Section 10. I/O Ports

HIGHLIGHTS

This section contains these major topics:

© 2008 Microchip Technology Inc.

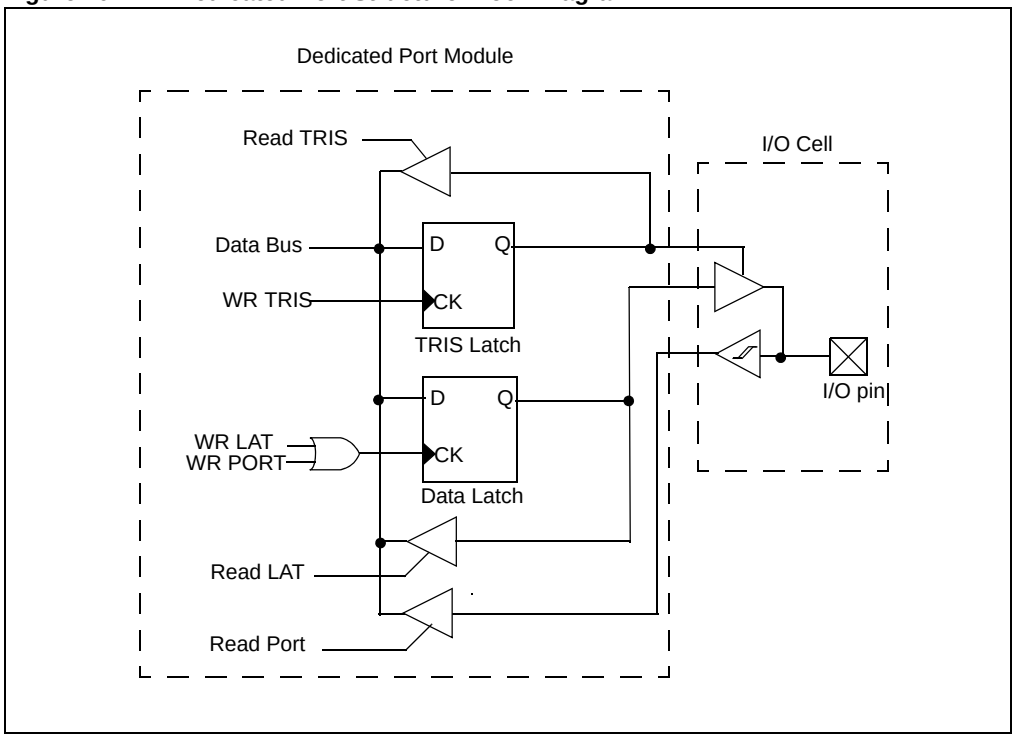
10.1 INTRODUCTION

This section provides information on the I/O ports for the PIC24H device family. All the device pins (except VDD, VSS, MCLR, and OSC1/CLKI) are shared between the peripherals and the general purpose I/O ports.

The general purpose I/O ports allow the PIC24H to monitor and control other devices. Most I/O pins are multiplexed with alternate function(s). The multiplexing depends on the peripheral features on the device variant. In general, when a peripheral is functioning, that pin may not be used as a general purpose I/O pin.

Figure 10-1 shows a block diagram of a typical I/O port. This block diagram does not take into account peripheral functions that may be multiplexed onto the I/O pin.

Figure 10-1: Dedicated Port Structure Block Diagram



© 2008 Microchip Technology Inc.

10.2 I/O PORTx CONTROL REGISTER

All I/O ports have four registers directly associated with the operation of the port, where 'x' is a letter that denotes the particular I/O port:

- TRISx: Data Direction register
- PORTx: I/O Port register
- LATx: I/O Latch register
- ODCx: Open-Drain control register

Each I/O pin on the device has an associated bit in the TRIS, PORT and LAT registers.

Note:	The total number of ports and available I/O pins depend on the device variant. In a given device, all of the bits in a port control register may not be implemented. For further details, refer to the specific device data sheet.

10.2.1 TRIS Registers

The TRISx register control bits determine whether each pin associated with the I/O port is an input or an output. If the TRIS bit for an I/O pin is a '1', then the pin is an input. If the TRIS bit for an I/O pin is a '0', then the pin is configured for an output. An easy way to remember is that '1' looks like an I (input) and a '0' looks like an O (output). All port pins are defined as inputs after a Reset.

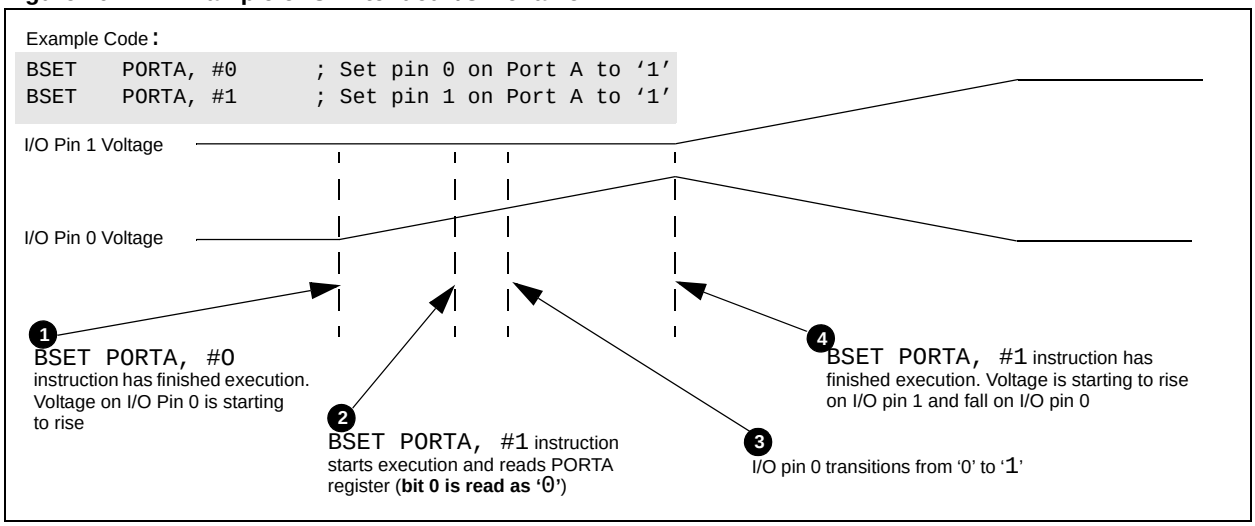
10.2.2 PORTx Registers

Data on an I/O pin is accessed via a PORTx register. A read of the PORTx register reads the value of the I/O pin, while a write to the PORTx register writes the value to the port data latch.

Many instructions, such as BSET and BCLR instructions, are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified, and then written to the port data latch. Care should be taken when read-modify-write commands are used on the PORTx registers and when some I/O pins associated with the port are configured as inputs. If an I/O pin configured as an input is changed to an output at a later time, an unexpected value may be output on the I/O pin. This effect occurs because the read-modify-write instruction reads the instantaneous value on the input pin and loads that value into the port data latch.

In addition, if read-modified-write instructions are used on the PORTx registers while I/O pins are configured as output, unintended I/O behavior may occur based on the device speed and I/O capacitive loading. Figure 10-2 illustrates unintended behavior that occurs if the user application attempts to set I/O bits 0 and 1 on Port A with two consecutive read-modify-write instructions in the PORTA register. At high CPU speeds and high capacitive loading on the I/O pins, the unintended result of the example code is that only I/O bit 1 is set high.

Figure 10-2: Example of Unintended I/O Behavior



When the first BSET instruction is executed, it writes a '1' to bit 0 in the PORTA register, which causes the voltage level to start rising to logic level 1 on pin 0 (refer to Step 1 in Figure 10-2). However, if the second BSET instruction is executed before the voltage level on pin 0 has reached the threshold for logic 1 (refer to Step 3 in Figure 10-2), the second BSET (read-modified-write) instruction reads '0' for bit 0, which it writes back into PORTA register (refer to Step 2 in Figure 10-2). In other words, instead of reading a value of 0x0001 from the PORTA register, it reads a value of 0x0000, modifies it to 0x0002 (instead of desired value of 0x0003), and writes that value back to the PORTA register. This causes the voltage on pin 0 to start falling to logic 0 and the voltage on pin 1 to start rising to logic 1 (refer to Step 4 in Figure 10-2).

10.2.3 LAT Registers

The LATx register associated with an I/O pin eliminates the problems that could occur with read-modify-write instructions. A read of the LATx register returns the values held in the port output latches, instead of the values on the I/O pins. A read-modify-write operation on the LAT register, associated with an I/O port, avoids the possibility of writing the input pin values into the port latches. A write to the LATx register has the same effect as a write to the PORTx register.

The following example uses the LATx register to set two I/O bits:

Example 10-1: Setting I/O Pins with LATx Register

```
BSET    LATA, #0              ;Set pin 0 on Port A to '1'
BSET    LATA, #1              ;Set pin 1 on Port A to '1'
```

The differences between the PORTx and LAT registers can be summarized as follows:

- A write to the PORTx register writes the data value to the port latch
- A write to the LATx register writes the data value to the port latch
- A read of the PORTx register reads the data value on the I/O pin
- A read of the LATx register reads the data value held in the port latch

Any bit and its associated data and control registers that are not valid for a particular device are disabled. That means the corresponding LATx and TRISx registers, and the port pin, read as zeros.

10.2.4 Open-Drain Control Registers

In addition to the PORTx, LAT and TRIS registers for data control, each port pin can also be individually configured for either digital or open-drain output. This is controlled by the Open-Drain Control register, ODCx, associated with each port. Setting any of the bits configures the corresponding pin to act as an open-drain output.

The open-drain feature allows the generation of outputs higher than V_{DD} (e.g., 5V) on any desired digital only pins by using external pull-up resistors. (The open-drain I/O feature is not supported on pins that have analog functionality multiplexed on the pin.) The maximum open-drain voltage allowed is the same as the maximum V_{IH} specification. The open-drain output feature is supported for both port pin and peripheral configurations.

© 2008 Microchip Technology Inc.

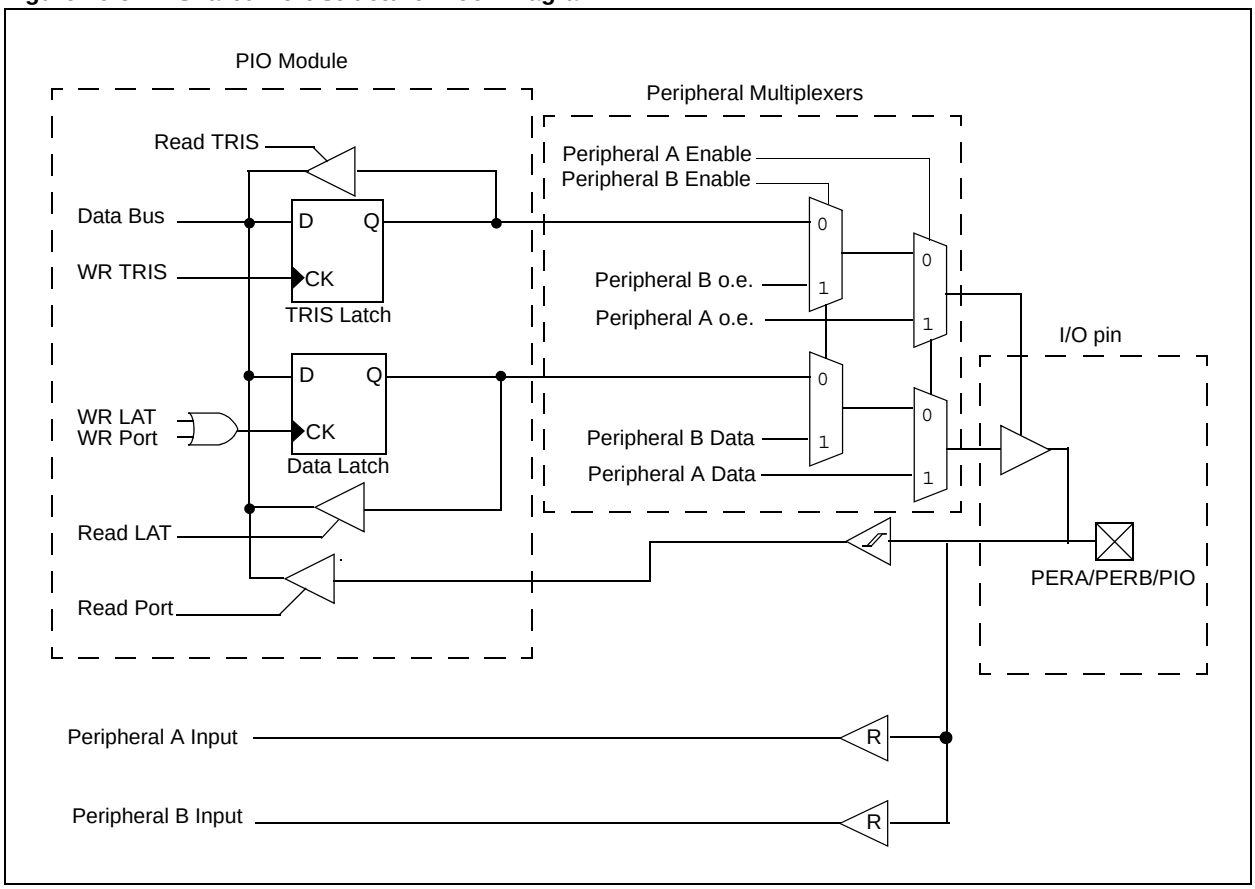
10.3 PERIPHERAL MULTIPLEXING

When a peripheral is enabled, the associated pin output drivers are typically module controlled while a few are user settable. The I/O pin may be read through the input data path, but the output driver for the I/O port bit is generally disabled.

An I/O port that shares a pin with another peripheral is always subservient to the peripheral. The peripheral's output buffer data and control signals are provided to a pair of multiplexers. The multiplexers select whether the peripheral, or the associated port, has ownership of the output data and control signals of the I/O pin. Figure 10-3 shows how ports are shared with other peripherals, and the associated I/O pin to which they are connected.

Note:	Some ports are shared with ADC module pins. The corresponding bits in the AD1PCFG and AD2PCFG registers, if present, must be set to '1' for I/O port functionality, even if the ADC module is turned off.

Figure 10-3: Shared Port Structure Block Diagram



10.3.1 I/O Multiplexing with Multiple Peripherals

For some PIC24H devices, especially those with a small number of I/O pins, multiple peripheral functions may be multiplexed on each I/O pin. Figure 10-3 shows an example of two peripherals multiplexed to the same I/O pin.

The name of the I/O pin defines the priority of each function associated with the pin. Figure 10-3 shows the conceptual I/O pin that has two multiplexed peripherals, Peripheral A and Peripheral B and is named PERA/PERB/PIO.

The I/O pin name is chosen so that the user can easily tell the priority of the functions assigned to the pin. Figure 10-3 is an example showing how Peripheral A has the highest priority for control of the pin. If Peripheral A and Peripheral B are enabled at the same time, Peripheral A will take control of the I/O pins.

10.3.1.1 SOFTWARE INPUT PIN CONTROL

Some of the functions assigned to an I/O pin may be input functions that do not take control of the pin output driver. An example of one such peripheral is the Input Capture module. If the I/O pin associated with the Input Capture is configured as an output, using the appropriate TRIS control bit, the user can manually affect the state of the Input Capture pin through its corresponding PORTx register. This behavior can be useful in some situations, especially for testing purposes, when no external signal is connected to the input pin.

Figure 10-3 shows how the organization of the peripheral multiplexers determine if the peripheral input pin can be manipulated in software using the PORTx register. The conceptual peripherals shown in this figure disconnect the PORTx data from the I/O pin when the peripheral function is enabled.

In general, the following peripherals allow their input pins to be controlled manually through the PORTx registers:

- External Interrupt pins
- Timer Clock Input pins
- Input Capture pins
- PWM Fault pins

Most serial communication peripherals, when enabled, take full control of the I/O pin, so that the input pins associated with the peripheral cannot be affected through the corresponding PORTx registers. These peripherals include the following:

- SPI
- I^2C™
- UART
- ECAN™

Note:	Some peripherals may not be present on all device variants. For further information, refer to the specific device data sheet.

10.3.1.2 PIN CONTROL SUMMARY

When a peripheral is enabled, the associated pin output drivers are typically module controlled while a few are user settable. The term “module control” means that the associated port pin output driver is disabled and the pin can only be controlled and accessed by the peripheral. The term ‘user settable’ means that the associated peripheral port pin output driver is user configurable via the associated TRISx SFR. The TRISx register must be set properly for the peripheral to function properly. For ‘user settable’ peripheral pins, the actual port pin state can always be read via the PORTx SFR.

An Input Capture peripheral makes a good example of a user settable peripheral. The user must write the associated TRIS register to configure the Input Capture pin as an input. Since the I/O pin circuitry is still active when the Input Capture is enabled, the following method can be used to manually produce capture events using software: The Input Capture pin is configured as an output using the associated TRIS register. Then, the software can write values to the corresponding LAT register drive to internally control the Input Capture pin and force capture events.

For example, an INTx pin can be configured as an output and then by writing to the associated LATx bit an INTx interrupt, if enabled, can be generated.

The UART is an example of a Module Control peripheral. When the UART is enabled, the PORTx and TRIS registers have no effect, and cannot be used to write the RX and TX pins. Most communication peripheral functions available on the PIC24H are Module Control peripherals.

For example, the SPI module can be configured for Master mode in which only the SDO pin is required. In this scenario the SDI pin can be configured as a general purpose output pin by clearing (setting to a logic ‘0’) the associated TRISx bit. For more information on how pins can be configured for a module, refer to that specific module section.

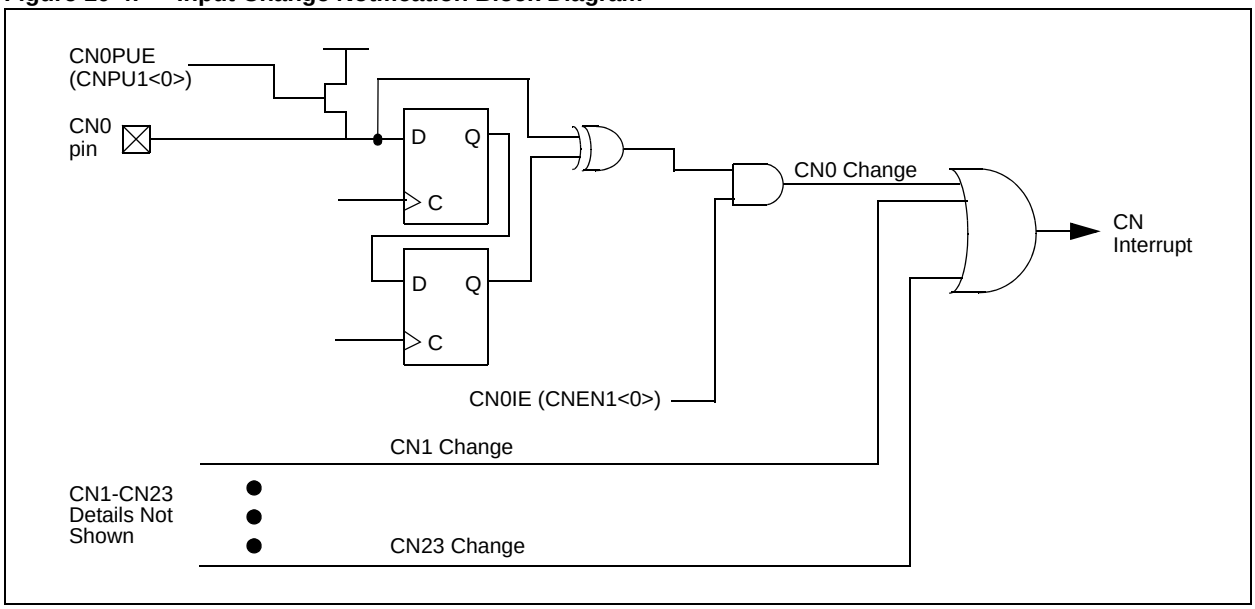
© 2008 Microchip Technology Inc.

10.4 CHANGE NOTIFICATION (CN) PINS

The Change Notification (CN) pins provide PIC24H device family the ability to generate interrupt requests to the processor in response to a change of state on selected input pins. Upto 24 input pins can be selected (enabled) for generating CN interrupts. The total number of available CN inputs is dependent on the selected PIC24H device. For further details, refer to the device data sheet.

Figure 10-4 shows the basic function of the CN hardware.

Figure 10-4: Input Change Notification Block Diagram



10.4.1 CN Control Registers

There are four control registers associated with the CN module. The CNEN1 and CNEN2 registers contain the CNxIE control bits, where ‘x’ denotes the number of the CN input pin. The CNxIE bit must be set for a CN input pin to interrupt the CPU.

The CNPU1 and CNPU2 registers contain the CNxPUE control bits. Each CN pin has a weak pull-up device connected to the pin, which can be enabled or disabled using the CNxPUE control bits. The weak pull-up devices act as a current source that is connected to the pin and eliminate the need for external resistors when push button or keypad devices are connected. For CN pull-up device current specifications, refer to the ‘Electrical Specifications’ section of the device data sheet.

10.4.2 CN Configuration and Operation

The CN pins are configured as follows:

1. Set the associated bit in the TRISx register to ensure that the CN pin is configured as a digital input.
2. Set the appropriate bits in the CNEN1 and CNEN2 registers to enable interrupts for the selected CN pins.
3. Turn on the weak pull-up devices (if desired) for the selected CN pins by setting the appropriate bits in the CNPU1 and CNPU2 registers.
4. Clear the CNIF interrupt flag in the IFSx register.
5. Select the desired interrupt priority for CN interrupts using the CNIP<2:0> control bits in the IPCx register.
6. Enable CN interrupts using the CNIE control bit in the IECx register.

When a CN interrupt occurs, the user should read the PORTx register associated with the CN pin(s). This clears the mismatch condition and sets up the CN logic to detect the next pin change. The current PORTx value can be compared to the PORTx read value obtained at the last CN interrupt to determine the pin that changed.

© 2008 Microchip Technology Inc.

The CN pins have a minimum input pulse width specification. For further details, refer to the 'Electrical Specifications' section of the device data sheet.

Example 10-2: Configuring and Using CN Interrupts

```
void configure CN(void)
{
    CNEN1bits.CN3IE = 1;          // Enable CN3 pin for interrupt detection
    IEC1bits.CNIE = 1;            // Enable CN interrupts
    IFS1bits.CNIF = 0;            // Reset CN interrupt
}

void __attribute__ ((__interrupt__)) _CNInterrupt(void)
{
    // Insert ISR code here

    IFS1bits.CNIF = 0;            // Clear CN interrupt
}
```

10.5 CN OPERATION IN SLEEP AND IDLE MODES

The CN module continues to operate during Sleep or Idle mode. If one of the enabled CN pins changes states, the CNIF status bit in the IFSx register is set. If the CNIE bit is set in the IECx register, the device wakes from Sleep or Idle mode and resumes operation.

If the assigned priority level of the CN interrupt is equal to, or less than, the current CPU priority level, device execution continues from the instruction immediately following the `SLEEP` or `IDLE` instruction.

If the assigned priority level of the CN interrupt is greater than the current CPU priority level, device execution continues from the CN interrupt vector address.

© 2008 Microchip Technology Inc.

10.6 REGISTERS

10.6.1 Change Notification Registers

These registers enable and disable corresponding CN interrupts and pull-up resistors.

- **CNEN1: Input Change Notification Interrupt Enable Register 1**
- **CNEN2: Input Change Notification Interrupt Enable Register 2**
- **CNPU1: Input Change Notification Pull-up Enable Register 1**
- **CNPU2: Input Change Notification Pull-up Enable Register 2**

Register 10-1: CNEN1: Input Change Notification Interrupt Enable Register 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CN15IE	CN14IE	CN13IE	CN12IE	CN11IE	CN10IE	CN9IE	CN8IE
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
CN7IE	CN6IE	CN5IE	CN4IE	CN3IE	CN2IE	CN1IE	CN0IE
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 15-0 **CNxIE:** Input Change Notification Interrupt Enable bits
1 = Enable interrupt on input change
0 = Disable interrupt on input change

Register 10-2: CNEN2: Input Change Notification Interrupt Enable Register 2

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
CN23IE	CN22IE	CN21IE	CN20IE	CN19IE	CN18IE	CN17IE	CN16IE
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 7-0 **CNxIE:** Input Change Notification Interrupt Enable bits
1 = Enable interrupt on input change
0 = Disable interrupt on input change

© 2008 Microchip Technology Inc.

Register 10-3: CNPU1: Input Change Notification Pull-up Enable Register 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CN15PUE	CN14PUE	CN13PUE	CN12PUE	CN11PUE	CN10PUE	CN9PUE	CN8PUE
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
CN7PUE	CN6PUE	CN5PUE	CN4PUE	CN3PUE	CN2PUE	CN1PUE	CN0PUE
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 15-0 **CNxPUE:** Input Change Notification Pull-up Enable bits
1 = Enable pull-up on input change
0 = Disable pull-up on input change

Register 10-4: CNPU2: Input Change Notification Pull-up Enable Register 2

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
CN23PUE	CN22PUE	CN21PUE	CN20PUE	CN19PUE	CN18PUE	CN17PUE	CN16PUE
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 7-0 **CNxPUE:** Input Change Notification Pull-up Enable bits
1 = Enable pull-up on input change
0 = Disable pull-up on input change

© 2008 Microchip Technology Inc.

10.7 RELATED APPLICATION NOTES

This section lists application notes related to this section of the manual. These application notes may not be written specifically for the PIC24H device family, but the concepts are pertinent and could be used with modification and possible limitations. The current application notes related to the I/O Ports are:

Title	Application Note #
Implementing Wake-up on Key Stroke	AN552

Note: For additional Application Notes and code examples for the PIC24H device family, visit the Microchip web site (www.microchip.com).

10.8 REVISION HISTORY

Revision A (February 2007)

This is the initial release of this document.

Revision B (June 2007)

Minor updates were made to this document.

Revision C (September 2008)

This revision incorporates the following content updates:

- Registers:
  - CNPU: Input Change Notification Pull-up Enable Register 2 (see Register 10-4):

    CNxPUE: Input Change Notification Pull-up Enable bits

    1 = Enable pull-up on input change

    0 = Disable pull-up on input change
- Additional minor corrections such as language and formatting updates are incorporated in the entire document.

© 2008 Microchip Technology Inc.