤ 1048575
s ∈ [0,1]

Operation: (PC) + 4 → TOS,
k → PC<20:1>,
if s = 1
(W) → WS,
(Status) → STATUSS,
(BSR) → BSRS

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1st word (k<7:0>) | 1110 | 110s | $k_7$kkk | kkkk$_0$ |
| 2nd word(k<19:8>) | 1111 | $k_{19}$kkk | kkkk | kkkk$_8$ |

Description: Subroutine call of entire 2-Mbyte memory range. First, return address (PC + 4) is pushed onto the return stack. If 's' = 1, the W, STATUS and BSR registers are also pushed into their respective shadow registers, WS, STATUSS and BSRS. If 's' = 0, no update occurs (default). Then, the 20-bit value 'k' is loaded into PC<20:1>. CALL is a 2-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'k'<7:0>, | PUSH PC to stack | Read literal 'k'<19:8>, Write to PC |
| No operation | No operation | No operation | No operation |

Example: HERE CALL THERE, 1

Before Instruction
PC = address (HERE)
After Instruction
PC = address (THERE)
TOS = address (HERE + 4)
WS = W
BSRS = BSR
STATUSS = Status

**SUBFSR** **Subtract Literal from FSR**

Syntax: SUBFSR f, k

Operands: $0 \le k \le 63$

$f \in [0, 1, 2]$

Operation: FSR(f) – k → FSRf

Status Affected: None

Encoding:

| 1110 | 1001 | ffkk | kkkk |
|---|---|---|---|

Description: The 6-bit literal 'k' is subtracted from the contents of the FSR specified by 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: SUBFSR 2, 23h

Before Instruction
FSR2 = 03FFh

After Instruction
FSR2 = 03DCh

**SUBULNK** **Subtract Literal from FSR2 and Return**

Syntax: SUBULNK k

Operands: $0 \le k \le 63$

Operation: FSR2 – k → FSR2
(TOS) → PC

Status Affected: None

Encoding:

| 1110 | 1001 | 11kk | kkkk |
|---|---|---|---|

Description: The 6-bit literal 'k' is subtracted from the contents of the FSR2. A RETURN is then executed by loading the PC with the TOS. The instruction takes two cycles to execute; a NOP is performed during the second cycle.
This may be thought of as a special case of the SUBFSR instruction, where f = 3 (binary '11'); it operates only on FSR2.

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process Data | Write to destination |
| No Operation | No Operation | No Operation | No Operation |

Example: SUBULNK 23h

Before Instruction
FSR2 = 03FFh
PC = 0100h

After Instruction
FSR2 = 03DCh
PC = (TOS)

### 25.2.3 BYTE-ORIENTED AND BIT-ORIENTED INSTRUCTIONS IN INDEXED LITERAL OFFSET MODE

| **Note:** | Enabling the PIC18 instruction set extension may cause legacy applications to behave erratically or fail entirely. |
|---|---|

In addition to eight new commands in the extended set, enabling the extended instruction set also enables Indexed Literal Offset Addressing mode (**Section 5.7.1 "Indexed Addressing with Literal Offset"**). This has a significant impact on the way that many commands of the standard PIC18 instruction set are interpreted.

When the extended set is disabled, addresses embedded in opcodes are treated as literal memory locations: either as a location in the Access Bank ('a' = 0), or in a GPR bank designated by the BSR ('a' = 1). When the extended instruction set is enabled and 'a' = 0, however, a file register argument of 5Fh or less is interpreted as an offset from the pointer value in FSR2 and not as a literal address. For practical purposes, this means that all instructions that use the Access RAM bit as an argument – that is, all byte-oriented and bit-oriented instructions, or almost half of the core PIC18 instructions – may behave differently when the extended instruction set is enabled.

When the content of FSR2 is 00h, the boundaries of the Access RAM are essentially remapped to their original values. This may be useful in creating backward compatible code. If this technique is used, it may be necessary to save the value of FSR2 and restore it when moving back and forth between C and assembly routines in order to preserve the Stack Pointer. Users must also keep in mind the syntax requirements of the extended instruction set (see **Section 25.2.3.1 "Extended Instruction Syntax with Standard PIC18 Commands"**).

Although the Indexed Literal Offset Addressing mode can be very useful for dynamic stack and pointer manipulation, it can also be very annoying if a simple arithmetic operation is carried out on the wrong register. Users who are accustomed to the PIC18 programming must keep in mind that, when the extended instruction set is enabled, register addresses of 5Fh or less are used for Indexed Literal Offset Addressing.

Representative examples of typical byte-oriented and bit-oriented instructions in the Indexed Literal Offset Addressing mode are provided on the following page to show how execution is affected. The operand conditions shown in the examples are applicable to all instructions of these types.

#### 25.2.3.1 Extended Instruction Syntax with Standard PIC18 Commands

When the extended instruction set is enabled, the file register argument, 'f', in the standard byte-oriented and bit-oriented commands is replaced with the literal offset value, 'k'. As already noted, this occurs only when 'f' is less than or equal to 5Fh. When an offset value is used, it must be indicated by square brackets ("[ ]"). As with the extended instructions, the use of brackets indicates to the compiler that the value is to be interpreted as an index or an offset. Omitting the brackets, or using a value greater than 5Fh within brackets, will generate an error in the MPASM assembler.

If the index argument is properly bracketed for Indexed Literal Offset Addressing, the Access RAM argument is never specified; it will automatically be assumed to be '0'. This is in contrast to standard operation (extended instruction set disabled) when 'a' is set on the basis of the target address. Declaring the Access RAM bit in this mode will also generate an error in the MPASM assembler.

The destination argument, 'd', functions as before.

In the latest versions of the MPASM™ assembler, language support for the extended instruction set must be explicitly invoked. This is done with either the command line option, /y, or the PE directive in the source listing.

### 25.2.4 CONSIDERATIONS WHEN ENABLING THE EXTENDED INSTRUCTION SET

It is important to note that the extensions to the instruction set may not be beneficial to all users. In particular, users who are not writing code that uses a software stack may not benefit from using the extensions to the instruction set.

Additionally, the Indexed Literal Offset Addressing mode may create issues with legacy applications written to the PIC18 assembler. This is because instructions in the legacy code may attempt to address registers in the Access Bank below 5Fh. Since these addresses are interpreted as literal offsets to FSR2 when the instruction set extension is enabled, the application may read or write to the wrong data addresses.

When porting an application to the PIC18(L)F2X/4XK22, it is very important to consider the type of code. A large, re-entrant application that is written in 'C' and would benefit from efficient compilation will do well when using the instruction set extensions. Legacy applications that heavily use the Access Bank will most likely not benefit from using the extended instruction set.

## 27.4 DC Characteristics: RC Idle Supply Current, PIC18(L)F2X/4XK22

| PIC18LF2X/4XK22 | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +125°C | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC18F2X/4XK22** | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +125°C | | | | | |
| **Param No.** | **Device Characteristics** | **Typ** | **Max** | **Units** | **Conditions** | | |
| D045 | Supply Current (IDD)(1),(2) | 0.5 | 18 | µA | -40°C | VDD = 1.8V | FOSC = 31 kHz (**RC_IDLE** mode, LFINTOSC source) |
| | | 0.6 | 18 | µA | +25°C | | |
| | | 0.7 | — | µA | +60°C | | |
| | | 0.75 | 20 | µA | +85°C | | |
| | | 2.3 | 22 | µA | +125°C | | |
| D046 | | 1.1 | 20 | µA | -40°C | VDD = 3.0V | |
| | | 1.2 | 20 | µA | +25°C | | |
| | | 1.3 | — | µA | +60°C | | |
| | | 1.4 | 22 | µA | +85°C | | |
| | | 3.2 | 25 | µA | +125°C | | |
| D047 | | 17 | 30 | µA | -40°C | VDD = 2.3V | FOSC = 31 kHz (**RC_IDLE** mode, LFINTOSC source) |
| | | 13 | 30 | µA | +25°C | | |
| | | 14 | 30 | µA | +85°C | | |
| | | 15 | 45 | µA | +125°C | | |
| D048 | | 19 | 35 | µA | -40°C | VDD = 3.0V | |
| | | 15 | 35 | µA | +25°C | | |
| | | 16 | 35 | µA | +85°C | | |
| | | 17 | 50 | µA | +125°C | | |
| D049 | | 21 | 40 | µA | -40°C | VDD = 5.0V | |
| | | 15 | 40 | µA | +25°C | | |
| | | 16 | 40 | µA | +85°C | | |
| | | 18 | 60 | µA | +125°C | | |
| D050 | | 0.11 | 0.20 | mA | -40°C to +125°C | VDD = 1.8V | FOSC = 500 kHz (**RC_IDLE** mode, MFINTOSC source) |
| D051 | | 0.12 | 0.25 | mA | -40°C to +125°C | VDD = 3.0V | |
| D052 | | 0.14 | 0.21 | mA | -40°C to +125°C | VDD = 2.3V | FOSC = 500 kHz (**RC_IDLE** mode, MFINTOSC source) |
| D053 | | 0.15 | 0.25 | mA | -40°C to +125°C | VDD = 3.0V | |
| D054 | | 0.20 | 0.31 | mA | -40°C to +125°C | VDD = 5.0V | |

**Note 1:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

Test condition: All Peripheral Module Control bits in PMD0, PMD1 and PMD2 set to '1'.

**2:** The test conditions for all IDD measurements in active operation mode are:

All I/O pins set as outputs driven to Vss;

$\overline{MCLR}$ = VDD;

OSC1 = external square wave, from rail-to-rail (PRI_RUN and PRI_IDLE only).

## 27.10 Analog Characteristics

**TABLE 27-1: COMPARATOR SPECIFICATIONS**

| **Operating Conditions:** 1.8V < $V_{DD}$ < 5.5V, -40°C < TA < +125°C (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristics** | **Min** | **Typ** | **Max** | **Units** | **Comments** |
| CM01 | $V_{IOFF}$ | Input Offset Voltage | — | 3 | 40 | mV | High-Power mode $V_{REF}$ = $V_{DD}$/2 |
| | | | — | 4 | 60 | mV | Low-Power mode $V_{REF}$ = $V_{DD}$/2 |
| CM02 | $V_{ICM}$ | Input Common-mode Voltage | $V_{SS}$ | — | $V_{DD}$ | V | |
| CM04* | $T_{RESP}$ | Response Time(1) | — | 200 | 400 | ns | High-Power mode |
| | | | — | 600 | 3500 | ns | Low-Power mode |
| CM05* | $T_{MC2OV}$ | Comparator Mode Change to Output Valid | — | — | 10 | µs | |

* These parameters are characterized but not tested.

**Note 1:** Response time measured with one comparator input at $V_{DD}$/2, while the other input transitions from $V_{SS}$ to $V_{DD}$.

**TABLE 27-2: DIGITAL-TO-ANALOG CONVERTER (DAC) SPECIFICATIONS**

| **Operating Conditions:** 2.0V < $V_{DD}$ < 5.5V, -40°C < TA < +125°C (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristics** | **Min** | **Typ** | **Max** | **Units** | **Comments** |
| CV01* | $C_{LSB}$ | Step Size(2) | — | $V_{DD}$/32 | — | V | |
| CV02* | $C_{ACC}$ | Absolute Accuracy | — | — | ± 1/2 | LSb | $\Delta V_{SRC} \geq$ 2.0V |
| CV03* | $C_R$ | Unit Resistor Value (R) | — | 5k | — | Ω | |
| CV04* | $C_{ST}$ | Settling Time(1) | — | — | 10 | µs | |
| CV05* | $V_{SRC+}$ | DAC Positive Reference | $V_{SRC-}$ +2 | — | $V_{DD}$ | V | |
| CV06* | $V_{SRC-}$ | DAC Negative Reference | $V_{SS}$ | — | $V_{SRC+}$ -2 | V | |
| CV07* | $\Delta V_{SRC}$ | DAC Reference Range ($V_{SRC+}$ - $V_{SRC-}$) | 2 | — | $V_{DD}$ | V | |

* These parameters are characterized but not tested.

**Note 1:** Settling time measured while CVRR = 1 and CVR3:CVR0 transitions from '0000' to '1111'.

**2:** See **Section 22.0 "Digital-to-Analog Converter (DAC) Module"** for more information.

**FIGURE 28-1: PIC18LF2X/4XK22 BASE IPD**

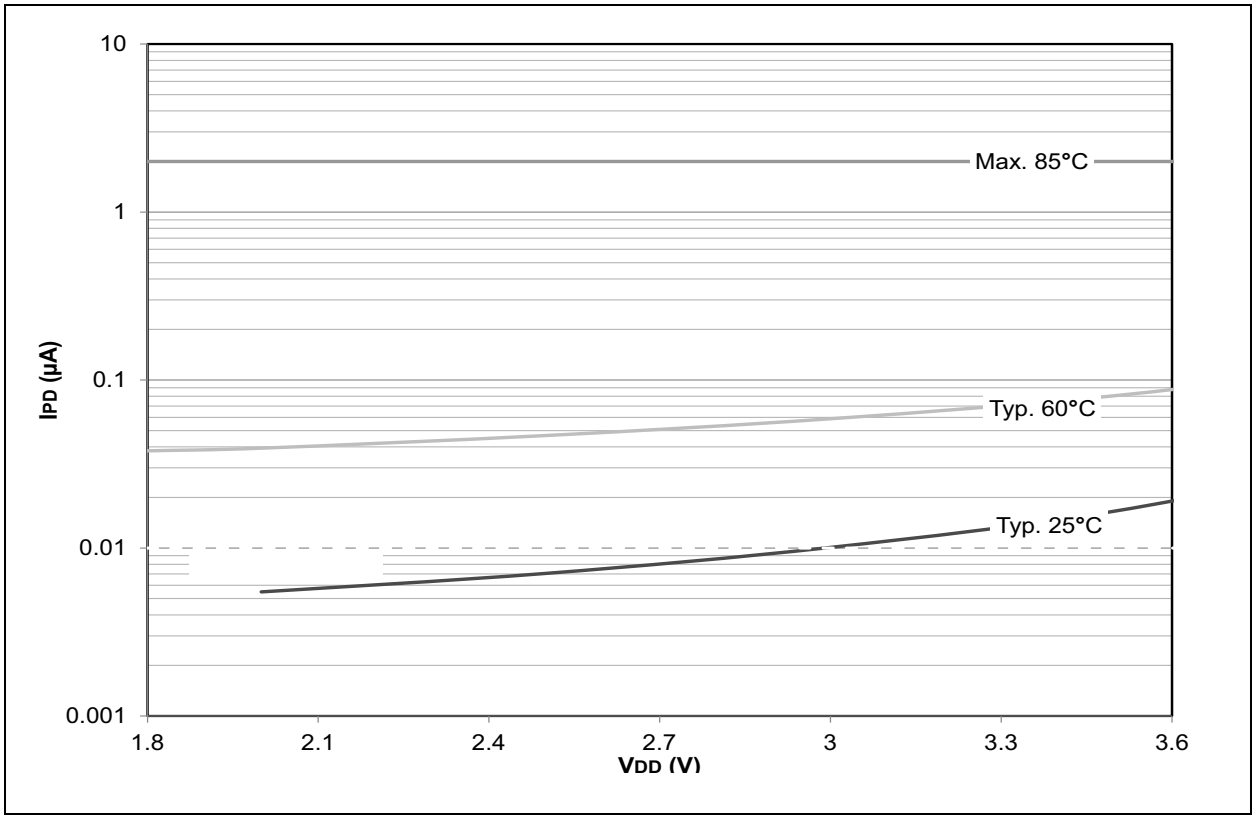



**FIGURE 28-2: PIC18F2X/4XK22 BASE IPD**

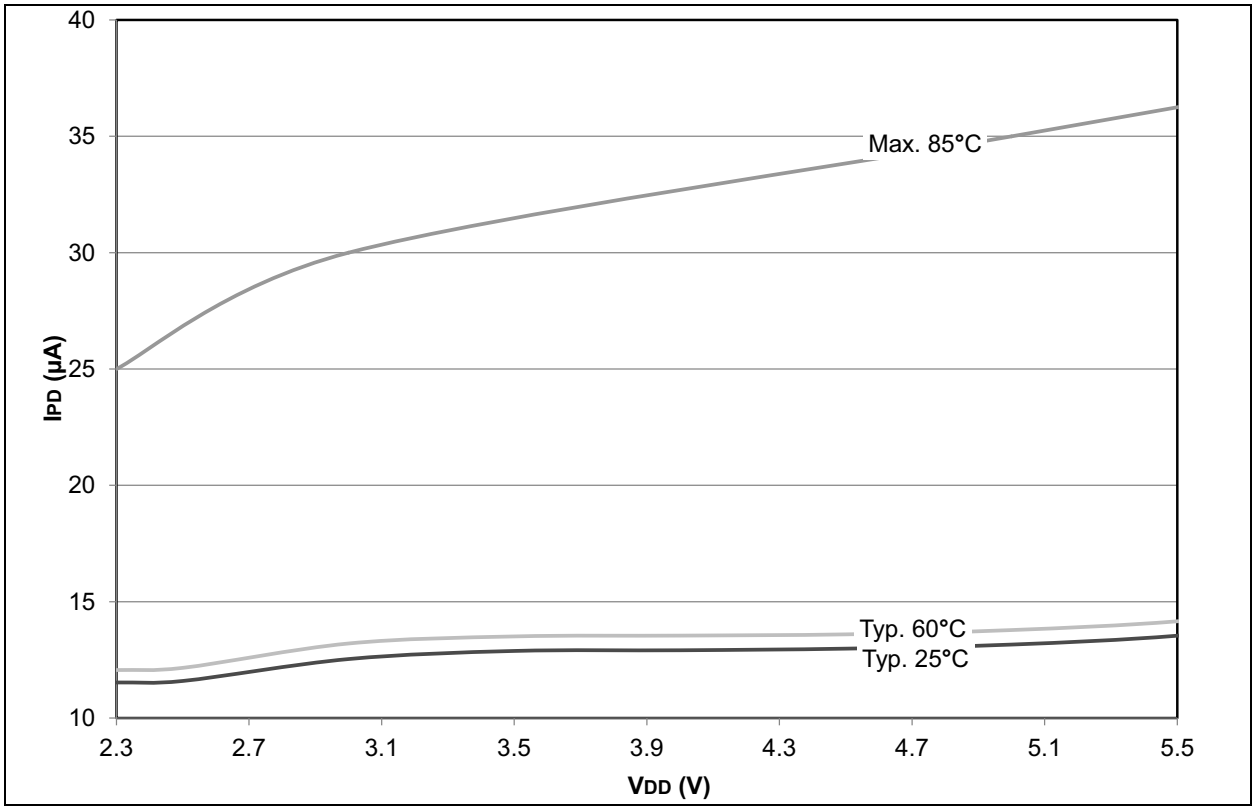


**FIGURE 28-84: PIC18(L)F2X/4XK22 PIN INPUT LEAKAGE**

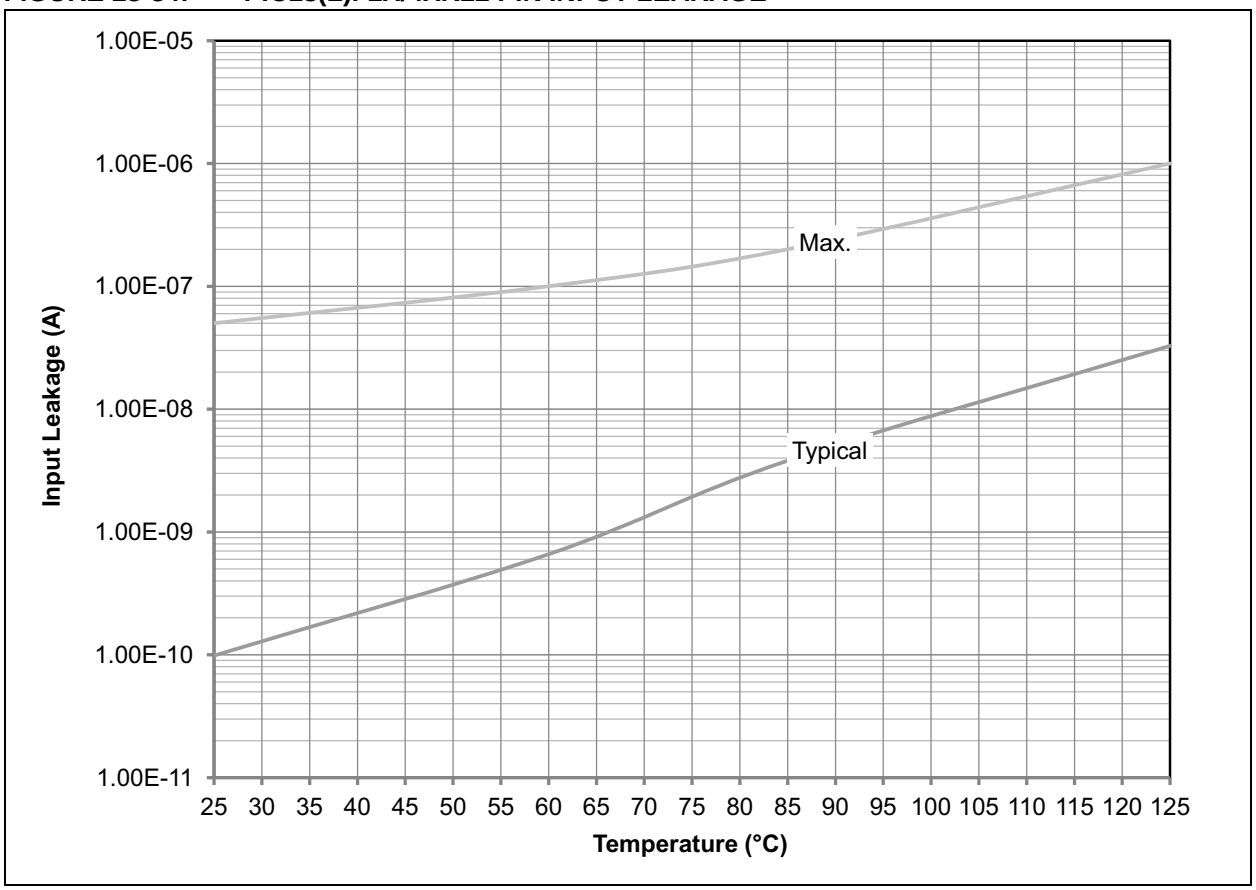


**FIGURE 28-93: PIC18(L)F2X/4XK22 TYPICAL DAC ABS. ERROR $V_{DD}$ = 2.5V, 3.0V, & 5.5V**