Section 12. I/O Ports

HIGHLIGHTS

This section of the manual contains the following topics:

12.1 Introduction 12-2
12.2 Control Registers 12-4
12.3 Modes of Operation 12-25
12.4 Interrupts 12-31
12.5 Operation in Power-Saving and DEBUG Modes 12-33
12.6 Effects of Various Resets 12-34
12.7 I/O Port Application 12-35
12.8 I/O Pin Control 12-36
12.9 Design Tips 12-37
12.10 Related Application Notes 12-38
12.11 Revision History 12-39

12.1 INTRODUCTION

The general purpose I/O pins can be considered the simplest of peripherals. They allow the PIC32MX microcontroller to monitor and control other devices. To add flexibility and functionality to a device, some pins are multiplexed with alternate function(s). These functions depend on which peripheral features are on the device. In general, when a peripheral is functioning, that pin may not be used as a general purpose I/O pin.

Following are some of the key features of this module:

- Individual output pin open-drain enable/disable
- Individual input pin pull-up enable/disable
- Monitor select inputs and generate interrupt on mismatch condition
- Operate during CPU SLEEP and IDLE modes
- Fast bit manipulation using CLR, SET and INV registers

A block diagram of a typical I/O port structure is shown in Figure 12-1. The diagram depicts the many peripheral functions that can be multiplexed onto the I/O pin.

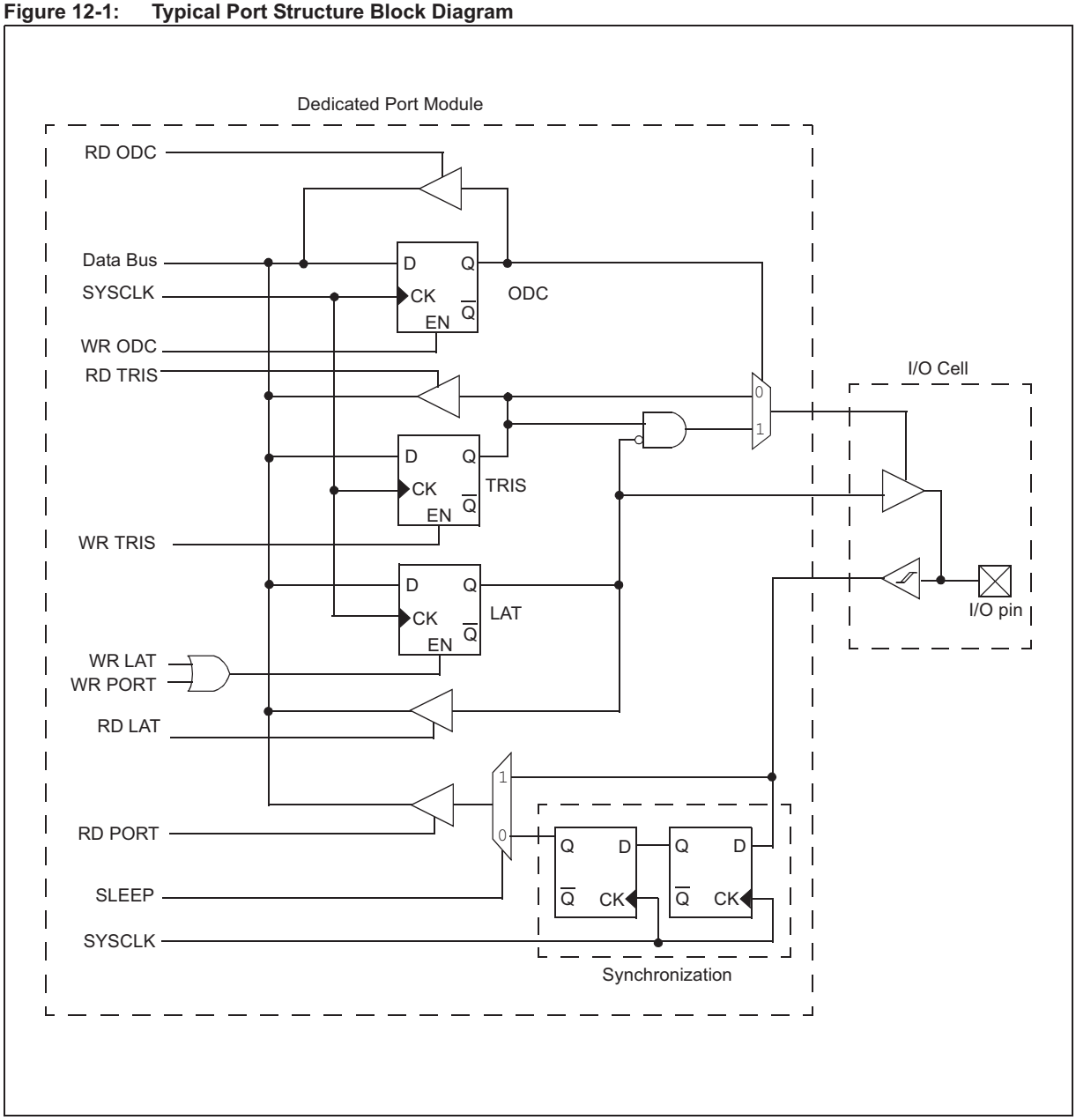
The I/O Ports module consists of the following Special Function Registers (SFRs):

- TRISx: Data Direction register for the module ‘x’
- PORTx: PORT register for the module ‘x’
- LATx: Latch register for the module ‘x’
- ODCx: Open-Drain Control register for the module ‘x’
- CNCON: Interrupt-on-Change Control register
- CNEN: Input Change Notification Interrupt Enable register
- CNPUE: Input Change Notification Pull-up Enable register

The I/O Ports module also has the following associated bits for interrupt control:

- Interrupt Enable Control bits for CN events (CNIE) in INT register IEC1: Interrupt Enable Control Register 1.
- Interrupt Flag Status bits for CN events (CNIF) in INT register IFS1: Interrupt Flag Status Register 1.
- Interrupt Priority Control bits (CNIP<2:0>) in INT register IPC6: Interrupt Priority Control Register 6.

Figure 12-1: Typical Port Structure Block Diagram



12.2 CONTROL REGISTERS

Before reading and writing any I/O port, the desired pin(s) should be properly configured for the application. Each I/O port has three registers directly associated with the operation of the port: TRIS, PORT and LAT. Each I/O port pin has a corresponding bit in these registers. Depending on the PIC32MX device variant, up to seven I/O ports are available. Through out this section, the letter 'x', denotes any or all port module instances. For example "TRISx" would represent TRISA, TRISB, TRISC, etc. Any bit and its associated data and control registers that is not valid for a particular device will be disabled and will read as zeros.

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 might not be available. Refer to the specific device data sheet for further details.

12.2.1 TRIS (tri-state) Registers

TRIS registers configure the data direction flow through port I/O pin(s). The TRIS register bits determine whether a PORT I/O pin is an input or an output.

- A TRIS bit set = 1 configures the corresponding I/O port pin as an input.
- A TRIS bit set = 0 configures the corresponding I/O port pin as an output.
- A read from a TRIS register reads the last value written to the TRIS register.
- All I/O port pins are defined as inputs after a Power-on Reset.

12.2.2 PORT Registers

PORT registers allow I/O pins to be accessed (read).

- A write to a PORT register writes to the corresponding LAT register (PORT data latch). Those I/O port pin(s) configured as outputs are updated.
- A write to a PORT register is the effectively the same as a write to a LAT register.
- A read from a PORT register reads the synchronized signal applied to the port I/O pins.

12.2.3 LAT Registers

LAT registers (PORT data latch) hold data written to port I/O pin(s).

- A write to a LAT register latches data to corresponding port I/O pins. Those I/O port pin(s) configured as outputs are updated.
- A read from LAT register reads the data held in the PORT data latch, not from the port I/O pins.

12.2.4 SET, CLR, INV I/O Port Registers

In addition to the TRIS, PORT, and LAT base registers, each port module is associated with a SET, CLR and INV register which provides atomic bit manipulations and allowing faster I/O pin operations. As the name of the register implies, a value written to a SET, CLR or INV register effectively performs the implied operation, but only on the corresponding base register and only bits specified as '1' are modified. Bits specified as '0' are not modified.

- Writing 0x0001 to TRISASET register sets only bit 0 in base register TRISA
- Writing 0x0020 to PORTDCLR register clears only bit 5 in base register PORTD
- Writing 0x9000 to LATCINV register inverts only bits 15 and 12 in the base register LATC.

Reading SET, CLR and INV registers return an undefined value. To see the affects of a write operation to a SET, CLR or INV register, the base register must be read instead.

The SET, CLR and INV registers are not exclusive to TRIS, PORT and LAT registers. Other I/O port module registers ODC, CNEN and CNPUE also feature these bit manipulation registers.

A typical method to toggle an I/O pin requires a read-modify-write operation performed on a PORT register in software. For example, a read from a PORTx register, mask and modify the desired output bit(s), write the resulting value back to the PORTx register. This method is vulnerable to a read-modify-write issue where the port value may change after it is read and before the modified data can be written back, thus changing the previous state. This method also requires more instructions.

```
PORTA ^= 0x0001;
```

A more efficient and atomic method uses the PORTxINV register. A write to the PORTxINV register effectively performs a read-modify-write operation on the target base register, equivalent to the software operation described above, however, it is done in hardware. To toggle an I/O pin using this method, a "1" is written to the corresponding bit in the PORTxINV register. This operation will read the PORTx register, invert only those bits specified as '1' and write the resulting value to the LATx register, thus toggling the corresponding I/O pin(s) all in a single atomic instruction cycle.

```
PORTAINV = 0x0001;
```


TRISx SET,CLR,INV Register Behavior

- A value written to a TRISxSET register reads the TRISx base register, sets any bit(s) specified as '1', writes the modified value back to the TRISx base register.
- A value written to a TRISxCLR register reads the TRISx base register, clears any bit(s) specified as '1', writes the modified value back to the TRISx base register.
- A value written to a TRISxINV register reads the TRISx base register, inverts any bit(s) specified as '1', writes the modified value back to the TRISx base register.
- Any bit(s), specified as '0', are not modified.

PORTx SET,CLR,INV Register Behavior

- A value written to a PORTxSET register reads the PORTx base register, sets any bit(s) specified as '1', writes the modified value back to the LATx base register. Those I/O port pin(s) configured as outputs are updated.
- A value written to a PORTxCLR register reads the PORTx base register, clears any bit(s) specified as '1', writes the modified value back to the LATx base register. Those I/O port pin(s) configured as outputs are updated.
- A value written to a PORTxINV register reads the PORTx base register, inverts any bit(s) specified as '1', writes the modified value back to the LATx base register. Those I/O port pin(s) configured as outputs are updated.
- Any bit(s), specified as '0', are not modified.

LATx SET,CLR,INV Register Behavior

- A value written to a LATxSET register reads the LATx base register, sets any bit(s) specified as '1', writes the modified value back to the LATx base register. Those I/O port pin(s) configured as outputs are updated.
- A value written to a LATxCLR register reads the LATx base register, clears any bit(s) specified as '1', writes the modified value back to the LATx base register. Those I/O port pin(s) configured as outputs are updated.
- A value written to a LATxINV register reads the LATx base register, inverts any bit(s) specified as '1', writes the modified value back to the LATx base register. Those I/O port pin(s) configured as outputs are updated.

Any bit(s), specified as '0', are not modified.

12.2.5 ODC Registers

Each I/O pin can be individually configured for either normal digital output or open-drain output. This is controlled by the Open-Drain Control register, ODCx, associated with each I/O pin. If the ODC bit for an I/O pin is a '1', then the pin acts as an open-drain output. If the ODC bit for an I/O pin is a '0', then the pin is configured for a normal digital output (ODC bit is valid only for output pins). After a Reset, the status of all the bits of the ODCx register is set to '0'.

The open-drain feature allows the generation of outputs higher than VDD on any desired digital only pins by using external pull-up resistors. The maximum open-drain voltage allowed is the same as the maximum VIH specification. The ODC register setting takes effect in all the I/O modes, allowing the output to behave as an open-drain even if a peripheral is controlling the pin. Although the user could achieve the same effect by manipulating the corresponding LAT and TRIS bits, this procedure will not allow the peripheral to operate in Open-Drain mode (except for the default operation of the I2C™ pins). Since I2C pins are already open-drain pins, the ODCx settings do not affect the I2C pins. Also, the ODCx settings do not affect the JTAG output characteristics as the JTAG scan cells are inserted between the ODCx logic and the I/O.

12.2.6 CN Control Registers

Several I/O pins may be individually configured to generate an interrupt when a change on an input pin is detected. There are three control registers associated with the CN (Change Notice) module. The CNCON control register is used to enable or disable the CN module. The CNEN register contains the CNENx control bits, where 'x' denotes the number of the CN input pin. The CNPUE register contains the CNPUEx control bits. Each CN pin has a pull-up device connected to the pin which can be enabled or disabled using the CNPUEx control bits. The 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 Characteristics"** section of the specific device data sheet for CN pull-up device current specifications.

The following table provides a brief summary of all I/O ports-related registers. Corresponding registers appear after the summary, followed by a detailed description of each register.

Table 12-1: I/O Ports SFR Summary

Name		Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
TRISx	31:0	—	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	—
	15:8	TRISx<15:8>							
	7:0	TRISx<7:0>							
TRISxCLR	31:0	Write clears selected bits in TRISx, read yields undefined value							
TRISxSET	31:0	Write sets selected bits in TRISx, read yields undefined value							
TRISxINV	31:0	Write inverts selected bits in TRISx, read yields undefined value							
PORTx	31:0	—	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	—
	15:8	PORTx<15:8>							
	7:0	PORTx<7:0>							
PORTxCLR	31:0	Write clears selected bits in LATx, read yields undefined value							
PORTxSET	31:0	Write sets selected bits in LATx, read yields undefined value							
PORTxINV	31:0	Write inverts selected bits in LATx, read yields undefined value							
LATx	31:24	—	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	—
	15:8	LATx<15:8>							
	7:0	LATx<7:0>							
LATxCLR	31:0	Write clears selected bits in LATx, read yields undefined value							
LATxSET	31:0	Write sets selected bits in LATx, read yields undefined value							
LATxINV	31:0	Write inverts selected bits in LATx, read yields undefined value							
ODCx	31:24	—	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	—
	15:8	ODCx<5:8>							
	7:0	ODCx<7:0>							
ODCxCLR	31:0	Write clears selected bits in ODCx, read yields undefined value							
ODCxSET	31:0	Write sets selected bits in ODCx, read yields undefined value							
ODCxINV	31:0	Write inverts selected bits in ODCx, read yields undefined value							

Table 12-1: I/O Ports SFR Summary (Continued)

Name		Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
CNCON	31:24	—	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	—
	15:8	ON	FRZ	SIDL	—	—	—	—	—
	7:0	—	—	—	—	—	—	—	—
CNCONCLR	31:0	Write clears selected bits in CNCON, read yields undefined value							
CNCONSET	31:0	Write sets selected bits in CNCON, read yields undefined value							
CNCONINV	31:0	Write inverts selected bits in CNCON, read yields undefined value							
CNEN	31:24	—	—	—	—	—	—	—	—
	23:16	—	—	CNEN<21:16>					
	15:8	CNEN<15:8>							
	7:0	CNEN<7:0>							
CNENCLR	31:0	Write clears selected bits in CNEN, read yields undefined value							
CNENSET	31:0	Write sets selected bits in CNEN, read yields undefined value							
CNENINV	31:0	Write inverts selected bits in CNEN, read yields undefined value							
CNPUE	31:24	—	—	—	—	—	—	—	—
	23:16	—	—	CNPUE<21:16>					
	15:8	CNPUE<15:8>							
	7:0	CNPUE<7:0>							
CNPUECLR	31:0	Write clears selected bits in CNPUE read yields undefined value							
CNPUESET	31:0	Write sets selected bits in CNPUE, read yields undefined value							
CNPUEINV	31:0	Write inverts selected bits in CNPUE, read yields undefined value							
IEC1	31:24	—	—	—	—	—	—	USBIE	FCEIE
	23:16	—	—	—	—	DMA3IE	DMA2IE	DMA1IE	DMA0IE
	15:8	RTCCIE	FSCMIE	I2C2MIE	I2C2SIE	I2C2BIE	U2TXIE	U2RXIE	U2EIE
	7:0	SPI2RXIE	SPI2TXIE	SPI2EIE	CMP2IE	CMP1IE	PMPIE	AD1IE	CNIE
IFS1	31:24	—	—	—	—	—	—	USBIF	FCEIF
	23:16	—	—	—	—	DMA3IF	DMA2IF	DMA1IF	DMA0IF
	15:8	RTCCIF	FSCMIF	I2C2MIF	I2C2SIF	I2C2BIF	U2TXIF	U2RXIF	U2EIF
	7:0	SPI2RXIF	SPI2TXIF	SPI2EIF	CMP2IF	CMP1IF	PMPIF	AD1IF	CNIF
IPC6	31:24	—	—	—	AD1IP<2:0>			AD1IS<1:0>	
	23:16	—	—	—	CNIP<2:0>			CNIS<1:0>	
	15:8	—	—	—	I2C1IP<2:0>			I2C1IS<1:0>	
	7:0	—	—	—	U1IP<2:0>			U1IS<1:0>	

Register 12-1: TRISx: TRIS Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31							bit 24

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 23							bit 16

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TRIS<15:8>							
bit 15							bit 8

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TRIS<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 31-16 **Reserved:** Write ‘0’; ignore read

bit 15-0 **TRISx<15:0>:** TRIS Register bits

1 = Corresponding port pin “Input”

0 = Corresponding port pin ‘Output”

Register 12-2: TRISxCLR: TRIS Clear Register

Write clears selected bits in TRISx, read yields undefined value	
bit 31	bit 0

bit 31-0 **Clears selected bits in TRISx**

A write of '1' in one or more bit positions clears the corresponding bit(s) in TRISx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `TRISxCLR = 0x00008001` will clear bits 15 and 0 in TRISx register.

Register 12-3: TRISxSET: TRIS Set Register

Write sets selected bits in TRISx, read yields undefined value	
bit 31	bit 0

bit 31-0 **Sets selected bits in TRISx**

A write of '1' in one or more bit positions sets the corresponding bit(s) in TRISx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `TRISxSET = 0x00008001` will set bits 15 and 0 in TRISx register.

Register 12-4: TRISxINV: TRIS Invert Register

Write inverts selected bits in TRISx, read yields undefined value	
bit 31	bit 0

bit 31-0 **Inverts selected bits in TRIS**

A write of '1' in one or more bit positions inverts the corresponding bit(s) in TRISx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `TRISxINV = 0x00008001` will invert bits 15 and 0 in TRISx register.

Register 12-5: PORTx: PORT Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31							bit 24

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 23							bit 16

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
PORT<15:8>							
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
PORT<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 31-16 **Reserved:** Write ‘0’; ignore read

bit 15-0 **PORTx<15:0>:** PORT Register bits
Read = Value on port pins
Write = Value written to the LATx register, PORT latch and I/O pins

Register 12-6: PORTxCLR: PORT Clear Register

Write clears selected bits in LATx, read yields undefined value
bit 31 bit 0

bit 31-0 **Clears selected bits in LATx**

A write of '1' in one or more bit positions clears the corresponding bit(s) in LATx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `PORTxCLR = 0x00008001` will clear bits 15 and 0 in LATx register.

Register 12-7: PORTxSET: PORT Set Register

Write sets selected bits in LATx, read yields undefined value
bit 31 bit 0

bit 31-0 **Sets selected bits in** LA**Tx**

A writes of '1' in one or more bit positions sets the corresponding bit(s) in LATx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `PORTxSET = 0x00008001` will set bits 15 and 0 in LATx register.

Register 12-8: PORTxINV: PORT Invert Register

Write inverts selected bits in LATx, read yields undefined value
bit 31 bit 0

bit 31-0 **Inverts selected bits in** LA**Tx**

A write of '1' in one or more bit positions inverts the corresponding bit(s) in LATx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `PORTxINV = 0x00008001` will invert bits 15 and 0 in LATx register.

Register 12-9: LATx: LAT Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31							bit 24

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 23							bit 16

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-1x	R/W-x	R/W-x
LAT<15:8>							
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
LAT<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 31-16 **Reserved:** Write ‘0’; ignore read

bit 15-0 **LATx<15:0>:** LAT Register bits
Read = Value on PORT latch, not I/O pins
Write = Value written to PORT latch and I/O pins

Register 12-10: LATxCLR: LAT Clear Register

Write clears selected bits in LATx, read yields undefined value	
bit 31	bit 0

bit 31-0 **Clears selected bits in LATx**

A write of '1' in one or more bit positions clears the corresponding bit(s) in LATx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `LATxCLR = 0x00008001` will clear bits 15 and 0 in LATx register.

Register 12-11: LATxSET: LAT Set Register

Write sets selected bits in LATx, read yields undefined value	
bit 31	bit 0

bit 31-0 **Sets selected bits in LATx**

A write of '1' in one or more bit positions sets the corresponding bit(s) in LATx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `LATxSET = 0x00008001` will set bits 15 and 0 in LATx register.

Register 12-12: LATxINV: LAT Invert Register

Write inverts selected bits in LATx, read yields undefined value	
bit 31	bit 0

bit 31-0 **Inverts selected bits in LATx**

A write of '1' in one or more bit positions inverts the corresponding bit(s) in LATx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `LATxINV = 0x00008001` will invert bits 15 and 0 in LATx register.

Register 12-13: ODCx: Open Drain Configuration Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31							bit 24

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ODCx<15:8>							
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
ODCx<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 31-16 **Reserved:** Write ‘0’; ignore read

bit 15-0 **ODCx<15:0>:** ODCx Register bits
If a port pin is configured as an output (corresponding TRISx bit = 0)
1 = Port pin open-drain output enabled
0 = Port pin open-drain output disabled
If a port pin is configured as an input, ODCx bits have no effect.

Register 12-14: ODCxCLR: Open Drain Configuration Clear Register

Write clears selected bits in ODCx, read yields undefined value	
bit 31	bit 0

bit 31-0 **Clears selected bits in ODCx**

A write of '1' in one or more bit positions clears the corresponding bit(s) in ODCx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `ODCxCLR = 0x00008001` will clear bits 15 and 0 in ODCx register.

Register 12-15: ODCxSET: Open Drain Configuration Set Register

Write sets selected bits in ODCx, read yields undefined value	
bit 31	bit 0

bit 31-0 **Sets selected bits in ODCx**

A write of '1' in one or more bit positions sets the corresponding bit(s) in ODCx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `ODCxSET = 0x00008001` will set bits 15 and 0 in ODCx register.

Register 12-16: ODCxINV: Open Drain Configuration Invert Register

Write inverts selected bits in ODCx, read yields undefined value	
bit 31	bit 0

bit 31-0 **Inverts selected bits in ODCx**

A write of '1' in one or more bit positions inverts the corresponding bit(s) in ODCx register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `ODCxINV = 0x00008001` will invert bits 15 and 0 in ODCx register.

Register 12-17: CNCON: Interrupt-On-Change Control Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31							bit 24

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 23							bit 16

R/W-0	R/W-0	R/W-0	r-x	r-x	r-x	r-x	r-x
ON	FRZ	SIDL	—	—	—	—	—
bit 15							bit 8

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 31-16 **Reserved:** Write ‘0’; ignore read

bit 15 **ON:** Change Notice Module On bit
1 = CN Module is enabled
0 = CN Module is disabled

Note: When using 1:1 PBCLK divisor, the user’s software should not read/write the peripheral’s SFRs in the SYSCLK cycle immediately following the instruction that clears the module’s ON bit.

bit 14 **FRZ:** Freeze in Debug Exception Mode bit
1 = Freeze operation when CPU is in Debug Exception mode
0 = Continue operation when CPU is in Debug Exception mode

Note: FRZ is writable in Debug Exception mode only, it is forced to ‘0’ in normal mode.

bit 13 **SIDL:** Stop in IDLE Mode bit
1 = Discontinue operation when device enters IDLE mode
0 = Continue operation in IDLE mode

bit 12-0 **Reserved:** Write ‘0’; ignore read

Register 12-18: CNCONCLR: Interrupt-On-Change Control Clear Register

Write clears selected bits in CNCON, read yields undefined value
bit 31 bit 0

bit 31-0 **Clears selected bits in CNCON**

A write of '1' in one or more bit positions clears the corresponding bit(s) in CNCON register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `CNCONCLR = 0x00008000` will clear bit 15 in CNCON register.

Register 12-19: CNCONSET: Interrupt-On-Change Control Set Register

Write sets selected bits in CNCON, read yields undefined value
bit 31 bit 0

bit 31-0 **Sets selected bits in CNCON**

A write of '1' in one or more bit positions sets the corresponding bit(s) in CNCON register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `CNCONSET = 0x00008000` will set bit 15 in CNCON register.

Register 12-20: CNCONINV: Interrupt-On-Change Control Invert Register

Write inverts selected bits in CNCON, read yields undefined value
bit 31 bit 0

bit 31-0 **Inverts selected bits in CNCON**

A write of '1' in one or more bit positions inverts the corresponding bit(s) in CNCON register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `CNCONINV = 0x00008000` will invert bit 15 in CNCON register.

Register 12-21: CNEN: Input Change Notification Interrupt Enable Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31							bit 24

r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CNEN21	CNEN20	CNEN19	CNEN18	CNEN17	CNEN16
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CNEN15	CNEN14	CNEN13	CNEN12	CNEN11	CNEN10	CNEN9	CNEN8
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
CNEN7	CNEN6	CNEN5	CNEN4	CNEN3	CNEN2	CNEN1	CNEN0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 31-22 **Reserved:** Write ‘0’; ignore read

bit 21-0 **CNEN<21:0>:** CNEN Register bits

If a port pin is configured as an input (corresponding TRISx bit = 1)
1 = Port pin input change notice enabled
0 = Port pin input change notice disabled
If a port pin is configured as an output, CNENx bits have no effect.

Register 12-22: CNENCLR: Input Change Notification Interrupt Enable Register Clear Register

Write clears selected bits in CNEN, read yields undefined value
bit 31 bit 0

bit 31-0 **Clears selected bits in CNEN**

A write of '1' in one or more bit positions clears the corresponding bit(s) in CNEN register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `CNENCLR = 0x00008001` will clear bits 15 and 0 in CNEN register.

Register 12-23: CNENSET: Input Change Notification Interrupt Enable Register Set Register

Write sets selected bits in CNEN, read yields undefined value
bit 31 bit 0

bit 31-0 **Sets selected bits in CNEN**

A write of '1' in one or more bit positions sets the corresponding bit(s) in CNEN register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `CNENSET = 0x00008001` will set bits 15 and 0 in CNEN register.

Register 12-24: CNENINV: Input Change Notification Interrupt Enable Register Invert Register

Write inverts selected bits in CNEN, read yields undefined value
bit 31 bit 0

bit 31-0 **Inverts selected bits in CNEN**

A write of '1' in one or more bit positions inverts the corresponding bit(s) in CNEN register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `CNENINV = 0x00008001` will invert bits 15 and 0 in CNEN register.

Register 12-25: CNPUE: Input Change Notification Pull-up Enable Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31							bit 24

r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CNPUE21	CNPUE20	CNPUE19	CNPUE18	CNPUE17	CNPUE16
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CNPUE15	CNPUE14	CNPUE13	CNPUE12	CNPUE11	CNPUE10	CNPUE9	CNPUE8
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
CNPUE7	CNPUE6	CNPUE5	CNPUE4	CNPUE3	CNPUE2	CNPUE1	CNPUE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 31-22 **Reserved:** Write ‘0’; ignore read

bit 21-0 **CNPUE<21:0>:** CNPUE Register bits

If a port pin is configured as an input (corresponding TRISx bit = 1)

1 = port pin pull-up enabled

0 = port pin pull-up disabled

If a port pin is configured as an output, the corresponding CNPUEx bit should be disabled.

Register 12-26: CNPUECLR: Interrupt Change Pull-up Enable Clear Register

Write clears selected bits in CNPUE, read yields undefined value	
bit 31	bit 0

bit 31-0 **Clears selected bits in CNPUE**

A write of '1' in one or more bit positions clears the corresponding bit(s) in CNPUE register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `CNPUECLR = 0x00008001` will clear bits 15 and 0 in CNPUE register.

Register 12-27: CNPUESET: Interrupt Change Pull-up Enable Set Register

Write sets selected bits in CNPUE, read yields undefined value	
bit 31	bit 0

bit 31-0 **Sets selected bits in CNPUE**

A write of '1' in one or more bit positions sets the corresponding bit(s) in CNPUE register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `CNPUESET = 0x00008001` will set bits 15 and 0 in CNPUE register.

Register 12-28: CNPUEINV: Interrupt Change Pull-up Enable Invert Register

Write inverts selected bits in CNPUE, read yields undefined value	
bit 31	bit 0

bit 31-0 **Inverts selected bits in CNPUE**

A write of '1' in one or more bit positions inverts the corresponding bit(s) in CNPUE register and does not affect unimplemented or read-only bits. A write of '0' will not affect the register.

Example: `CNPUEINV = 0x00008001` will invert bits 15 and 0 in CNPUE register.

Register 12-29: IEC1: Interrupt Enable Control Register 1(1)

r-x	r-x	r-x	r-x	r-x	r-x	R/W-0	R/W-0
—	—	—	—	—	—	USBIE	FCEIE
bit 31							bit 24

r-x	r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	DMA3IE	DMA2IE	DMA1IE	DMA0IE
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RTCCIE	FSCMIE	I2C2MIE	I2C2SIE	I2C2BIE	U2TXIE	U2RXIE	U2EIE
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
SPI2RXIE	SPI2TXIE	SPI2EIE	CMP2IE	CMP1IE	PMPIE	AD1IE	CNIE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 0 **CNIE:** Change Notice Interrupt Enable bit
1 = Interrupt is enabled
0 = Interrupt is disabled

Note 1: Shaded bit names in this Interrupt register control other PIC32MX peripherals and are not related to the I/O Port Change Notice.

Register 12-30: IFS1: Interrupt Flag Status Control Register 1(1)

r-x	r-x	r-x	r-x	r-x	r-x	R/W-0	R/W-0
—	—	—	—	—	—	USBIF	FCEIF
bit 31							bit 24

r-x	r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	DMA3IF	DMA2IF	DMA1IF	DMA0IF
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RTCCIF	FSCMIF	I2C2MIF	I2C2SIF	I2C2BIF	U2TXIF	U2RXIF	U2EIF
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
SPI2RXIF	SPI2TXIF	SPI2EIF	CMP2IF	CMP1IF	PMPIF	AD1IF	CNIF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 0 **CNIE:** Change Notice Interrupt Enable bit
1 = Interrupt is enabled
0 = Interrupt is disabled

Note 1: Shaded bit names in this Interrupt register control other PIC32MX peripherals and are not related to the I/O Port Change Notice.

Register 12-31: IPC6: Interrupt Priority Control Register 6(1)

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	AD1IP<2:0>			AD1IS<1:0>	
bit 31							bit 24

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	CNIP<2:0>			CNIS<1:0>	
bit 23							bit 16

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	I2C1IP<2:0>			I2C1IS<1:0>	
bit 15							bit 8

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	U1IP<2:0>			U1IS<1:0>	
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 20-18 **CNIP<2:0>:** Change Notice Interrupt Priority bits
111 = Interrupt Priority is 7
110 = Interrupt Priority is 6
101 = Interrupt Priority is 5
100 = Interrupt Priority is 4
011 = Interrupt Priority is 3
010 = Interrupt Priority is 2
001 = Interrupt Priority is 1
000 = Interrupt is disabled

bit 17-16 **CNIS<1:0>:** Change Notice Interrupt Subpriority bits
11 = Interrupt Subpriority is 3
10 = Interrupt Subpriority is 2
01 = Interrupt Subpriority is 1
00 = Interrupt Subpriority is 0

Note 1: Shaded bit names in this Interrupt register control other PIC32MX peripherals and are not related to the I/O Port Change Notice.

12.3 MODES OF OPERATION

12.3.1 Digital Inputs

Pins are configured as digital inputs by setting the corresponding TRIS register bits = 1. When configured as inputs, they are Schmitt Triggers. Several digital pins share functionality with analog inputs and default to the analog inputs at Power-on Reset. Setting the corresponding bit in the AD1PCFG register = 1 enables the pin as a digital pin.

Note:	Refer to the specific device data sheet for further details regarding input buffer types. To use pins that are multiplexed with the 10-bit Analog-to-Digital Converter (A/D) module for digital I/O, the corresponding bits in the AD1PCFG register must be set to '1' – even if the A/D module is turned off.

12.3.2 Analog Inputs

Certain pins can be configured as analog inputs used by the A/D and Comparator modules. Setting the corresponding bits in the AD1PCFG register = 0 enables the pin as an analog input pin, independent of the TRIS register setting for the corresponding pin.

12.3.3 Digital Outputs

Pins are configured as digital outputs by setting the corresponding TRIS register bits = 0. When configured as digital outputs, these pins are CMOS drivers or can be configured as open-drain outputs by setting the corresponding bits in the ODC register.

12.3.4 Analog Outputs

Certain pins can be configured as analog outputs, such as the CVREF output voltage used in the Comparator module. Configuring the Comparator module to provide this output will present the analog output voltage on the pin, independent of the TRIS register setting for the corresponding pin.

Note:	Refer to the specific device data sheet for further details regarding the use of A\D and Comparator modules.

12.3.5 Open-Drain Configuration

In addition to the PORT, LAT and TRIS registers for data control, each port pin configured as a digital output can also select between an active drive output and open-drain output. This is controlled by the Open-Drain Control register, ODCx, associated with each port. From Power-on Reset, when an I/O pin is configured as a digital output, its output is active drive by default. Setting a bit in the ODCx register = 1 configures the corresponding pin as an open-drain output.

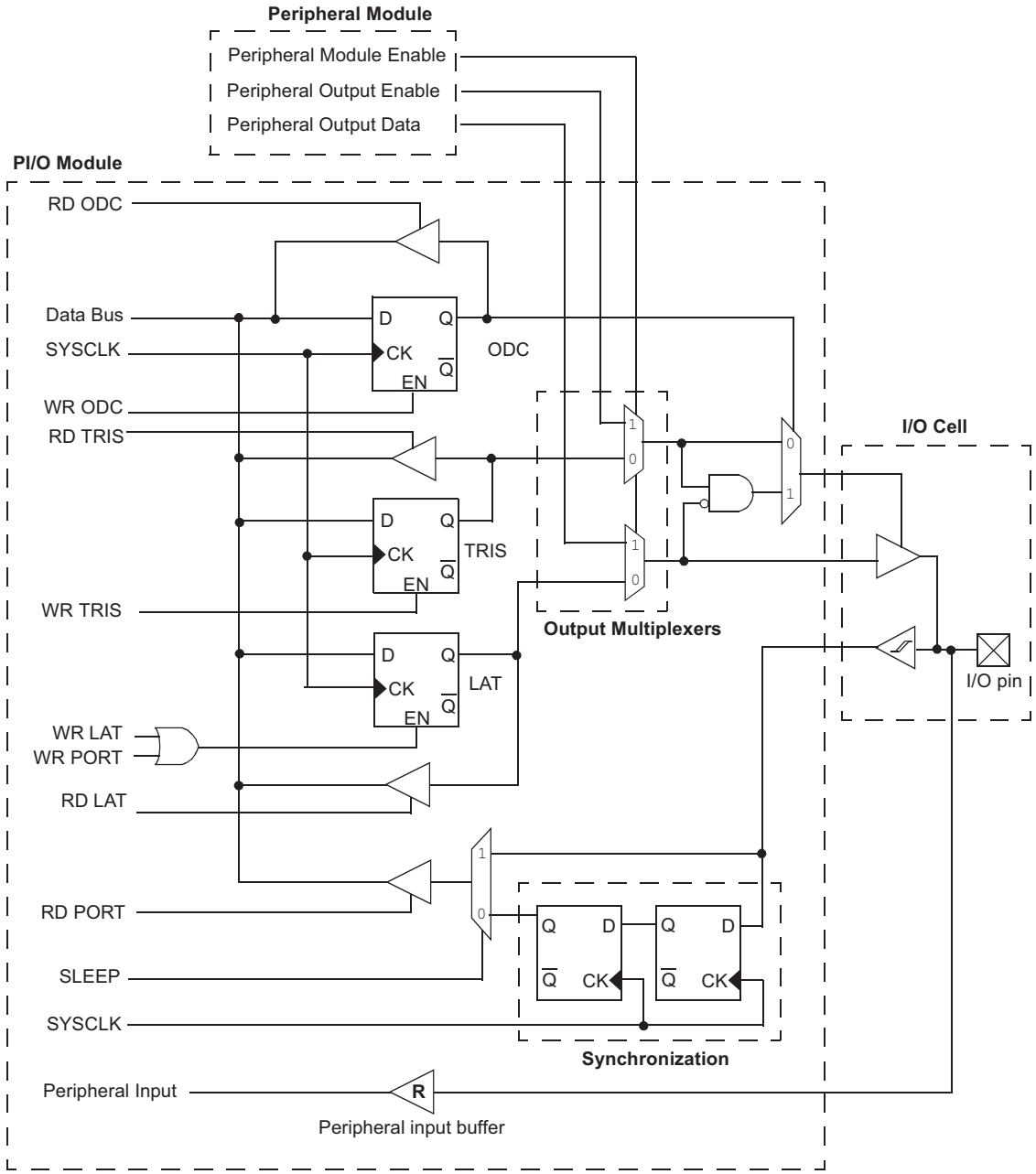
The open-drain feature allows the generation of outputs higher than V_{DD} (e.g., 5V) on any desired digital-only pins by using external pull up resistors. The maximum open-drain voltage allowed is the same as the maximum V_{IH} specification.

12.3.6 Peripheral Multiplexing

Many pins also support one or more peripheral modules. When configured to operate with a peripheral, a pin may not be used for general input or output. In many cases, a pin must still be configured for input or output, although some peripherals override the TRIS configuration. Figure 12-2 shows how ports are shared with other peripherals, and the associated I/O pin to which they are connected. For some PIC32MX devices, multiple peripheral functions may be multiplexed on each I/O pin. The priority of the peripheral function depends on the order of the pin description in the pin diagram of the specific product data sheet.

Note that the output of a pin can be controlled by the TRISx register bit or, in some cases, by the peripheral itself.

Figure 12-2: Block Diagram of a Typical Shared Port Structure



Notes: This block diagram is a general representation of a shared port/peripheral structure for illustration purposes only. The actual structure for any specific port/peripheral combination may be different than what is shown here.

R = Peripheral input buffer types may vary. Refer to the PIC32MX data sheet for peripheral details.

12.3.6.1 Multiplexed Digital Input Peripheral

The following conditions are characteristic of a multiplexed digital input peripheral:

- Peripheral does not control the TRISx register.
  Some peripherals require the pin be configured as an input by setting the corresponding TRISx bit = 1.
- Peripheral input path is independent of I/O input path and uses an input buffer that is dependent on the peripheral.
- PORTx register data input path is not affected and is able to read the pin value.

12.3.6.2 Multiplexing Digital Output Peripheral

The following conditions are characteristic of a multiplexed digital output peripheral:

- Peripheral controls the output data.
  Some peripherals require the pin be configured as an output by setting the corresponding TRISx bit = 0.
- If a peripheral pin has an automatic tri-state feature, e.g., PWM outputs, the peripheral has the ability to tri-state the pin.
- Pin output driver type could be affected by peripheral, e.g., drive strength, slew rate, etc.
- PORTx register output data has no effect.

12.3.6.3 Multiplexing Digital Bidirectional Peripheral

The following conditions are characteristic of a multiplexed digital bidirectional peripheral:

- Peripheral automatically configures the pin as an output, but not as an input.
  Some peripherals require the pin be configured as an input by setting the corresponding TRISx bit = 1.
- Peripherals control output data.
- Pin output driver type could be affected by peripheral (e.g., drive strength, slew rate, etc.).
- PORTx register data input path is not affected and is able to read the pin value.
- PORTx register output data has no effect.

12.3.6.4 Multiplexing Analog Input Peripheral

The following condition is characteristic of a multiplexed analog input peripheral:

All digital port input buffers are disabled and PORTx registers read '0' to prevent crowbar current.

12.3.6.5 Multiplexing Analog Output Peripheral

The following conditions are characteristic of a multiplexed analog output peripheral:

- All digital port input buffers are disabled and PORTx registers read '0' to prevent crowbar current.
- Analog output is driven onto the pin independent of the associated TRISx setting.

Note:	In order to use pins that are multiplexed with the A/D module for digital I/O, the corresponding bits in the AD1PCFG register must be set to '1' – even if the A/D module is turned off.

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

As shown in Figure 12-2, 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 LAT registers:

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

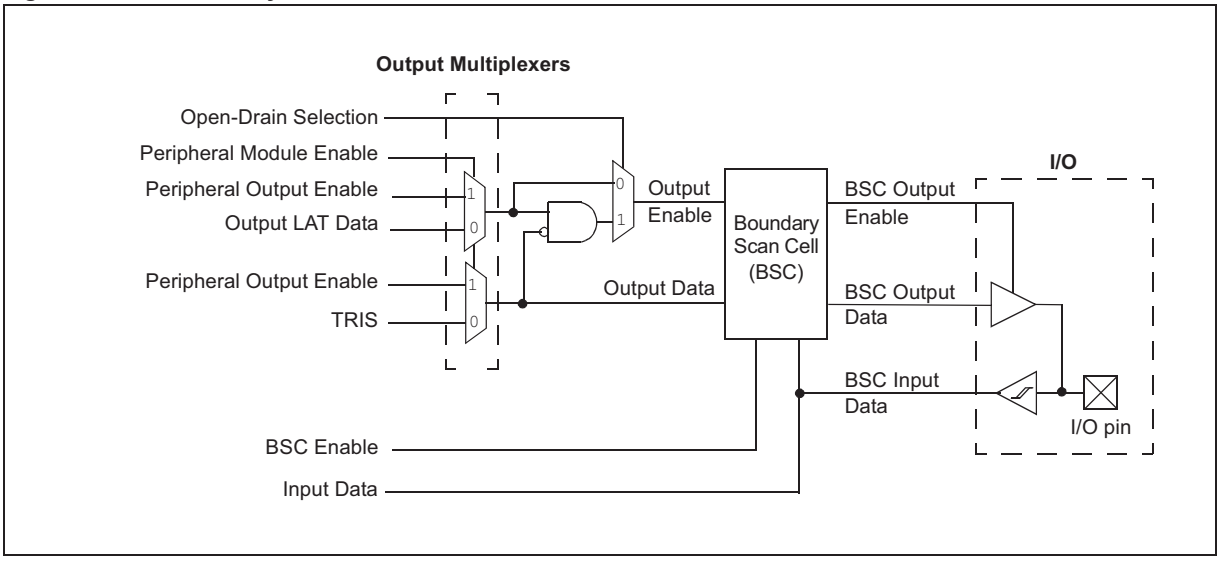
Most serial communication peripherals, when enabled, take full control of the I/O pin so that the input pins associated with the peripheral cannot be affected through the corresponding PORT registers. These peripherals include the following modules:

- SPI
- UART
- I^2CUART

12.3.7 Boundary Scan Cell Connections

The PIC32MX device supports JTAG boundary scan. A Boundary Scan Cell (BSC) is inserted between the internal I/O logic circuit and the I/O pin, as shown in Figure 12-3. Most of the I/O pads have boundary scan cells, however, JTAG pads do not. For normal I/O operation, the BSC is disabled and hence bypassed: The output enable input of the BSC is directly connected to the BSC output enable, and the output data input of the BSC is directly connected to the BSC output data. The pads that do not have BSC are the power supply pads (VDD, VSS and VCAP/VDDCORE) and the JTAG pads (TCK, TDI, TDO and TMS).

Figure 12-3: Boundary Scan Cell Connections



12.3.8 Port Descriptions

Refer to the specific device data sheet for a description of the available I/O ports and peripheral multiplexing details.

12.3.9 Change Notification Pins

The Change Notification (CN) pins provide PIC32MX devices the ability to generate interrupt requests to the processor in response to a change of state on selected input pins (corresponding TRISx bits must = 1). Up to 22 input pins may be selected (enabled) for generating CN interrupts. The total number of available CN inputs is dependent on the selected PIC32MX device. Refer to the specific device data sheet for further details.

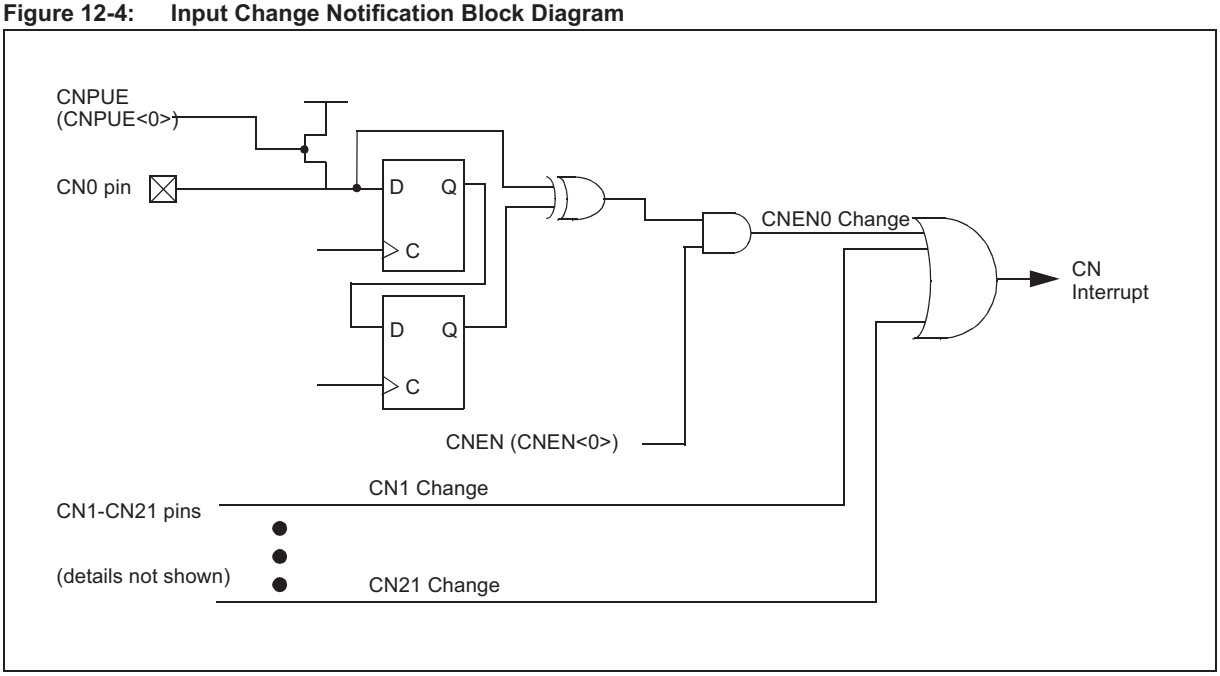
The enabled pin values are compared with the values sampled during the last read operation of the designated PORT register. If the pin value is different from the last value read, a mismatch condition is generated. The mismatch condition can occur on any of the enabled input pins. The mismatches are ORed together to provide a single interrupt-on-change signal. The enabled pins are sampled on every internal system clock cycle, SYSCLK.

Each CN pin has a pull up connected to it. The pull ups 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. The pull ups are enabled separately using the CNPUE register, which contain the control bits for each of the CN pins. Setting any of the CNPUE register bits enables the pull up for the corresponding pins.

Note:	Pull up on CN pins should always be disabled whenever the port pin is configured as a digital output.

Figure 12-4 shows the basic function of the CN hardware.

Figure 12-4: Input Change Notification Block Diagram



12.3.9.1 CN Configuration and Operation

The CN pins are configured as follows:

1. Disable CPU interrupts.
2. Set desired CN I/O pin as input by setting corresponding TRISx register bits = 1.
   Note: If the I/O pin is shared with an analog peripheral, it may be necessary to set the corresponding AD1PCFG register bit = 1 to ensure that the I/O pin is a digital input.
3. Enable CN Module ON (CNCON<15>) = 1.
4. Enable individual CN input pin(s), enable optional pull up(s).
5. Read corresponding PORT registers to clear mismatch condition on CN input pins.
6. Configure the CN interrupt priority, CNIP<2:0>, and subpriority CNIS<1:0>.
7. Clear CN interrupt flag, CNIF = 0.
8. Enable CN interrupt enable, CNIE = 1.
9. Enable CPU interrupts.

When a CN interrupt occurs, the user 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 or during initialization, and used to determine which pin changed.

The CN pins have a minimum input pulse-width specification. Refer to the **"Electrical Characteristics"** section of the data sheet for the specific device to learn more.

12.4 INTERRUPTS

12.4.1 Interrupt Configuration

The CN module has a dedicated interrupt flag bit CNIF and a corresponding interrupt enable/mask bit, CNIE. These bits are used to determine the source of an interrupt and to enable or disable an individual interrupt source.

The CNIE bit is used to define the behavior of the Interrupt Controller when a corresponding CNIF is set. When the CNIE bit is clear, the Interrupt Controller module does not generate a CPU interrupt for the event. If the CNIE bit is set, the Interrupt Controller module will generate an interrupt to the CPU when the corresponding CNIF bit is set (subject to the priority and subpriority as outlined below).

It is the responsibility of the user's software routine that services a particular interrupt to clear the appropriate interrupt flag bit before the service routine is complete.

The priority of the CN module can be set with the CNIP<2:0> bits. This priority defines the priority group to which the interrupt source will be assigned. The priority groups range from a value of 7 (the highest priority), to a value of 0 (which does not generate an interrupt). An interrupt being serviced will be preempted by an interrupt in a higher priority group.

The subpriority bits allow setting the priority of a interrupt source within a priority group. The values of the subpriority CNIS<1:0> range from 3 (the highest priority), to 0 (the lowest priority). An interrupt with the same priority group but having a higher subpriority value will preempt a lower subpriority interrupt that is in progress.

The priority group and subpriority bits allow more than one interrupt source to share the same Priority and subpriority. If simultaneous interrupts occur in this configuration, the natural order of the interrupt sources within a priority/sub-group pair determine the interrupt generated. The natural priority is based on the vector numbers of the interrupt sources. The lower the vector number the higher the natural priority of the interrupt. Any interrupts that were overridden by natural order will then generate their respective interrupts based on priority, subpriority, and natural order after the interrupt flag for the current interrupt is cleared.

After an enabled interrupt is generated, the CPU jumps to the vector assigned to that interrupt. The vector number for the interrupt is the same as the natural order number. The CPU then begins executing code at the vector address. The user's code at this vector address should perform any application specific operations and clear the CNIF interrupt flag, and then exit. Refer to **Section 8. "Interrupts"** for the vector address table details for more information on interrupts.

Table 12-2: Change Notice Interrupt Vector with EBASE = 0x8000:0000

Interrupt	Vector/ Natural Order	IRQ Number	Vector Address IntCtl.VS = 0x01	Vector Address IntCtl.VS = 0x02	Vector Address IntCtl.VS = 0x04	Vector Address IntCtl.VS = 0x08	Vector Address IntCtl.VS = 0x10
CN	23	23	8000 04E0	8000 07C0	8000 0D80	8000 1900	8000 3000

Table 12-3: Example of Priority and Subpriority Assignment

Interrupt	Priority Group	Subpriority	Vector/Natural Order
CN	7	3	23

Example 12-1: Change Notice Configuration Example

```
/*
    The following code example illustrates a Change Notice interrupt configuration for pins
    CN1(PORTC.RC13), CN4(PORTB.RB2) and CN18(PORTF.RF5).
*/

    /* NOTE: disable vector interrupts prior to configuration */

    CNCON = 0x8000;        // Enable Change Notice module
    CNEN =  0x00040012;    // Enable individual CN pins CN1, CN4 and CN18
    CNPUE = 0x00040012;    // Enable weak pull ups for pins CN1, CN4 and CN18

    /* read port(s) to clear mismatch on change notice pins */
    PORTB;
    PORTC;
    PORTF;

    IPC6SET = 0x00140000;  // Set priority level=5
    IPC6SET = 0x00030000;  // Set Subpriority level=3
                           // Could have also done this in single
                           // operation by assigning IPC6SET = 0x00170000

    IFS1CLR  = 0x0001;     // Clear the interrupt flag status bit
    IEC1SET  = 0x0001;     // Enable Change Notice interrupts

    /* re-enable vector interrupts after configuration */
```

Example 12-2: Change Notice ISR Code Example

```
/*
    The following code example demonstrates a simple interrupt service routine for CN
    interrupts. The user's code at this vector can perform any application specific
    operations. The user's code must read the CN corresponding PORT registers to clear the
    mismatch conditions before clearing the CN interrupt status flag. Finally, the CN
    interrupt status flag must be cleared before exiting.
*/
void __ISR(_CHANGE_NOTICE_VECTOR, ipl5 ChangeNoticeHandler(void)
{
    ... perform application specific operations in response to the interrupt

    readB = PORTB          // Read PORTB to clear CN4 mismatch condition
    readC = PORTC          // Read PORTC to clear CN1 mismatch condition
    readF = PORTF          // Read PORTF to clear CN18 mismatch condition
    ...
    IFS1CLR  = 0x0001;     // Be sure to clear the CN interrupt status
                           // flag before exiting the service routine.
{
```

Note: The CN ISR code example shows MPLAB® C32 C compiler specific syntax. Refer to your compiler manual regarding support for ISRs.

12.5 OPERATION IN POWER-SAVING AND DEBUG MODES

Note: In this manual, a distinction is made between a power mode as it is used in a specific module, and a power mode as it is used by the device, e.g., Sleep mode of the Comparator and SLEEP mode of the CPU. To indicate which type of power mode is intended, uppercase and lowercase letters (Sleep, Idle, Debug) signify a module power mode, and all uppercase letters (SLEEP, IDLE, DEBUG) signify a device power mode.

12.5.1 I/O Port Operation in SLEEP Mode

As the device enters SLEEP mode, the system clock is disabled; however, the CN module continues to operate. If one of the enabled CN pins changes state, the Status bit CNIF (IFS1<0>) will be set. If the CNIE bit (IEC1<0>) is set, and its priority is greater than current CPU priority, the device will wake from SLEEP or IDLE mode and execute the CN Interrupt Service Routine.

If the assigned priority level of the CN interrupt is less than or equal to the current CPU priority level, the CPU will not be awakened and the device will enter IDLE mode.

12.5.2 I/O Port Operation in IDLE Mode

As the device enters IDLE mode, the system clock sources remain functional. The SIDL bit (CNCON<13>) selects whether the module will stop or continue functioning on IDLE.

- If SIDL = 1, the module will continue to sample Input CN I/O pins in IDLE mode, however, synchronization is disabled.
- If SIDL = 0, the module will continue to synchronize and sample Input CN I/O pins in IDLE mode.

12.5.3 I/O Port Operation in DEBUG Mode

The FRZ bit (CNCON<14>) determines whether the CN module will run or stop while the CPU is executing DEBUG exception code (i.e., application is halted) in DEBUG mode.

- If FRZ = 0, the module continues to operate even when application is halted in DEBUG mode.
- If FRZ = 1 and application is halted in DEBUG mode, the module will freeze its operations and make no changes to the state of the CN module. The module will resume its operation after CPU resumes execution.

Note: The FRZ bit is readable and writable only when the CPU is executing in Debug Exception mode. In all other modes, the FRZ bit reads as '0'. If FRZ bit is changed during DEBUG mode, the new value does not take effect until the current Debug Exception mode is exited and re-entered. During the Debug Exception mode, the FRZ bit reads the state of the peripheral when entering DEBUG mode.

12.6 EFFECTS OF VARIOUS RESETS

12.6.1 Device Reset

All TRIS, LAT, PORT, ODC, CNEN, CNPUE and CNCON registers are forced to their Reset states upon a device Reset.

12.6.2 Power-On Reset

All TRIS, LAT, PORT, ODC, CNEN, CNPUE and CNCON registers are forced to their Reset states upon a Power-on Reset.

12.6.3 Watchdog Reset

All TRIS, LAT, PORT, ODC, CNEN, CNPUE and CNCON registers are unchanged upon a Watchdog Reset.

12.7 I/O Port Application

Example 12-3: Code Example

```
/*
 The following code example illustrates configuring
 RB0, RB1 as analog (default) inputs, RB2 as a digital
 input, RB3 as digital output and RB4 as digital output
 with open-drain enabled using SET, CLR atomic SFR registers.*/

 AD1PCFGCLR = 0x0003;       // RB0, RB1 = analog pins
 TRISBSET = 0x0003;         // RB0, RB1 = inputs

 AD1PCFGSET = 0x000C;       // RB2, RB3 = digital pins
 TRISBSET =  0x0004;        // RB2 = input
 TRISBCLR =  0x0018;        // RB3, RB4 = outputs

 ODCBSET = 0x0010;          // RB4 open-drain enabled

/*
 The following code example illustrates same configuration
 above using Base SFR registers directly.*/

 AD1PCFG = 0x001C;          // RB0, RB1 = analog pins; RB2, RB3, RB4 = digital pins
 TRISB =  0x0007;           // RB0, RB1, RB2 = inputs; RB3, RB4 = outputs

 ODCB = 0x0010;             // RB4 open-drain enabled
```

12.8 I/O PIN CONTROL

Table provides a summary of I/O pin mode settings.

Table 12-4: I/O Pin Configurations

Required Settings for Digital Pin Control							
Mode or Pin Usage	**Pin Type**	**Buffer Type**	**TRIS Bit**	**ODC Bit**	**CNEN Bit**	**CNPUE Bit(1)**	**AD1PCFG Bit**
Input	IN	ST	1	—	—	—	1
CN	IN	ST	1	—	1	1	1
Output	OUT	CMOS	0	0	—	—	1
Open Drain	OUT	OPEN	0	1	—	—	1

Required Settings for Analog Pin Control							
Mode or Pin Usage	**Pin Type**	**Buffer Type**	**TRIS Bit**	**ODC Bit**	**CNEN Bit**	**CNPUE Bit(1)**	**AD1PCFG Bit**
ANx Input	IN	A	1	—	—	—	0
CV Output	OUT	A	—	—	—	—	0

Required Settings for JTAG Pin Control(2)							
Mode or Pin Usage	**Pin Type**	**Buffer Type**	**TRIS Bit**	**ODC Bit**	**CNEN Bit**	**CNPUE Bit(1)**	**AD1PCFG Bit**
TCK	IN	ST	—	—	—	—	—
TDI	IN	ST	—	—	—	—	—
TMS	IN	ST	—	—	—	—	—
TDO	OUT	CMOS	—	—	—	—	—

Required Settings for ICSP Pin Control(3)							
Mode or Pin Usage	**Pin Type**	**Buffer Type**	**TRIS Bit**	**ODC Bit**	**CNEN Bit**	**CNPUE Bit(1)**	**AD1PCFG Bit**
PGC	IN	ST	—	—	—	—	—
	OUT	CMOS	—	—	—	—	—
PGD	IN	ST	—	—	—	—	—
	OUT	CMOS	—	—	—	—	—

Legend: CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
I = Input
O = Output

Note 1: The CN Enable Pull-up bit is optional.

2: The pin control for the JTAG module is automatically set when JTAG is enabled and the corresponding DEBUGGING mode is selected. No user configuration is required.

3: The pin control for the ICSP™ module is set automatically when entering ICSP mode. No user configuration is required.

12.9 DESIGN TIPS

Question 1: ***How should I configure my unused I/O pins?***

Answer: I/O pins that are not used can be set as outputs (corresponding TRIS bit = 0) and driven low (corresponding LAT bit = 0) in software.

Question 2: ***Is it possible to connect PIC32MX I/O pins to a 5V device?***

Answer: Yes, with limitations. PIC32MX I/O pins are 5V tolerant when configured as an input, which means the pin can tolerate an input up to 5V. When configured as an output, an I/O pin can only drive as high as the voltage supplied to the PIC32MX V_{DD} pin, which is limited to 3.6V. Depending on the 5V device's input pin design, this may not be sufficient to be correctly read as a logic "high" signal. For a detailed discussion on interfacing different logic level families, refer to the "*Microchip 3V Tips 'n Tricks*" (DS41285) guide.

12.10 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 PIC32MX 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: Please visit the Microchip web site (www.microchip.com) for additional application notes and code examples for the PIC32MX family of devices.

12.11 REVISION HISTORY

Revision A (August 2007)

This is the initial released version of this document.

Revision B (October 2007)

Updated document to remove Confidential status.

Revision C (April 2008)

Revised status to Preliminary; Revised U-0 to r-x; Revised Register 12-13; Revised Figure 12-1 and 12-2.

Revision D (May 2008)

Revised Register 12-17, add note to FRZ; Add note to Registers 12-19, 12-30, 12-31; Revised Example 12-1 and 12-2; Change Reserved bits from “Maintain as” to “Write”; Added Note to ON bit (CNCON Register).

NOTES:

