Section 30. I/O Ports with Peripheral Pin Select

HIGHLIGHTS

This section contains these major topics:

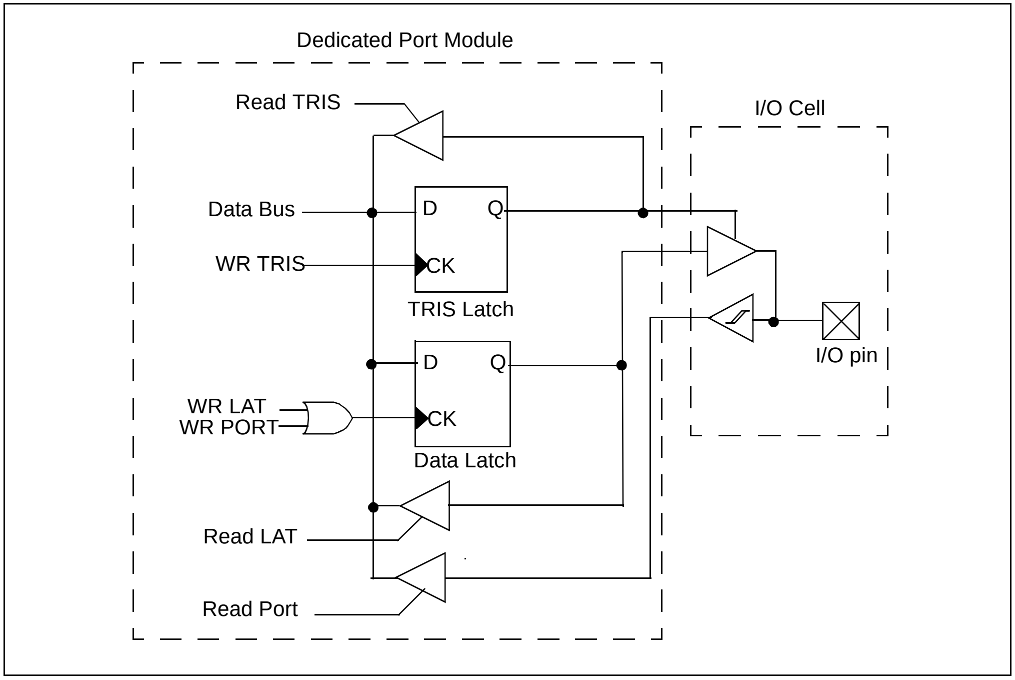
30.1 INTRODUCTION

This section provides information on the I/O ports with peripheral pin select for the PIC24H family of devices. All of the device pins (except VDD, VSS, $\overline{\text{MCLR}}$, 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 will depend on the peripheral features of the device variant. In general, when a peripheral is functioning, that pin may not be used as a general purpose I/O pin.

Figure 30-1 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 30-1: Dedicated Port Structure Block Diagram



30.2 I/O PORT CONTROL REGISTERS

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 will depend on the device variant. In a given device, all of the bits in a port control register may not be implemented. Refer to the specific device data sheet for further details.

30.2.1 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 as an output. An easy way to remember this is that a ‘`1`’ looks like an I (input) and a ‘`0`’ looks like an O (output). All port pins are defined as inputs after a Reset.

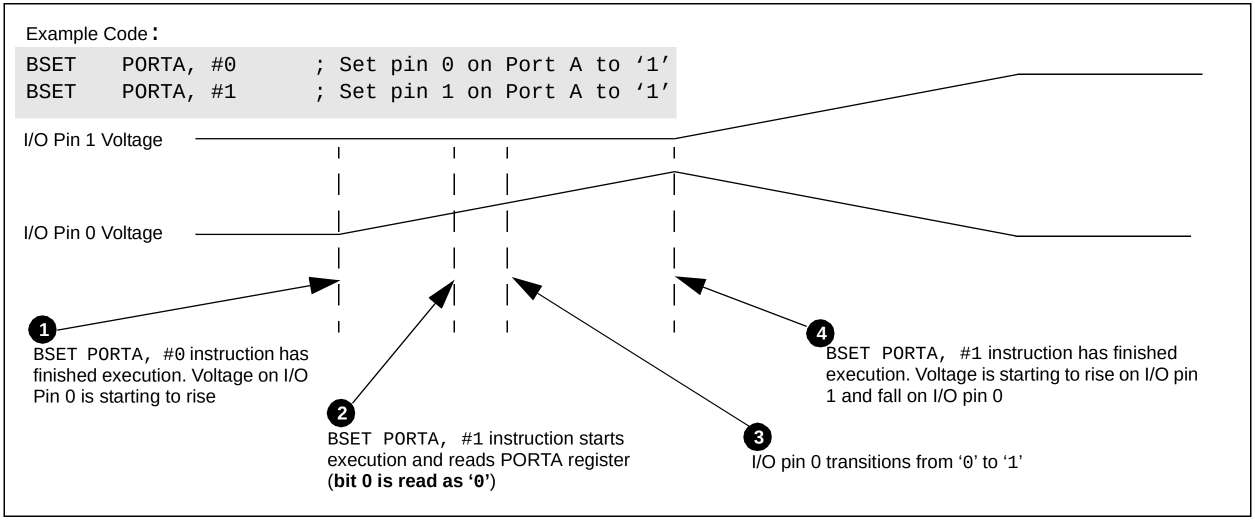
30.2.2 PORT Registers

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 the `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 some 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 30-2 illustrates unintended behavior that occurs if the user application attempts to set I/O bits 0 and 1 on PORTA 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 30-2: Example of Unintended I/O Behavior



Example Code:

```
BSET    PORTA, #0      ; Set pin 0 on Port A to ‘1’
BSET    PORTA, #1      ; Set pin 1 on Port A to ‘1’
```

1. `BSET PORTA, #0` instruction has finished execution. Voltage on I/O Pin 0 is starting to rise

2. `BSET PORTA, #1` instruction starts execution and reads PORTA register (**bit 0 is read as ‘`0`’**)

3. I/O pin 0 transitions from ‘`0`’ to ‘`1`’

4. `BSET PORTA, #1` instruction has finished execution. Voltage is starting to rise on I/O pin 1 and fall on I/O pin 0

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 (see Step 1 in Figure 30-2). However, if the second BSET instruction is executed before the voltage level on pin 0 has reached the threshold for logic 1 (Step 3 in Figure 30-2), the second BSET (read-modified-write) instruction reads ‘0’ for bit 0, which it writes back into the PORTA register (Step 2 in Figure 30-2). 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 level 0 and the voltage on pin 1 to start rising to logic level 1 (Step 4 in Figure 30-2).

30.2.3 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 30-1: Setting IO 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 LATx 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 will be disabled. This means the corresponding LATx and TRISx registers, and the port pin, will read as zeros.

30.2.4 Open-Drain Control Registers

In addition to the PORT, 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.

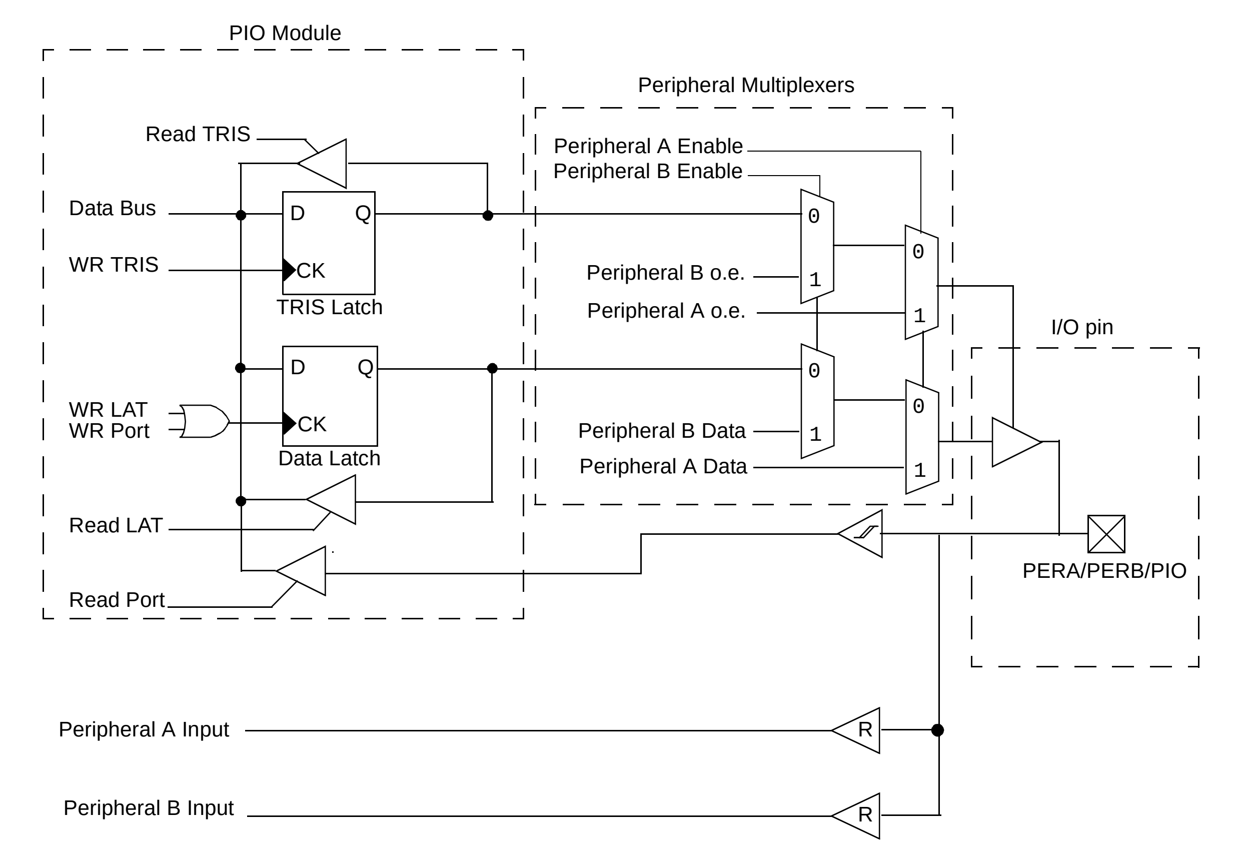
30.3 PERIPHERAL MULTIPLEXING

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 30-3 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 30-3: Shared Port Structure Block Diagram



30.3.1 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 30-3 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. The conceptual I/O pin, shown in Figure 30-3, 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 application can easily determine the priority of the functions assigned to the pin. For the example shown in Figure 30-3, 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.

30.3.1.1 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 PORT register. This behavior can be useful in some situations, especially for testing purposes, when no external signal is connected to the input pin.

Referring to Figure 30-3, the organization of the peripheral multiplexers will determine if the peripheral input pin can be manipulated in software using the PORT register. The conceptual peripherals shown in this figure disconnect the PORT 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 PORT registers:

- External Interrupt pins
- Timer Clock Input pins
- Input Capture pins
- Output Compare 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 PORT registers. These peripherals include the following:

- SPI
- I^{2}C™
- DCI
- UART
- ECAN™

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

30.3.1.2 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 in software via the associated TRISx Special Function Register (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 application 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.

As another 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 PORT and TRIS registers have no effect and cannot be used to read or 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.

30.4 PERIPHERAL PIN SELECT

A major challenge in general-purpose devices is providing the largest possible set of peripheral features while minimizing the conflict of features on I/O pins. The challenge is even greater on low-pin count devices. In an application where more than one peripheral is needed to be assigned to a single pin, inconvenient workarounds in application code or a complete redesign may be the only option.

Peripheral pin select configuration provides an alternative to these choices, by enabling users peripheral set selection and their placement on a wide range of I/O pins. By increasing the pinout options available on a particular device, users can better tailor the microcontroller to their entire application, rather than trimming the application to fit the device.

The peripheral pin select configuration feature operates over a fixed subset of digital I/O pins. Users may independently map the input and/or output of most digital peripherals to any one of these I/O pins. Peripheral pin select is performed in software, and generally does not require the device to be reprogrammed. Hardware safeguards are included that prevent accidental or spurious changes to the peripheral mapping, once it has been established.

30.4.1 Available Pins

The peripheral pin select feature is used with a range of up to 16 pins; the number of available pins is dependent on the particular device and its pin count. Pins that support the peripheral pin select feature include the designation “RPn” in their full pin designation, where “RP” designates a remappable peripheral and “n” is the remappable pin number.

30.4.2 Available Peripherals

The peripherals managed by the peripheral pin select are all digital-only peripherals. These include general serial communications (UART and SPI), general-purpose timer clock inputs, timer-related peripherals (input capture and output compare), and interrupt-on-change inputs.

In comparison, some digital-only peripheral modules are never included in the peripheral pin select feature. This is because the peripheral’s function requires special I/O circuitry on a specific port and cannot be easily connected to multiple pins. These modules include I^{2}C. A similar requirement excludes all modules with analog inputs, such as the A/D converter.

Note:	For a specific list of peripheral pin select supported peripherals, refer to the device data sheet.

A key difference between remappable and non-remappable peripherals is that remappable peripherals are not associated with a default I/O pin. The peripheral must always be assigned to a specific I/O pin before it can be used. In contrast, non-remappable peripherals are always available on a default pin, assuming that the peripheral is active and not conflicting with another peripheral.

When a remappable peripheral is active on a given I/O pin, it takes priority over all other digital I/O and digital communication peripherals associated with the pin. Priority is given regardless of the type of peripheral that is mapped. Remappable peripherals never take priority over any analog functions associated with the pin.

30.4.3 Controlling Peripheral Pin Select

Peripheral pin select features are controlled through two sets of SFRs: one to map peripheral inputs, and one to map outputs. Because they are separately controlled, a particular peripheral’s input and output (if the peripheral has both) can be placed on any selectable function pin without constraint.

The association of a peripheral to a peripheral selectable pin is handled in two different ways, depending on if an input or output is being mapped.

30.4.3.1 INPUT MAPPING

The inputs of the peripheral pin select options are mapped on the basis of the peripheral; that is, a control register associated with a peripheral dictates the pin it will be mapped to. The RPINRx registers are used to configure peripheral input mapping, see Register 30-5 through Register 30-13. Each register contains sets of 5-bit fields, with each set associated with one of the remappable peripherals. Programming a given peripheral's bit field with an appropriate 5-bit value maps the RPn pin with that value to that peripheral. For any given device, the valid range of values for any of the bit field corresponds to the maximum number of peripheral pin selections supported by the device.

For example, Figure 30-4 illustrates remappable pin selection for the U1RX input.

Figure 30-4: Remappable Input for U1RX

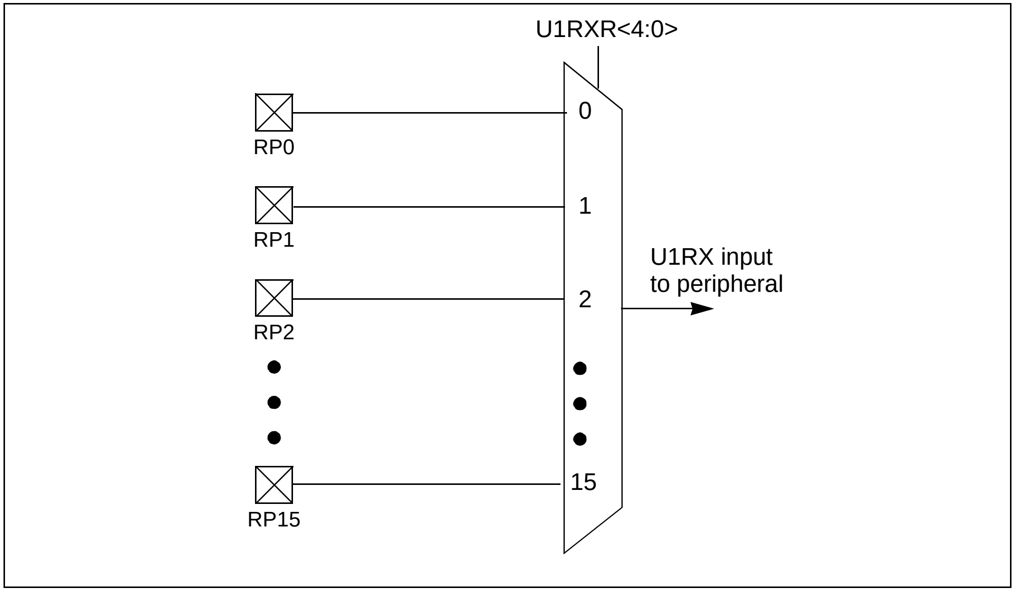


Table 30-1: Selectable Input Sources (Maps Input to Function)

Input Name[1]	Function Name	Register	Configuration Bits
External Interrupt 1	INT1	RPINR0	INT1R<4:0>
External Interrupt 2	INT2	RPINR1	INT2R<4:0>
Timer2 External Clock	T2CK	RPINR3	T2CKR<4:0>
Timer3 External Clock	T3CK	RPINR3	T3CKR<4:0>
Input Capture 1	IC1	RPINR7	IC1R<4:0>
Input Capture 2	IC2	RPINR7	IC2R<4:0>
Input Capture 7	IC7	RPINR10	IC7R<4:0>
Input Capture 8	IC8	RPINR10	IC8R<4:0>
Output Compare Fault A	OCFA	RPINR11	OCFAR<4:0>
UART1 Receive	U1RX	RPINR18	U1RXR<4:0>
UART1 Clear To Send	$\overline{\text{U1CTS}}$	RPINR18	U1CTSR<4:0>
SPI1 Data Input	SDI1	RPINR20	SDI1R<4:0>
SPI1 Clock Input	SCK1	RPINR20	SCK1R<4:0>
SPI1 Slave Select Input	$\overline{\text{SS1}}$	RPINR21	SS1R<4:0>

Note 1: Unless noted, all inputs use the Schmitt input buffers.

Note: Table 30-1 and Figure 30-4 provide examples of selectable input sources for a generic device. Refer to the specific device data sheet for further information.

30.4.3.2 OUTPUT MAPPING

In contrast to inputs, the outputs of the peripheral pin select options are mapped on the basis of the pin; in this case, a control register associated with a particular pin dictates the peripheral output to be mapped. The RPORx registers are used to control output mapping. Like the RPINRx registers, each register contains sets of 5-bit fields, with each set associated with one RPn pin, see Register 30-14 through Register 30-21. The value of the bit field corresponds to one of the peripherals, and that peripheral's output is mapped to the pin, see Table 30-2 and Figure 30-5.

The list of peripherals for output mapping also includes a null value of '00000' because of the mapping technique. This permits any given pin to remain unconnected from the output of any of the pin selectable peripherals.

Figure 30-5: Multiplexing of Remappable Output for RPn

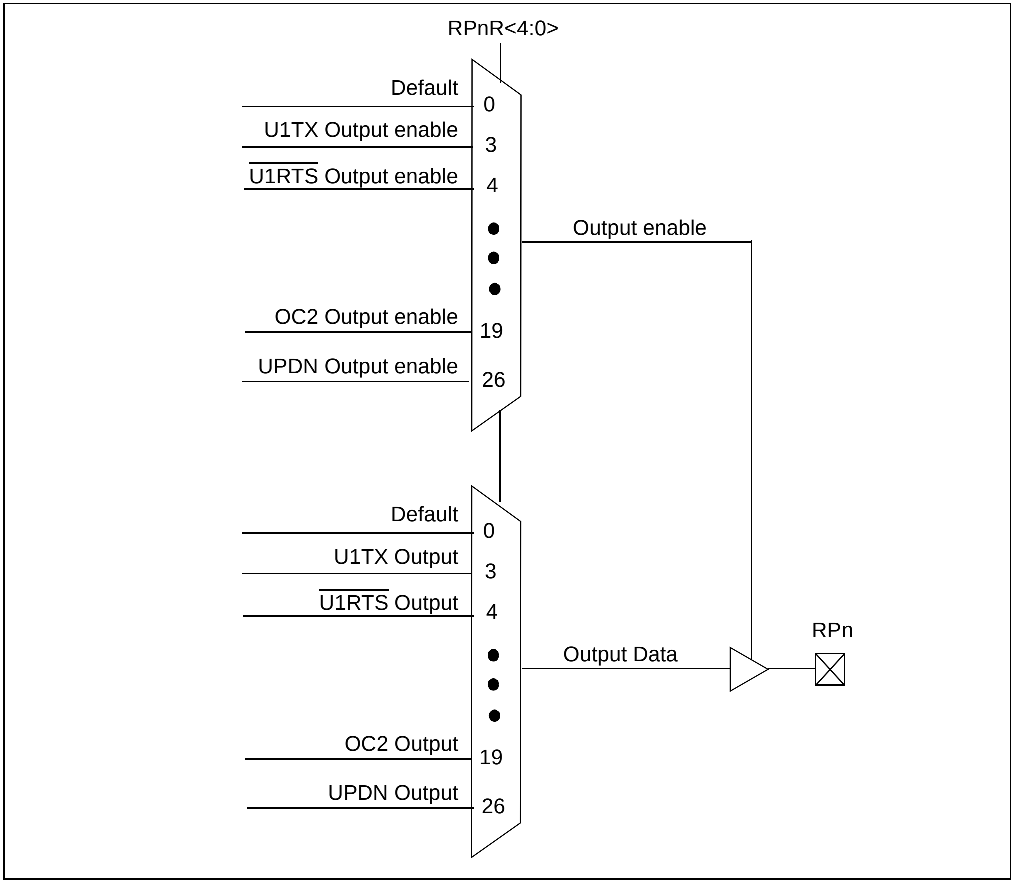


Table 30-2: Output Selection for Remappable Pin (RPn)

Function	RPnR<4:0>	Output Name
NULL	00000	RPn tied to default port pin
U1TX	00011	RPn tied to UART1 Transmit
$\overline{\text{U1RTS}}$	00100	RPn tied to UART1 Ready To Send
SDO1	00111	RPn tied to SPI1 Data Output
SCK1OUT	01000	RPn tied to SPI1 Clock Output
SS1OUT	01001	RPn tied to SPI1 Slave Select Output
OC1	10010	RPn tied to Output Compare 1
OC2	10011	RPn tied to Output Compare 2

Note: Table 30-2 and Figure 30-5 provide examples of selectable output sources for a generic device. Refer to the specific device data sheet for further information.

30.4.3.3 MAPPING LIMITATIONS

The control schema of peripheral select pins is not limited to a small range of fixed peripheral configurations. There are no mutual or hardware-enforced lockouts between any of the peripheral mapping SFRs; literally any combination of peripheral mappings across any or all of the RPn pins is possible. This includes both many-to-one and one-to-many mappings of peripheral inputs and outputs to pins. While such mappings may be technically possible from a configuration point of view, they may not be supportable from an electrical point of view.

30.4.4 Controlling Configuration Changes

Because peripheral remapping can be changed during runtime, some restrictions on peripheral remapping are needed to prevent accidental configuration changes. All PIC24H devices include three features to prevent alterations to the peripheral map:

- Control register lock sequence
- Continuous state monitoring
- Configuration bit remapping lock

30.4.4.1 CONTROL REGISTER LOCK.

Under normal operation, writes to the RPINRx and RPORx registers are not allowed; attempted writes will appear to execute normally, but the contents of the registers will remain unchanged. To change these registers, they must be unlocked in hardware. The register lock is controlled by the IOLOCK bit (OSCCON<6>). Setting IOLOCK prevents writes to the control registers; clearing IOLOCK allows writes.

To set or clear IOLOCK, a specific command sequence must be executed:

1. Write 0x46 to OSCCON<7:0>
2. Write 0x57 to OSCCON<7:0>
3. Clear (or set) IOLOCK as a single operation

IOLOCK remains in one state until changed. This allows all of the peripheral pin selects to be configured with a single unlock sequence followed by an update to all control registers, then locked with a second lock sequence.

> **Note:** MPLAB® C30 provides built-in C language functions for unlocking the OSCCON register.
>
> ```
> __builtin_write_OSCCONL(value)
> __builtin_write_OSCCONH(value)
> ```
>
> See the MPLAB help files for more information.

30.4.4.2 CONTINUOUS STATE MONITORING

In addition to being protected from direct writes, the contents of the RPINRx and RPORx registers are constantly monitored in hardware by shadow registers. If an unexpected change in any of the registers occurs (such as cell disturbances caused by ESD or other external events), a configuration mismatch Reset will be triggered.

30.4.4.3 CONFIGURATION BIT PIN SELECT LOCK

As an additional level of safety, the device can be configured to prevent more than one write session to the RPINRx and RPORx registers. The IOL1WAY (FOSC<5>) configuration bit blocks the IOLOCK bit from being cleared after it has been set once.

In the default (unprogrammed) state, IOL1WAY is set, restricting users to one write session. Programming IOL1WAY allows users unlimited access (with the proper use of the unlock sequence) to the peripheral pin select registers.

30.4.5 Considerations for Peripheral Pin Selection

The ability to control peripheral pin selection introduces several considerations into application design that most users would never think of otherwise. This is particularly true for several common peripherals, which are only available as remappable peripherals.

The main consideration is that the peripheral pin selects are not available on default pins in the device's default (reset) state. More specifically, since all RPINRx and RPORx registers reset to 0000h, this means all peripheral pin select inputs are tied to RP0, while all peripheral pin select outputs are disconnected. This means that before any other application code is executed, the user must initialize the device with the proper peripheral configuration. Since the IOLOCK bit resets in the unlocked state, it is not necessary to execute the unlock sequence after the device has come out of reset. For the sake of application safety, however, it is always a good idea to set IOLOCK and lock the configuration after writing to the control registers.

The unlock sequence must be executed as an assembly-language routine, in the same manner as changes to the oscillator configuration because the unlock sequence is timing-critical. If the bulk of the application is written in C or another high-level language, the unlock sequence should be performed by writing inline assembly.

Choosing the configuration requires the review of all peripheral pin selects and their pin assignments, especially those that will not be used in the application. In all cases, unused pin selectable peripherals should be disabled completely. Unused peripherals should have their inputs assigned to an unused RPn pin function. I/O pins with unused RPn functions should be configured with the null peripheral output.

The assignment of a peripheral to a particular pin does not automatically perform any other configuration of the pin's I/O circuitry. In theory, this means adding a pin selectable output to a pin may mean inadvertently driving an existing peripheral input when the output is driven. Users must be familiar with the behavior of other fixed peripherals that share a remappable pin, and know when to enable or disable them. To be safe, fixed digital peripherals that share the same pin should be disabled when not in use.

Along these lines, configuring a remappable pin for a specific peripheral does not automatically turn that feature on. The peripheral must be specifically configured for operation and enabled, as if it were tied to a fixed pin. Where this happens in the application code (immediately following device reset and peripheral configuration, or inside the main application routine), depends on the peripheral and its use in the application.

A final consideration is that peripheral pin select functions neither override analog inputs, nor reconfigure pins with analog functions for digital I/O. If a pin is configured as an analog input on device reset, it must be explicitly reconfigured as a digital I/O when used with a peripheral pin select.

Example 30-2 shows a configuration for bidirectional communication with flow control using UART1. The following input and output functions are used:

- Input Functions: U1RX, $\overline{\text{U1CTS}}$
- Output Functions: U1TX, $\overline{\text{U1RTS}}$

Example 30-2: Configuring UART1 Input and Output Functions

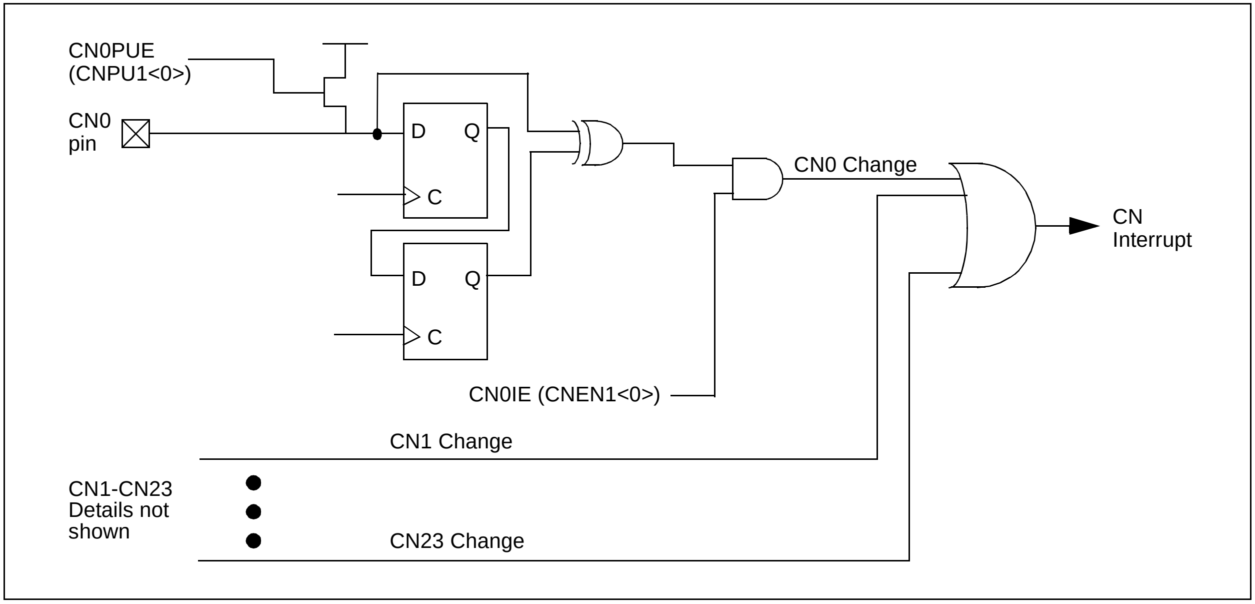
```
//*************************************************************
// Unlock Registers
//*************************************************************
__builtin_write_OSCCONL(OSCCON & ~(1<<6));

//************************************************************
// Configure Input Functions
// (See Table 30-1)
//*************************************************************
//*******************************
// Assign U1Rx To Pin RP0
//***************************
RPINR18bits.U1RXR = 0;

//***************************
// Assign U1CTS To Pin RP1
//***************************
RPINR18bits.U1CTSR = 1;

//*************************************************************
// Configure Output Functions
// (See Table 30-2)
//*************************************************************
//***************************
// Assign U1Tx To Pin RP2
//***************************
RPOR1bits.RP2R = 3;

//***************************
// Assign U1RTS To Pin RP3
//***************************
RPOR1bits.RP3R = 4;

//*************************************************************
// Lock Registers
//*************************************************************
__builtin_write_OSCCONL(OSCCON | (1<<6));
```

30.5 CHANGE NOTIFICATION (CN) PINS

The Change Notification (CN) pins provide PIC24H devices with the ability to generate interrupt requests to the processor in response to a change of state on selected input pins. Up to 24 input pins may be selected (enabled) for generating CN interrupts. The total number of available CN inputs is dependent on the selected PIC24H device. Refer to the device data sheet for further details.

Figure 30-6 shows the basic function of the CN hardware.

Figure 30-6: Input Change Notification Block Diagram



30.5.1 CN Control Registers

There are four control registers associated with the CN module: CNEN1, CNEN2, CNPU1, and CNPU2.

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. Refer to the “Electrical Specifications” section of the device data sheet for CN pull-up device current specifications.

30.5.2 CN Configuration and Operation

The CN pins are configured as follows:

1. Ensure that the CN pin is configured as a digital input by setting the associated bit in the TRISx register.
2. Enable interrupts for the selected CN pins by setting the appropriate bits in the CNEN1 and CNEN2 registers.
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 application should read the PORT register associated with the CN pin(s). This will clear the mismatch condition and set up the CN logic to detect the next pin change. The current PORT value can be compared to the PORT read value obtained at the last CN interrupt to determine the pin that changed.

The CN pins have a minimum input pulse width specification. Refer to the “Electrical Specifications” section of the device data sheet for further details.

Example 30-3: 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
}
```

30.6 CN OPERATION IN SLEEP AND IDLE MODES

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

If the assigned priority level of the CN interrupt is equal to or less than the current CPU priority level, device execution will continue 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 will continue from the CN interrupt vector address.

30.7 REGISTERS

30.7.1 Change Notification Registers

These registers are used to 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**

30.7.2 Peripheral Pin Select Registers

These registers are used to configure input and output functionality of the PIC24H device pins.

- **RPINR0: Peripheral Pin Select Input Register 0**
- **RPINR1: Peripheral Pin Select Input Register 1**
- **RPINR3: Peripheral Pin Select Input Register 3**
- **RPINR7: Peripheral Pin Select Input Register 7**
- **RPINR10: Peripheral Pin Select Input Register 10**
- **RPINR11: Peripheral Pin Select Input Register 11**
- **RPINR18: Peripheral Pin Select Input Register 18**
- **RPINR20: Peripheral Pin Select Input Register 20**
- **RPINR21: Peripheral Pin Select Input Register 21**
- **RPOR0: Peripheral Pin Select Output Register 0**
- **RPOR1: Peripheral Pin Select Output Register 1**
- **RPOR2: Peripheral Pin Select Output Register 2**
- **RPOR3: Peripheral Pin Select Output Register 3**
- **RPOR4: Peripheral Pin Select Output Register 4**
- **RPOR5: Peripheral Pin Select Output Register 5**
- **RPOR6: Peripheral Pin Select Output Register 6**
- **RPOR7: Peripheral Pin Select Output Register 7**

Register 30-1: 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 30-2: CNEN2: Input Change Notification Interrupt Enable Register 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	—	—	—
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

Register 30-3: 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 30-4: CNPU2: Input Change Notification Pull-up Enable Register 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	—	—	—
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 interrupt on input change
0 = Disable interrupt on input change

Register 30-5: RPINR0: Peripheral Pin Select Input Register 0

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	INT1R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **INT1R<4:0>:** Assign External Interrupt 1 (INTR1) to the Corresponding RPn pin
11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

bit 7-0 **Unimplemented:** Read as ‘0’

Register 30-6: RPINR1: Peripheral Pin Select Input Register 1

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	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	INT2R<4:0>				
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 15-5 **Unimplemented:** Read as ‘0’

bit 4-0 **INTR2R<4:0>:** Assign External Interrupt 2 (INTR2) to the Corresponding RPn pin
11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

Register 30-7: RPINR3: Peripheral Pin Select Input Register 3

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	T3CKR<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	T2CKR<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **T3CKR<4:0>:** Assign Timer3 External Clock (T3CK) to the Corresponding RPn pin

11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **T2CKR<4:0>:** Assign Timer2 External Clock (T2CK) to the Corresponding RPn pin

11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

Register 30-8: RPINR7: Peripheral Pin Select Input Register 7

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	IC2R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	IC1R<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **IC2R<4:0>:** Assign Input Capture 2 (IC2) to the Corresponding RPn pin

`11111` = Input tied Vss
`01111` = Input tied to RP15
.
.
.
`00001` = Input tied to RP1
`00000` = Input tied to RP0

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **IC1R<4:0>:** Assign Input Capture 1 (IC1) to the Corresponding RPn pin

`11111` = Input tied Vss
`01111` = Input tied to RP15
.
.
.
`00001` = Input tied to RP1
`00000` = Input tied to RP0

Register 30-9: RPINR10: Peripheral Pin Select Input Register 10

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	IC8R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	IC7R<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **IC8R<4:0>:** Assign Input Capture 8 (IC8) to the corresponding pin RPn pin

11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **IC7R<4:0>:** Assign Input Capture 7 (IC7) to the corresponding pin RPn pin

11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

Register 30-10: RPINR11: Peripheral Pin Select Input Register 11

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	OCFAR<4:0>				
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 15-5 **Unimplemented:** Read as ‘0’

bit 4-0 **OCFAR<4:0>:** Assign Output Capture A (OCFA) to the Corresponding RPn pin

11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

Register 30-11: RPINR18: Peripheral Pin Select Input Register 18

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	U1CTSR<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	U1RXR<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **U1CTSR<4:0>:** Assign UART1 Clear to Send ($\overline{\text{U1CTS}}$) to the Corresponding RPn pin

11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **U1RXR<4:0>:** Assign UART1 Receive (U1RX) to the Corresponding RPn pin

11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

Register 30-12: RPINR20: Peripheral Pin Select Input Register 20

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	SCK1R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	SDI1R<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **SCK1R<4:0>:** Assign SPI1 Clock Input (SCK1IN) to the Corresponding RPn pin

11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **SDI1R<4:0>:** Assign SPI1 Data Input (SDI1) to the Corresponding RPn pin

11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

Register 30-13: RPINR21: Peripheral Pin Select Input Register 21

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	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	SS1R<4:0>				
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 15-15 **Unimplemented:** Read as ‘0’

bit 4-0 **SS1R<4:0>:** Assign SPI1 Slave Select Input (SS1IN) to the Corresponding RPn pin

11111 = Input tied Vss
01111 = Input tied to RP15
.
.
.
00001 = Input tied to RP1
00000 = Input tied to RP0

Register 30-14: RPOR0: Peripheral Pin Select Output Register 0

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP1R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP0R<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **RP1R<4:0>:** Peripheral Output Function is Assigned to RP1 Output Pin (see Table 30-2 for peripheral function numbers)

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **RP0R<4:0>:** Peripheral Output Function is Assigned to RP0 Output Pin (see Table 30-2 for peripheral function numbers)

Register 30-15: RPOR1: Peripheral Pin Select Output Register 1

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP3R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP2R<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **RP3R<4:0>:** Peripheral Output Function is Assigned to RP3 Output Pin (see Table 30-2 for peripheral function numbers)

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **RP2R<4:0>:** Peripheral Output Function is Assigned to RP2 Output Pin (see Table 30-2 for peripheral function numbers)

Register 30-16: RPOR2: Peripheral Pin Select Output Register 2

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP5R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP4R<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **RP5R<4:0>:** Peripheral Output Function is Assigned to RP5 Output Pin (see Table 30-2 for peripheral function numbers)

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **RP4R<4:0>:** Peripheral Output Function is Assigned to RP4 Output Pin (see Table 30-2 for peripheral function numbers)

Register 30-17: RPOR3: Peripheral Pin Select Output Register 3

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP7R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP6R<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **RP7R<4:0>:** Peripheral Output Function is Assigned to RP7 Output Pin (see Table 30-2 for peripheral function numbers)

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **RP6R<4:0>:** Peripheral Output Function is Assigned to RP6 Output Pin (see Table 30-2 for peripheral function numbers)

Register 30-18: RPOR4: Peripheral Pin Select Output Register 4

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP9R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP8R<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **RP9R<4:0>:** Peripheral Output Function is Assigned to RP9 Output Pin (see Table 30-2 for peripheral function numbers)

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **RP8R<4:0>:** Peripheral Output Function is Assigned to RP8 Output Pin (see Table 30-2 for peripheral function numbers)

Register 30-19: RPOR5: Peripheral Pin Select Output Register 5

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP11R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP10R<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **RP11R<4:0>:** Peripheral Output Function is Assigned to RP11 Output Pin (see Table 30-2 for peripheral function numbers)

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **RP10R<4:0>:** Peripheral Output Function is Assigned to RP10 Output Pin (see Table 30-2 for peripheral function numbers)

Register 30-20: RPOR6: Peripheral Pin Select Output Register 6

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP13R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP12R<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **RP13R<4:0>:** Peripheral Output Function is Assigned to RP13 Output Pin (see Table 30-2 for peripheral function numbers)

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **RP12R<4:0>:** Peripheral Output Function is Assigned to RP12 Output Pin (see Table 30-2 for peripheral function numbers)

Register 30-21: RPOR7: Peripheral Pin Select Output Register 7

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP15R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP14R<4:0>				
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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **RP15R<4:0>:** Peripheral Output Function is Assigned to RP15 Output Pin (see Table 30-2 for peripheral function numbers)

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **RP14R<4:0>:** Peripheral Output Function is Assigned to RP14 Output Pin (see Table 30-2 for peripheral function numbers)

30.8 RELATED APPLICATION NOTES

This section lists application notes that are related to this section of the manual. These application notes may not be written specifically for the PIC24H Product 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 with Peripheral Pin Select include the following:

Title	**Application Note #**
Implementing Wake-Up on Key Stroke	AN552

Note: Please visit the Microchip web site (www.microchip.com) for additional application notes and code examples for the PIC24H family of devices.

30.9 REVISION HISTORY

Revision A (February 2007)

This is the initial release of this document.

Revision B (June 2008)

This revision includes the following updates:

- **Example 30-2: "Configuring UART1 Input and Output Functions"** has been updated with entirely new content.
- Additional minor corrections such as language and formatting updates have been incorporated throughout the document.

NOTES: