**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 | 16-Bit |
| Speed | 32MHz |
| Connectivity | I²C, IrDA, LINbus, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, HLVD, POR, PWM, WDT |
| Number of I/O | 18 |
| Program Memory Size | 16KB (5.5K x 24) |
| Program Memory Type | FLASH |
| EEPROM Size | 512 x 8 |
| RAM Size | 1K x 8 |
| Voltage - Supply (Vcc/Vdd) | 1.8V ~ 3.6V |
| Data Converters | A/D 12x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 125°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 20-SSOP (0.209", 5.30mm Width) |
| Supplier Device Package | 20-SSOP |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic24f16kl401-e-ss |

## Analog Features:

- 10-Bit, up to 12-Channel Analog-to-Digital (A/D) Converter:
  - 500 ksps conversion rate
  - Conversion available during Sleep and Idle
- Dual Rail-to-Rail Analog Comparators with Programmable Input/Output Configuration
- On-Chip Voltage Reference

## Special Microcontroller Features:

- Operating Voltage Range of 1.8V to 3.6V
- 10,000 Erase/Write Cycle Endurance Flash Program Memory, Typical
- 100,000 Erase/Write Cycle Endurance Data EEPROM, Typical
- Flash and Data EEPROM Data Retention: 40 Years Minimum
- Self-Programmable under Software Control
- Programmable Reference Clock Output

- Fail-Safe Clock Monitor (FSCM) Operation:
  - Detects clock failure and switches to on-chip, Low-Power RC (LPRC) oscillator
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Flexible Watchdog Timer (WDT):
  - Uses its own Low-Power RC oscillator
  - Windowed operating modes
  - Programmable period of 2 ms to 131s
- In-Circuit Serial Programming™ (ICSP™) and In-Circuit Emulation (ICE) via 2 Pins
- Programmable High/Low-Voltage Detect (HLVD)
- Programmable Brown-out Reset (BOR):
  - Configurable for software controlled operation and shutdown in Sleep mode
  - Selectable trip points (1.8V, 2.7V and 3.0V)
  - Low-power 2.0V POR re-arm

## TO OUR VALUED CUSTOMERS

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

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

### Most Current Data Sheet

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

**http://www.microchip.com**

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

### Errata

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

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

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

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

### Customer Notification System

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

## 4.2.2 DATA MEMORY ORGANIZATION AND ALIGNMENT

To maintain backward compatibility with PIC® devices and improve data space memory usage efficiency, the PIC24F instruction set supports both word and byte operations. As a consequence of byte accessibility, all Effective Address (EA) calculations are internally scaled to step through word-aligned memory. For example, the core recognizes that Post-Modified Register Indirect Addressing mode [Ws++] will result in a value of Ws + 1 for byte operations and Ws + 2 for word operations.

Data byte reads will read the complete word, which contains the byte, using the LSB of any EA to determine which byte to select. The selected byte is placed onto the LSB of the data path. That is, data memory and the registers are organized as two parallel, byte-wide entities with shared (word) address decode, but separate write lines. Data byte writes only write to the corresponding side of the array or register, which matches the byte address.

All word accesses must be aligned to an even address. Mis-aligned word data fetches are not supported, so care must be taken when mixing byte and word operations, or translating from 8-bit MCU code. If a mis-aligned read or write is attempted, an address error trap will be generated. If the error occurred on a read, the instruction underway is completed; if it occurred on a write, the instruction will be executed, but the write will not occur. In either case, a trap is then executed, allowing the system and/or user to examine the machine state prior to execution of the address Fault.

All byte loads into any W register are loaded into the LSB; the MSB is not modified.

A Sign-Extend (SE) instruction is provided to allow the users to translate 8-bit signed data to 16-bit signed values. Alternatively, for 16-bit unsigned data, users can clear the MSB of any W register by executing a Zero-Extend (ZE) instruction on the appropriate address.

Although most instructions are capable of operating on word or byte data sizes, it should be noted that some instructions operate only on words.

## 4.2.3 NEAR DATA SPACE

The 8-Kbyte area between 0000h and 1FFFh is referred to as the Near Data Space (NDS). Locations in this space are directly addressable via a 13-bit absolute address field within all memory direct instructions. The remainder of the data space is addressable indirectly. Additionally, the whole data space is addressable using MOV instructions, which support Memory Direct Addressing (MDA) with a 16-bit address field. For PIC24F16KL402 family devices, the entire implemented data memory lies in Near Data Space.

## 4.2.4 SFR SPACE

The first 2 Kbytes of the Near Data Space, from 0000h to 07FFh, are primarily occupied with Special Function Registers (SFRs). These are used by the PIC24F core and peripheral modules for controlling the operation of the device.

SFRs are distributed among the modules that they control and are generally grouped together by the module. Much of the SFR space contains unused addresses; these are read as '0'. The SFR space, where the SFRs are actually implemented, is provided in Table 4-2. Each implemented area indicates a 32-byte region, where at least one address is implemented as an SFR. A complete listing of implemented SFRs, including their addresses, is provided in Table 4-3 through Table 4-18.

**TABLE 4-2: IMPLEMENTED REGIONS OF SFR DATA SPACE**

| | SFR Space Address | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **xx00** | **xx20** | **xx40** | **xx60** | **xx80** | **xxA0** | **xxC0** | **xxE0** |
| **000h** | Core | | | ICN | Interrupts | | | — |
| **100h** | Timers | — | TMR | — | — | CCP | — | — | — |
| **200h** | MSSP | UART | | — | — | — | — | I/O | — |
| **300h** | A/D | | — | — | — | — | — | — |
| **400h** | — | — | — | — | — | — | — | ANSEL | — |
| **500h** | — | — | — | — | — | — | — | — |
| **600h** | — | CMP | — | — | — | — | — | — | — |
| **700h** | — | — | System | NVM/PMD | — | — | — | — |

**Legend:** — = No implemented SFRs in this block.

**FIGURE 8-1:** **PIC24F INTERRUPT VECTOR TABLE**

| | | |
|---|---|---|
| Reset – GOTO Instruction | 000000h | |
| Reset – GOTO Address | 000002h | |
| Reserved | 000004h | |
| Oscillator Fail Trap Vector | | |
| Address Error Trap Vector | | |
| Stack Error Trap Vector | | |
| Math Error Trap Vector | | |
| Reserved | | |
| Reserved | | |
| Reserved | | |
| Interrupt Vector 0 | 000014h | |
| Interrupt Vector 1 | | |
| — | | |
| — | | **Interrupt Vector Table (IVT)[1]** |
| — | | |
| Interrupt Vector 52 | 00007Ch | |
| Interrupt Vector 53 | 00007Eh | |
| Interrupt Vector 54 | 000080h | |
| — | | |
| — | | |
| — | | |
| Interrupt Vector 116 | 0000FCh | |
| Interrupt Vector 117 | 0000FEh | |
| Reserved | 000100h | |
| Reserved | 000102h | |
| Reserved | | |
| Oscillator Fail Trap Vector | | |
| Address Error Trap Vector | | |
| Stack Error Trap Vector | | |
| Math Error Trap Vector | | |
| Reserved | | |
| Reserved | | |
| Reserved | | |
| Interrupt Vector 0 | 000114h | |
| Interrupt Vector 1 | | |
| — | | **Alternate Interrupt Vector Table (AIVT)[1]** |
| — | | |
| — | | |
| Interrupt Vector 52 | 00017Ch | |
| Interrupt Vector 53 | 00017Eh | |
| Interrupt Vector 54 | 000180h | |
| — | | |
| — | | |
| — | | |
| Interrupt Vector 116 | | |
| Interrupt Vector 117 | 0001FEh | |
| Start of Code | 000200h | |

Decreasing Natural Order Priority

**Note 1:** See Table 8-2 for the interrupt vector list.

## 8.3 Interrupt Control and Status Registers

Depending on the particular device, the PIC24F16KL402 family of devices implements up to 28 registers for the interrupt controller:

- INTCON1
- INTCON2
- IFS0 through IFS5
- IEC0 through IEC5
- IPC0 through IPC7, ICP9, IPC12, ICP16, ICP18 and IPC20
- INTTREG

Global interrupt control functions are controlled from INTCON1 and INTCON2. INTCON1 contains the Interrupt Nesting Disable (NSTDIS) bit, as well as the control and status flags for the processor trap sources. The INTCON2 register controls the external interrupt request signal behavior and the use of the AIV table.

The IFSx registers maintain all of the interrupt request flags. Each source of interrupt has a status bit, which is set by the respective peripherals or external signal, and is cleared via software.

The IECx registers maintain all of the interrupt enable bits. These control bits are used to individually enable interrupts from the peripherals or external signals.

The IPCx registers are used to set the Interrupt Priority Level for each source of interrupt. Each user interrupt source can be assigned to one of eight priority levels.

The INTTREG register contains the associated interrupt vector number and the new CPU Interrupt Priority Level, which are latched into the Vector Number (VECNUM<6:0>) and the Interrupt Level (ILR<3:0>) bit fields in the INTTREG register. The new Interrupt Priority Level is the priority of the pending interrupt.

The interrupt sources are assigned to the IFSx, IECx and IPCx registers in the same sequence listed in Table 8-2. For example, the INT0 (External Interrupt 0) is depicted as having a vector number and a natural order priority of 0. The INT0IF status bit is found in IFS0<0>, the INT0IE enable bit in IEC0<0> and the INT0IP<2:0> priority bits are in the first position of IPC0 (IPC0<2:0>).

Although they are not specifically part of the interrupt control hardware, two of the CPU control registers contain bits that control interrupt functionality. The ALU STATUS Register (SR) contains the IPL<2:0> bits (SR<7:5>). These indicate the current CPU Interrupt Priority Level. The user may change the current CPU priority level by writing to the IPL bits.

The CORCON register contains the IPL3 bit, which together with the IPL<2:0> bits, also indicates the current CPU priority level. IPL3 is a read-only bit so that the trap events cannot be masked by the user's software.

All interrupt registers are described in Register 8-3 through Register 8-30, in the following sections.

### REGISTER 8-7: IFS2: INTERRUPT FLAG STATUS REGISTER 2

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|-----|-----|-------|-----|-----|-----|-----|-----|
| — | — | T3GIF | — | — | — | — | — |
| 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 15-6    **Unimplemented:** Read as '0'

bit 5    **T3GIF:** Timer3 External Gate Interrupt Flag Status bit

1 = Interrupt request has occurred
0 = Interrupt request has not occurred

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

### REGISTER 8-8: IFS3: INTERRUPT FLAG STATUS REGISTER 3

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 | R/W-0 | U-0 |
|-----|-----|-----|-----|-----|-------|-------|-----|
| — | — | — | — | — | BCL2IF[1] | SSP2IF[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 15-3    **Unimplemented:** Read as '0'

bit 2    **BCL2IF:** MSSP2 I$^2$C™ Bus Collision Interrupt Flag Status bit[1]

1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 1    **SSP2IF:** MSSP2 SPI/I$^2$C Event Interrupt Flag Status bit[1]

1 = Interrupt request has occurred
0 = Interrupt request has not occurred

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

**Note 1:** These bits are unimplemented on PIC24FXXKL10X and PIC24FXXKL20X devices.

## 9.3 Control Registers

The operation of the oscillator is controlled by three Special Function Registers (SFRs):

- OSCCON
- CLKDIV
- OSCTUN

The OSCCON register (Register 9-1) is the main control register for the oscillator. It controls clock source switching and allows the monitoring of clock sources.

The Clock Divider register (Register 9-2) controls the features associated with Doze mode, as well as the postscaler for the FRC oscillator.

The FRC Oscillator Tune register (Register 9-3) allows the user to fine-tune the FRC oscillator. OSCTUN functionality has been provided to help customers compensate for temperature effects on the FRC frequency over a wide range of temperatures. The tuning step-size is an approximation and is neither characterized nor tested.

**REGISTER 9-1:    OSCCON: OSCILLATOR CONTROL REGISTER**

| U-0 | R-0, HSC | R-0, HSC | R-0, HSC | U-0 | R/W-x[1] | R/W-x[1] | R/W-x[1] |
|---|---|---|---|---|---|---|---|
| — | COSC2 | COSC1 | COSC0 | — | NOSC2 | NOSC1 | NOSC0 |
| bit 15 | | | | | | | bit 8 |

| R/SO-0, HSC | U-0 | R-0, HSC[2] | U-0 | R/CO-0, HS | R/W-0[3] | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CLKLOCK | — | LOCK | — | CF | SOSCDRV | SOSCEN | OSWEN |
| bit 7 | | | | | | | bit 0 |

| Legend: | | HSC = Hardware Settable/Clearable bit | |
|---|---|---|---|
| HS = Hardware Settable bit | CO = Clearable Only bit | SO = Settable Only 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 15      **Unimplemented:** Read as '0'

bit 14-12      **COSC<2:0>:** Current Oscillator Selection bits

     111 = 8 MHz Fast RC Oscillator with Postscaler (FRCDIV)
     110 = 500 kHz Low-Power Fast RC Oscillator (FRC) with Postscaler (LPFRCDIV)
     101 = Low-Power RC Oscillator (LPRC)
     100 = Secondary Oscillator (SOSC)
     011 = Primary Oscillator with PLL module (XTPLL, HSPLL, ECPLL)
     010 = Primary Oscillator (XT, HS, EC)
     001 = 8 MHz FRC Oscillator with Postscaler and PLL module (FRCPLL)
     000 = 8 MHz FRC Oscillator (FRC)

bit 11      **Unimplemented:** Read as '0'

bit 10-8      **NOSC<2:0>:** New Oscillator Selection bits[1]

     111 = 8 MHz Fast RC Oscillator with Postscaler (FRCDIV)
     110 = 500 kHz Low-Power Fast RC Oscillator (FRC) with Postscaler (LPFRCDIV)
     101 = Low-Power RC Oscillator (LPRC)
     100 = Secondary Oscillator (SOSC)
     011 = Primary Oscillator with PLL module (XTPLL, HSPLL, ECPLL)
     010 = Primary Oscillator (XT, HS, EC)
     001 = 8 MHz FRC Oscillator with Postscaler and PLL module (FRCPLL)
     000 = 8 MHz FRC Oscillator (FRC)

**Note 1:** Reset values for these bits are determined by the FNOSC<2:0> Configuration bits.

     **2:** Also resets to '0' during any valid clock switch or whenever a non-PLL Clock mode is selected.

     **3:** When SOSC is selected to run from a digital clock input rather than an external crystal (SOSCSRC = 0), this bit has no effect.

**REGISTER 16-1: CCPxCON: CCPx CONTROL REGISTER (STANDARD CCP MODULES)**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | DCxB1 | DCxB0 | CCPxM3[1] | CCPxM2[1] | CCPxM1[1] | CCPxM0[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 15-6     **Unimplemented:** Read as '0'

bit 5-4      **DCxB<1:0>**: PWM Duty Cycle Bit 1 and Bit 0 for CCPx Module bits
             <u>Capture and Compare modes</u>:
             Unused.
             <u>PWM mode</u>:
             These bits are the two Least Significant bits (bit 1 and bit 0) of the 10-bit PWM duty cycle. The eight Most Significant bits (DCxB<9:2>) of the duty cycle are found in CCPRxL.

bit 3-0      **CCPxM<3:0>**: CCPx Module Mode Select bits[1]
             `1111` = Reserved
             `1110` = Reserved
             `1101` = Reserved
             `1100` = PWM mode
             `1011` = Compare mode: Special Event Trigger; resets timer on CCPx match (CCPxIF bit is set)
             `1010` = Compare mode: Generates software interrupt on compare match (CCPxIF bit is set, CCPx pin reflects I/O state)
             `1001` = Compare mode: Initializes CCPx pin high; on compare match, forces CCPx pin low (CCPxIF bit is set)
             `1000` = Compare mode: Initializes CCPx pin low; on compare match, forces CCPx pin high (CCPxIF bit is set)
             `0111` = Capture mode: Every 16th rising edge
             `0110` = Capture mode: Every 4th rising edge
             `0101` = Capture mode: Every rising edge
             `0100` = Capture mode: Every falling edge
             `0011` = Reserved
             `0010` = Compare mode: Toggles output on match (CCPxIF bit is set)
             `0001` = Reserved
             `0000` = Capture/Compare/PWM is disabled (resets CCPx module)

**Note 1:** CCPxM<3:0> = `1011` will only reset the timer and not start the A/D conversion on a CCPx match.

**REGISTER 16-6: CCPTMRS0: CCP TIMER SELECT CONTROL REGISTER 0[(1)]**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-0 | U-0 | U-0 | R/W-0 | U-0 | U-0 | R/W-0 |
|-----|-------|-----|-----|-------|-----|-----|-------|
| — | C3TSEL0 | — | — | C2TSEL0 | — | — | C1TSEL0 |
| 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 15-7 **Unimplemented:** Read as '0'

bit 6 **C3TSEL0:** CCP3 Timer Selection bit

1 = CCP3 uses TMR3/TMR4
0 = CCP3 uses TMR3/TMR2

bit 5-4 **Unimplemented:** Read as '0'

bit 3 **C2TSEL0:** CCP2 Timer Selection bit

1 = CCP2 uses TMR3/TMR4
0 = CCP2 uses TMR3/TMR2

bit 2-1 **Unimplemented:** Read as '0'

bit 0 **C1TSEL0:** CCP1/ECCP1 Timer Selection bit

1 = CCP1/ECCP1 uses TMR3/TMR4
0 = CCP1/ECCP1 uses TMR3/TMR2

**Note 1:** This register is unimplemented on PIC24FXXKL20X/10X devices; maintain as '0'.

**REGISTER 21-1:** CVRCON: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| CVREN | CVROE | CVRSS | CVR4 | CVR3 | CVR2 | CVR1 | CVR0 |
| 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 15-8      **Unimplemented:** Read as '0'

bit 7      **CVREN:** Comparator Voltage Reference Enable bit

         1 = $CV_{REF}$ circuit is powered on
         0 = $CV_{REF}$ circuit is powered down

bit 6      **CVROE:** Comparator $V_{REF}$ Output Enable bit

         1 = $CV_{REF}$ voltage level is output on the $CV_{REF}$ pin
         0 = $CV_{REF}$ voltage level is disconnected from the $CV_{REF}$ pin

bit 5      **CVRSS:** Comparator $V_{REF}$ Source Selection bit

         1 = Comparator reference source, $CV_{RSRC} = V_{REF+} - V_{REF-}$
         0 = Comparator reference source, $CV_{RSRC} = AV_{DD} - AV_{SS}$

bit 4-0      **CVR<4:0>:** Comparator $V_{REF}$ Value Selection $0 \leq CVR<4:0> \leq 31$ bits

         <u>When CVRSS = 1:</u>
         $CV_{REF} = (V_{REF-}) + (CVR<4:0>/32) \cdot (V_{REF+} - V_{REF-})$

         <u>When CVRSS = 0:</u>
         $CV_{REF} = (AV_{SS}) + (CVR<4:0>/32) \cdot (AV_{DD} - AV_{SS})$

## 25.0 INSTRUCTION SET SUMMARY

> **Note:** This chapter is a brief summary of the PIC24F Instruction Set Architecture (ISA) and is not intended to be a comprehensive reference source.

The PIC24F instruction set adds many enhancements to the previous PIC® MCU instruction sets, while maintaining an easy migration from previous PIC MCU instruction sets. Most instructions are a single program memory word. Only three instructions require two program memory locations.

Each single-word instruction is a 24-bit word divided into an 8-bit 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:

- Word or byte-oriented operations
- Bit-oriented operations
- Literal operations
- Control operations

Table 25-1 lists the general symbols used in describing the instructions. The PIC24F instruction set summary in Table 25-2 lists all the instructions, along with the status flags affected by each instruction.

Most word or byte-oriented W register instructions (including barrel shift instructions) have three operands:

- The first source operand, which is typically a register 'Wb' without any address modifier
- The second source operand, which is typically a register 'Ws' with or without an address modifier
- The destination of the result, which is typically a register 'Wd' with or without an address modifier

However, word or byte-oriented file register instructions have two operands:

- The file register specified by the value, 'f'
- The destination, which could either be the file register, 'f', or the W0 register, which is denoted as 'WREG'

Most bit-oriented instructions (including simple rotate/shift instructions) have two operands:

- The W register (with or without an address modifier) or file register (specified by the value of 'Ws' or 'f')
- The bit in the W register or file register (specified by a literal value or indirectly by the contents of register, 'Wb')

The literal instructions that involve data movement may use some of the following operands:

- A literal value to be loaded into a W register or file register (specified by the value of 'k')
- The W register or file register where the literal value is to be loaded (specified by 'Wb' or 'f')

However, literal instructions that involve arithmetic or logical operations use some of the following operands:

- The first source operand, which is a register 'Wb' without any address modifier
- The second source operand, which is a literal value
- The destination of the result (only if not the same as the first source operand), which is typically a register 'Wd' with or without an address modifier

The control instructions may use some of the following operands:

- A program memory address
- The mode of the Table Read and Table Write instructions

All instructions are a single word, except for certain double-word instructions, which were made double-word instructions so that all of the required information is available in these 48 bits. In the second word, the 8 MSbs are '0's. If this second word is executed as an instruction (by itself), it will execute as a NOP.

Most single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the Program Counter (PC) 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. Notable exceptions are the BRA (unconditional/computed branch), indirect CALL/GOTO, all Table Reads and Table Writes, and RETURN/RETFIE instructions, which are single-word instructions but take two or three cycles.

Certain instructions that involve skipping over the subsequent instruction require either two or three cycles if the skip is performed, depending on whether the instruction being skipped is a single-word or two-word instruction. Moreover, double-word moves require two cycles. The double-word instructions execute in two instruction cycles.

**TABLE 25-2: INSTRUCTION SET OVERVIEW (CONTINUED)**

| Assembly Mnemonic | Assembly Syntax | | Description | # of Words | # of Cycles | Status Flags Affected |
|---|---|---|---|---|---|---|
| GOTO | GOTO | Expr | Go to Address | 2 | 2 | None |
| | GOTO | Wn | Go to Indirect | 1 | 2 | None |
| INC | INC | f | f = f + 1 | 1 | 1 | C, DC, N, OV, Z |
| | INC | f,WREG | WREG = f + 1 | 1 | 1 | C, DC, N, OV, Z |
| | INC | Ws,Wd | Wd = Ws + 1 | 1 | 1 | C, DC, N, OV, Z |
| INC2 | INC2 | f | f = f + 2 | 1 | 1 | C, DC, N, OV, Z |
| | INC2 | f,WREG | WREG = f + 2 | 1 | 1 | C, DC, N, OV, Z |
| | INC2 | Ws,Wd | Wd = Ws + 2 | 1 | 1 | C, DC, N, OV, Z |
| IOR | IOR | f | f = f .IOR. WREG | 1 | 1 | N, Z |
| | IOR | f,WREG | WREG = f .IOR. WREG | 1 | 1 | N, Z |
| | IOR | #lit10,Wn | Wd = lit10 .IOR. Wd | 1 | 1 | N, Z |
| | IOR | Wb,Ws,Wd | Wd = Wb .IOR. Ws | 1 | 1 | N, Z |
| | IOR | Wb,#lit5,Wd | Wd = Wb .IOR. lit5 | 1 | 1 | N, Z |
| LNK | LNK | #lit14 | Link Frame Pointer | 1 | 1 | None |
| LSR | LSR | f | f = Logical Right Shift f | 1 | 1 | C, N, OV, Z |
| | LSR | f,WREG | WREG = Logical Right Shift f | 1 | 1 | C, N, OV, Z |
| | LSR | Ws,Wd | Wd = Logical Right Shift Ws | 1 | 1 | C, N, OV, Z |
| | LSR | Wb,Wns,Wnd | Wnd = Logical Right Shift Wb by Wns | 1 | 1 | N, Z |
| | LSR | Wb,#lit5,Wnd | Wnd = Logical Right Shift Wb by lit5 | 1 | 1 | N, Z |
| MOV | MOV | f,Wn | Move f to Wn | 1 | 1 | None |
| | MOV | [Wns+Slit10],Wnd | Move [Wns+Slit10] to Wnd | 1 | 1 | None |
| | MOV | f | Move f to f | 1 | 1 | N, Z |
| | MOV | f,WREG | Move f to WREG | 1 | 1 | None |
| | MOV | #lit16,Wn | Move 16-bit Literal to Wn | 1 | 1 | None |
| | MOV.b | #lit8,Wn | Move 8-bit Literal to Wn | 1 | 1 | None |
| | MOV | Wn,f | Move Wn to f | 1 | 1 | None |
| | MOV | Wns,[Wns+Slit10] | Move Wns to [Wns+Slit10] | 1 | 1 | None |
| | MOV | Wso,Wdo | Move Ws to Wd | 1 | 1 | None |
| | MOV | WREG,f | Move WREG to f | 1 | 1 | None |
| | MOV.D | Wns,Wd | Move Double from W(ns):W(ns+1) to Wd | 1 | 2 | None |
| | MOV.D | Ws,Wnd | Move Double from Ws to W(nd+1):W(nd) | 1 | 2 | None |
| MUL | MUL.SS | Wb,Ws,Wnd | {Wnd+1, Wnd} = Signed(Wb) * Signed(Ws) | 1 | 1 | None |
| | MUL.SU | Wb,Ws,Wnd | {Wnd+1, Wnd} = Signed(Wb) * Unsigned(Ws) | 1 | 1 | None |
| | MUL.US | Wb,Ws,Wnd | {Wnd+1, Wnd} = Unsigned(Wb) * Signed(Ws) | 1 | 1 | None |
| | MUL.UU | Wb,Ws,Wnd | {Wnd+1, Wnd} = Unsigned(Wb) * Unsigned(Ws) | 1 | 1 | None |
| | MUL.SU | Wb,#lit5,Wnd | {Wnd+1, Wnd} = Signed(Wb) * Unsigned(lit5) | 1 | 1 | None |
| | MUL.UU | Wb,#lit5,Wnd | {Wnd+1, Wnd} = Unsigned(Wb) * Unsigned(lit5) | 1 | 1 | None |
| | MUL | f | W3:W2 = f * WREG | 1 | 1 | None |
| NEG | NEG | f | f = $\bar{f}$ + 1 | 1 | 1 | C, DC, N, OV, Z |
| | NEG | f,WREG | WREG = $\bar{f}$ + 1 | 1 | 1 | C, DC, N, OV, Z |
| | NEG | Ws,Wd | Wd = $\overline{Ws}$ + 1 | 1 | 1 | C, DC, N, OV, Z |
| NOP | NOP | | No Operation | 1 | 1 | None |
| | NOPR | | No Operation | 1 | 1 | None |
| POP | POP | f | Pop f from Top-of-Stack (TOS) | 1 | 1 | None |
| | POP | Wdo | Pop from Top-of-Stack (TOS) to Wdo | 1 | 1 | None |
| | POP.D | Wnd | Pop from Top-of-Stack (TOS) to W(nd):W(nd+1) | 1 | 2 | None |
| | POP.S | | Pop Shadow Registers | 1 | 1 | All |
| PUSH | PUSH | f | Push f to Top-of-Stack (TOS) | 1 | 1 | None |
| | PUSH | Wso | Push Wso to Top-of-Stack (TOS) | 1 | 1 | None |
| | PUSH.D | Wns | Push W(ns):W(ns+1) to Top-of-Stack (TOS) | 1 | 2 | None |
| | PUSH.S | | Push Shadow Registers | 1 | 1 | None |

**TABLE 25-2: INSTRUCTION SET OVERVIEW (CONTINUED)**

| Assembly Mnemonic | Assembly Syntax | | Description | # of Words | # of Cycles | Status Flags Affected |
|---|---|---|---|---|---|---|
| TBLRDH | TBLRDH | Ws,Wd | Read Prog<23:16> to Wd<7:0> | 1 | 2 | None |
| TBLRDL | TBLRDL | Ws,Wd | Read Prog<15:0> to Wd | 1 | 2 | None |
| TBLWTH | TBLWTH | Ws,Wd | Write Ws<7:0> to Prog<23:16> | 1 | 2 | None |
| TBLWTL | TBLWTL | Ws,Wd | Write Ws to Prog<15:0> | 1 | 2 | None |
| ULNK | ULNK | | Unlink Frame Pointer | 1 | 1 | None |
| XOR | XOR | f | f = f .XOR. WREG | 1 | 1 | N, Z |
| | XOR | f,WREG | WREG = f .XOR. WREG | 1 | 1 | N, Z |
| | XOR | #lit10,Wn | Wd = lit10 .XOR. Wd | 1 | 1 | N, Z |
| | XOR | Wb,Ws,Wd | Wd = Wb .XOR. Ws | 1 | 1 | N, Z |
| | XOR | Wb,#lit5,Wd | Wd = Wb .XOR. lit5 | 1 | 1 | N, Z |
| ZE | ZE | Ws,Wnd | Wnd = Zero-Extend Ws | 1 | 1 | C, Z, N |

**TABLE 26-4:** **HIGH/LOW–VOLTAGE DETECT CHARACTERISTICS**

| Standard Operating Conditions (unless otherwise stated) |
|---|

**Standard Operating Conditions (unless otherwise stated)**
Operating temperature  -40°C ≤ TA ≤ +85°C for industrial
                       -40°C ≤ TA ≤ +125°C for Extended

| Param No. | Symbol | Characteristic | | Min | Typ | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| DC18 | VHLVD | HLVD Voltage on VDD Transition | HLVDL<3:0> = `0000` | — | 1.85 | 1.94 | V | |
| | | | HLVDL<3:0> = `0001` | 1.81 | 1.90 | 2.00 | V | |
| | | | HLVDL<3:0> = `0010` | 1.85 | 1.95 | 2.05 | V | |
| | | | HLVDL<3:0> = `0011` | 1.90 | 2.00 | 2.10 | V | |
| | | | HLVDL<3:0> = `0100` | 1.95 | 2.05 | 2.15 | V | |
| | | | HLVDL<3:0> = `0101` | 2.06 | 2.17 | 2.28 | V | |
| | | | HLVDL<3:0> = `0110` | 2.12 | 2.23 | 2.34 | V | |
| | | | HLVDL<3:0> = `0111` | 2.24 | 2.36 | 2.48 | V | |
| | | | HLVDL<3:0> = `1000` | 2.31 | 2.43 | 2.55 | V | |
| | | | HLVDL<3:0> = `1001` | 2.47 | 2.60 | 2.73 | V | |
| | | | HLVDL<3:0> = `1010` | 2.64 | 2.78 | 2.92 | V | |
| | | | HLVDL<3:0> = `1011` | 2.74 | 2.88 | 3.02 | V | |
| | | | HLVDL<3:0> = `1100` | 2.85 | 3.00 | 3.15 | V | |
| | | | HLVDL<3:0> = `1101` | 2.96 | 3.12 | 3.28 | V | |
| | | | HLVDL<3:0> = `1110` | 3.22 | 3.39 | 3.56 | V | |

**TABLE 26-5:** **BOR TRIP POINTS**

**Standard Operating Conditions: 1.8V to 3.6V**
Operating temperature        -40°C ≤ TA ≤ +85°C for Industrial
                             -40°C ≤ TA ≤ +125°C for Extended

| Param No. | Symbol | Characteristic | | Min | Typ | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| DC19 | | BOR Voltage on VDD Transition | BORV = `00` | 1.85 | 2.0 | 2.15 | V | **Note 1** |
| | | | BORV = `01` | 2.90 | 3.0 | 3.38 | V | |
| | | | BORV = `10` | 2.53 | 2.7 | 3.07 | V | |
| | | | BORV = `11` | 1.75 | 1.85 | 2.05 | V | |

**Note 1:** LPBOR re-arms the POR circuit but does not cause a BOR.

**FIGURE 26-6:** **CAPTURE/COMPARE/PWM TIMINGS (ECCP1, ECCP2 MODULES)**



**Note:** Refer to Figure 26-3 for load conditions.
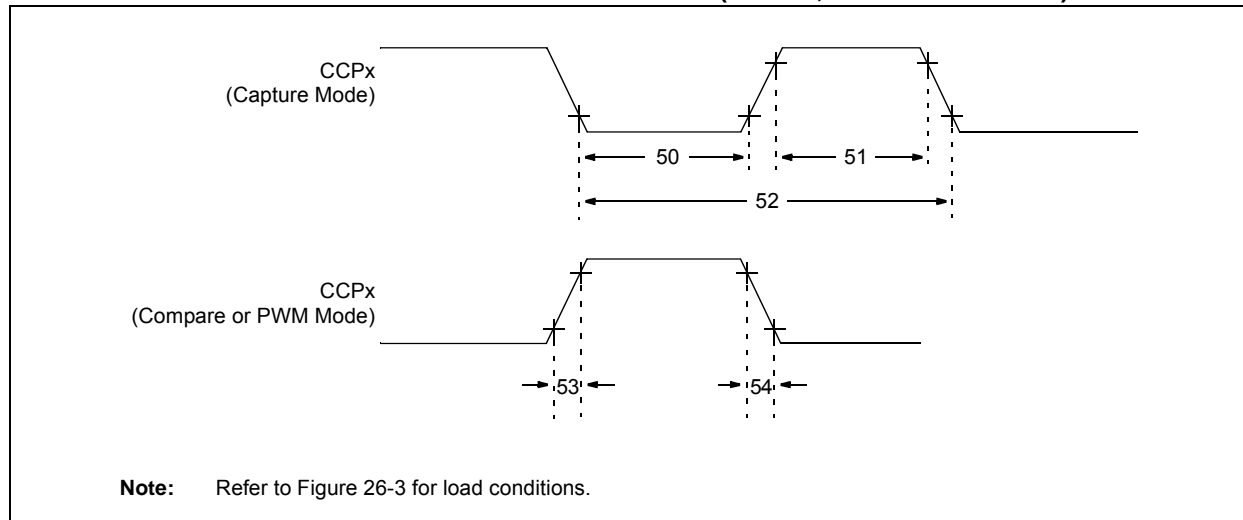
**TABLE 26-26:** **CAPTURE/COMPARE/PWM REQUIREMENTS (ECCP1, ECCP2 MODULES)**

| Param No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 50 | TccL | CCPx Input Low Time | No Prescaler | $0.5 T_{CY} + 20$ | — | ns | |
| | | | With Prescaler | 20 | — | ns | |
| 51 | TccH | CCPx Input High Time | No Prescaler | $0.5 T_{CY} + 20$ | — | ns | |
| | | | With Prescaler | 20 | — | ns | |
| 52 | TccP | CCPx Input Period | | Greater of: 40 or $\frac{2 T_{CY} + 40}{N}$ | — | ns | N = prescale value (1, 4 or 16) |
| 53 | TccR | CCPx Output Fall Time | | — | 25 | ns | |
| 54 | TccF | CCPx Output Fall Time | | — | 25 | ns | |

**FIGURE 26-13:** **MSSPx I²C™ BUS START/STOP BITS TIMING WAVEFORMS**



Note: Refer to Figure 26-3 for load conditions.
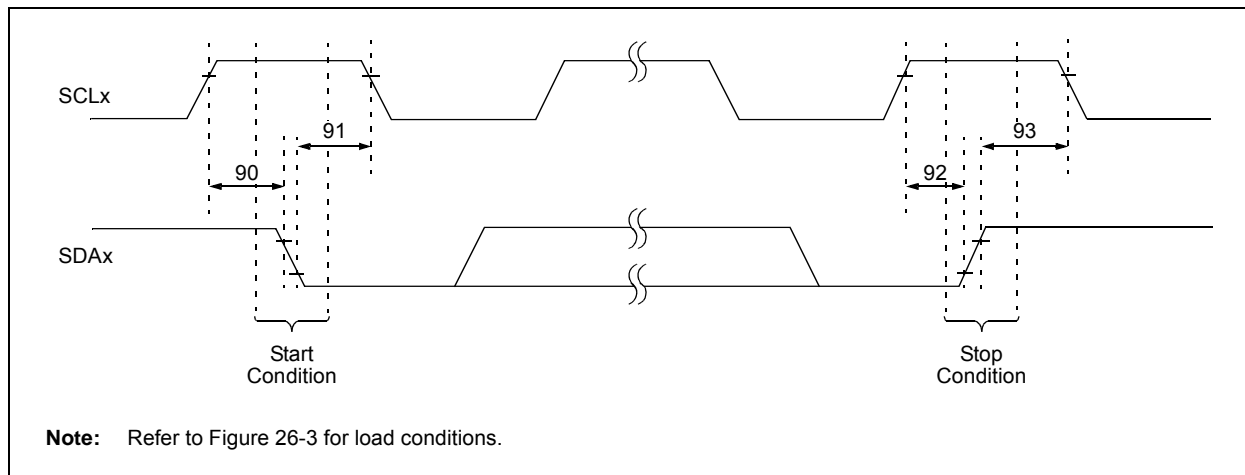
**TABLE 26-33:** **I²C™ BUS START/STOP BITS REQUIREMENTS (MASTER MODE)**

| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 90 | $T_{SU:STA}$ | Start Condition Setup Time | 100 kHz mode | 2($T_{OSC}$)(BRG + 1) | — | ns | Only relevant for Repeated Start condition |
| | | | 400 kHz mode | 2($T_{OSC}$)(BRG + 1) | — | | |
| 91 | $T_{HD:STA}$ | Start Condition Hold Time | 100 kHz mode | 2($T_{OSC}$)(BRG + 1) | — | ns | After this period, the first clock pulse is generated |
| | | | 400 kHz mode | 2($T_{OSC}$)(BRG + 1) | — | | |
| 92 | $T_{SU:STO}$ | Stop Condition Setup Time | 100 kHz mode | 2($T_{OSC}$)(BRG + 1) | — | ns | |
| | | | 400 kHz mode | 2($T_{OSC}$)(BRG + 1) | — | | |
| 93 | $T_{HD:STO}$ | Stop Condition Hold Time | 100 kHz mode | 2($T_{OSC}$)(BRG + 1) | — | ns | |
| | | | 400 kHz mode | 2($T_{OSC}$)(BRG + 1) | — | | |

28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

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



SILK SCREEN

C

Y1

G

E

X1
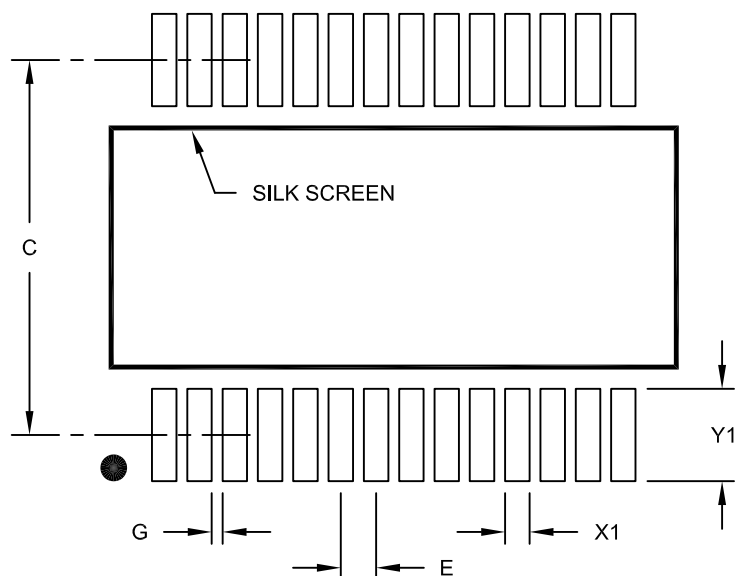
RECOMMENDED LAND PATTERN

| | Units | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 0.65 BSC | |
| Contact Pad Spacing | C | | 7.20 | |
| Contact Pad Width (X28) | X1 | | | 0.45 |
| Contact Pad Length (X28) | Y1 | | | 1.75 |
| Distance Between Pads | G | 0.20 | | |

Notes:
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2073A

## APPENDIX A: REVISION HISTORY

### Revision A (September 2011)

Original data sheet for the PIC24F16KL402 family of devices.

### Revision B (November 2011)

Updates DC Specifications in Tables 26-6 through 26-9 (all Typical and Maximum values).

Updates AC Specifications in Tables 26-7 through 26-30 (SPI Timing Requirements) with the addition of the FSCK specification.

Other minor typographic corrections throughout.

### Revision C (October 2013)

Adds +125°C Extended Temperature information.

Updates several packaging drawings in **Section 27.0 "Packaging Information"**. Other minor typographic corrections throughout.

## APPENDIX B: MIGRATING FROM PIC18/PIC24 TO PIC24F16KL402

The PIC24F16KL402 family combines traditional PIC18 peripherals with a faster PIC24 core to provide a low-cost, high-performance microcontroller with low-power consumption.

Code written for PIC18 devices can be migrated to the PIC24F16KL402 by using a C compiler that generates PIC24 machine level instructions. Assembly language code will need to be rewritten using PIC24 instructions. The PIC24 instruction set shares similarities to the PIC18 instruction set, which should ease porting of assembly code. Application code will require changes to support certain PIC24 peripherals.

Code written for PIC24 devices can be migrated to the PIC24F16KL402 without many code changes. Certain peripherals, however, will require application changes to support modules that were traditionally available only on PIC18 devices.

Refer to Table B-1 for a list of peripheral modules on the PIC24F16KL402 and where they originated from.

**TABLE B-1:** TABLE B-1: PIC24F16KL402 PERIPHERAL MODULE ORIGINATING ARCHITECTURE

| Peripheral Module | PIC18 | PIC24 |
|---|---|---|
| ECCP/CCP | X | — |
| MSSP (I$^2$C™/SPI) | X | — |
| Timer2/4 (8-bit) | X | — |
| Timer3 (16-bit) | X | — |
| Timer1 (16-bit) | — | X |
| 10-Bit A/D Converter | — | X |
| Comparator | — | X |
| Comparator Voltage Reference | — | X |
| UART | — | X |
| HLVD | — | X |

# INDEX

**NOTES:**