Microchip Technology - DSPIC33CH128MP202T-I/2N Datasheet

Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

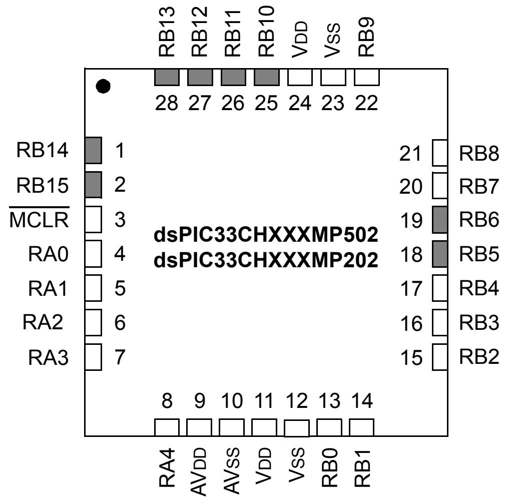
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	dsPIC
Core Size	16-Bit Dual-Core
Speed	180MHz, 200MHz
Connectivity	I²C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, Motor Control PWM, POR, PWM, QEI, WDT
Number of I/O	21
Program Memory Size	152KB (152K x 8)
Program Memory Type	FLASH, PRAM
EEPROM Size	-
RAM Size	20K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 23x12b; D/A 4x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-UQFN Exposed Pad
Supplier Device Package	28-UQFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/dspic33ch128mp202t-i-2n

Pin Diagrams (Continued)

28-Pin UQFN[1,2]



Note 1: **Shaded pins** are up to 5.5 VDC tolerant (refer to Table 3-28 and Table 4-25). For the list of analog ports, refer to Table 3-27 and Table 4-24.

2: The large center pad on the bottom of the package may be left floating or connected to VSS. The four-corner anchor pads are internally connected to the large bottom pad, and therefore, must be connected to the same net as the large center pad.

TABLE 5: 28-PIN UQFN

Pin #	Master Core	Slave Core
1	**RP46**/PWM1H/RB14	**S1RP46**/S1PWM1H/S1RB14
2	**RP47**/PWM1L/RB15	**S1RP47**/S1PWM6H/S1PWM1L/S1RB15
3	$\overline{\text{MCLR}}$	—
4	AN0/CMP1A/RA0	S1RA0
5	AN1/RA1	S1AN15/S1RA1
6	AN2/RA2	S1AN16/S1RA2
7	AN3/IBIAS0/RA3	S1AN0/S1CMP1A/S1PGA1P1/S1RA3
8	AN4/IBIAS1/RA4	$\overline{\text{S1MCLR3}}$/S1AN1/S1CMP2A/S1PGA2P1/S1PGA3P2/S1RA4
9	AVDD	AVDD
10	AVSS	AVSS
11	VDD	VDD
12	VSS	VSS
13	OSCI/CLKI/AN5/**RP32**/RB0	S1AN5/**S1RP32**/S1RB0
14	OSCO/CLKO/AN6/IBIAS2/**RP33**/RB1	S1AN4/**S1RP33**/S1RB1
15	DACOUT/AN7/CMP1D/**RP34**/INT0/RB2	$\overline{\text{S1MCLR2}}$/S1AN3/S1ANC0/S1ANC1/S1CMP1D/S1CMP2D/S1CMP3D/**S1RP34**/S1INT0/S1RB2
16	PGD2/AN8/**RP35**/RB3	S1PGD2/S1AN18/S1CMP3A/S1PGA3P1/**S1RP35**/S1RB3
17	PGC2/**RP36**/RB4	S1PGC2/S1AN9/**S1RP36**/S1PWM5L/S1RB4
18	PGD3/**RP37**/SDA2/RB5	S1PGD3/**S1RP37**/S1RB5
19	PGC3/**RP38**/SCL2/RB6	S1PGC3/**S1RP38**/S1RB6
20	TDO/AN9/**RP39**/RB7	$\overline{\text{S1MCLR1}}$/S1AN6/**S1RP39**/S1PWM5H/S1RB7
21	PGD1/AN10/**RP40**/SCL1/RB8	S1PGD1/S1AN7/**S1RP40**/S1SCL1/S1RB8
22	PGC1/AN11/**RP41**/SDA1/RB9	S1PGC1/**S1RP41**/S1SDA1/S1RB9
23	VSS	VSS
24	VDD	VDD
25	TMS/**RP42**/PWM3H/RB10	**S1RP42**/S1PWM3H/S1RB10
26	TCK/**RP43**/PWM3L/RB11	**S1RP43**/S1PWM8H/S1PWM3L/S1RB11
27	TDI/**RP44**/PWM2H/RB12	**S1RP44**/S1PWM2H/S1RB12
28	**RP45**/PWM2L/RB13	**S1RP45**/S1PWM7H/S1PWM2L/S1RB13

Legend: **RPn** and **S1RPn** represent remappable pins for Peripheral Pin Select functions.

Table of Contents

1.0 DEVICE OVERVIEW

Note 1: This data sheet summarizes the features of the dsPIC33CH128MP508 family of devices. It is not intended to be a comprehensive resource. To complement the information in this data sheet, refer to the related section of the *"dsPIC33/PIC24 Family Reference Manual"*, which is available from the Microchip web site (www.microchip.com).

2: Some registers and associated bits described in this section may not be available on all devices. Refer to **Section 3.2 "Master Memory Organization"** and **Section 4.2 "Slave Memory Organization"** in this data sheet for device-specific register and bit information.

This document contains device-specific information for the dsPIC33CH128MP508 Digital Signal Controller (DSC) and Microcontroller (MCU) devices.

dsPIC33CH128MP508 devices contain extensive Digital Signal Processor (DSP) functionality with a high-performance, 16-bit MCU architecture.

Figure 1-2 shows a general block diagram of the cores and peripheral modules of the Master and Slave. Table 1-1 lists the functions of the various pins shown in the pinout diagrams.

The Master core and Slave core can operate independently, and can be programmed and debugged separately during the application development. Both processor (Master and Slave) subsystems have their own interrupt controllers, clock generators, ICD, port logic, I/O MUXes and PPS. The device is equivalent to having two complete dsPIC® DSCs on a single die.

The Master core will execute the code from Program Flash Memory (PFM) and the Slave core will operate from Program RAM Memory (PRAM).

Once the code development is complete, the Master Flash will be programmed with the Master code, as well as the Slave code. After a Power-on Reset (POR), the Slave code from Master Flash will be loaded to the PRAM (program memory of the Slave) and the Slave can execute the code independently of the Master. The Master and Slave can communicate with each other using the Master Slave Interface (MSI) peripheral, and can exchange data between them.

Figure 1-1 shows the block diagram of the device operation during a POR and the process of transferring the Slave code from the Master to Slave PRAM.

The I/O ports are shared between the Master and Slave. Table 1 shows the number of peripherals and the shared peripherals that the Master and Slave own. There are Configuration bits in the Flash memory that specify the ownership (Master or Slave) of each device pin.

The default (erased) state of the Flash assigns all of the device pins to the Master.

The two cores (Master and Slave) can both be connected to debug tools, which support independent and simultaneous debugging. When the Slave core or Master core is debugged (non-Dual Debug mode), the $\overline{\text{S1MCLRx}}$ is not used. $\overline{\text{MCLR}}$ is used for programming and debugging both the Master core and the Slave core. $\overline{\text{S1MCLRx}}$ is only used when debugging both the cores at the same time.

In normal operation, the "owner" of a device pin is responsible for full control of that pin; this includes both the digital and analog functionality.

The pin owner's GPIO registers control all aspects of the I/O pad, including the ANSELx, CNPUx, CNPDx, ODCx registers and slew rate control.

Note: Both the Master and Slave cores can monitor a pin as an input, regardless of pin ownership. Pin ownership is valid only for the output functionality of the port.

TABLE 3-17: MASTER SFR BLOCK E00h

Register	Address	All Resets	Register	Address	All Resets	Register	Address	All Resets
I/O Ports (Continued)			CNCONB	E2A	0---0-----------	LATD	E5A	xxxxxxxxxxxxxxxx
ANSELA	E00	-----------11111	CNEN0B	E2C	0000000000000000	ODCD	E5C	0000000000000000
TRISA	E02	-----------11111	CNSTATB	E2E	0000000000000000	CNPUD	E5E	0000000000000000
PORTA	E04	-----------xxxxx	CNEN1B	E30	0000000000000000	CNPDD	E60	0000000000000000
LATA	E06	-----------xxxxx	CNFB	E32	0000000000000000	CNCOND	E62	0---0-----------
ODCA	E08	-----------00000	ANSELC	E38	--------1---1111	CNEN0D	E64	0000000000000000
CNPUA	E0A	-----------00000	TRISC	E3A	1111111111111111	CNSTATD	E66	0000000000000000
CNPDA	E0C	-----------00000	PORTC	E3C	xxxxxxxxxxxxxxxx	CNEN1D	E68	0000000000000000
CNCONA	E0E	0---0-----------	LATC	E3E	xxxxxxxxxxxxxxxx	CNFD	E6A	0000000000000000
CNEN0A	E10	-----------00000	ODCC	E40	0000000000000000	TRISE	E72	1111111111111111
CNSTATA	E12	-----------00000	CNPUC	E42	0000000000000000	PORTE	E74	xxxxxxxxxxxxxxxx
CNEN1A	E14	-----------00000	CNPDC	E44	0000000000000000	LATE	E76	xxxxxxxxxxxxxxxx
CNFA	E16	-----------00000	CNCONC	E46	0---0-----------	ODCE	E78	0000000000000000
ANSELB	E1C	------111---1111	CNEN0C	E48	0000000000000000	CNPUE	E7A	0000000000000000
TRISB	E1E	1111111111111111	CNSTATC	E4A	0000000000000000	CNPDE	E7C	0000000000000000
PORTB	E20	xxxxxxxxxxxxxxxx	CNEN1C	E4C	0000000000000000	CNCONE	E7E	0---0-----------
LATB	E22	xxxxxxxxxxxxxxxx	CNFC	E4E	0000000000000000	CNEN0E	E80	0000000000000000
ODCB	E24	0000000000000000	ANSELD	E54	-----1----------	CNSTATE	E82	0000000000000000
CNPUB	E26	0000000000000000	TRISD	E56	1111111111111111	CNEN1E	E84	0000000000000000
CNPDB	E28	0000000000000000	PORTD	E58	xxxxxxxxxxxxxxxx	CNFE	E86	0000000000000000

Legend: x = unknown or indeterminate value; “-” = unimplemented bits. Address and Reset values are in hexadecimal and binary, respectively.

3.2.7 MODULO ADDRESSING

Modulo Addressing mode is a method of providing an automated means to support circular data buffers using hardware. The objective is to remove the need for software to perform data address boundary checks when executing tightly looped code, as is typical in many DSP algorithms.

Modulo Addressing can operate in either Data or Program Space (since the Data Pointer mechanism is essentially the same for both). One circular buffer can be supported in each of the X (which also provides the pointers into Program Space) and Y Data Spaces. Modulo Addressing can operate on any W Register Pointer. However, it is not advisable to use W14 or W15 for Modulo Addressing since these two registers are used as the Stack Frame Pointer and Stack Pointer, respectively.

In general, any particular circular buffer can be configured to operate in only one direction, as there are certain restrictions on the buffer start address (for incrementing buffers) or end address (for decrementing buffers), based upon the direction of the buffer.

The only exception to the usage restrictions is for buffers that have a power-of-two length. As these buffers satisfy the start and end address criteria, they can operate in a Bidirectional mode (that is, address boundary checks are performed on both the lower and upper address boundaries).

3.2.7.1 Start and End Address

The Modulo Addressing scheme requires that a starting and ending address be specified and loaded into the 16-bit Modulo Buffer Address registers: XMODSRT, XMODEND, YMODSRT and YMODEND (see Table 3-4).

> **Note:** Y space Modulo Addressing EA calculations assume word-sized data (LSb of every EA is always clear).

The length of a circular buffer is not directly specified. It is determined by the difference between the corresponding start and end addresses. The maximum possible length of the circular buffer is 32K words (64 Kbytes).

3.2.7.2 W Address Register Selection

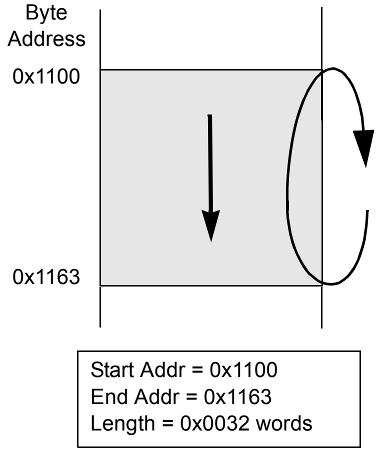
The Modulo and Bit-Reversed Addressing Control register, MODCON<15:0>, contains enable flags, as well as a W register field to specify the W Address registers. The XWM and YWM fields select the registers that operate with Modulo Addressing:

- If XWM = 1111, X RAGU and X WAGU Modulo Addressing is disabled
- If YWM = 1111, Y AGU Modulo Addressing is disabled

The X Address Space Pointer W (XWM) register, to which Modulo Addressing is to be applied, is stored in MODCON<3:0> (see Table 3.2.1). Modulo Addressing is enabled for X Data Space when XWM is set to any value other than '1111' and the XMODEN bit is set (MODCON<15>).

The Y Address Space Pointer W (YWM) register, to which Modulo Addressing is to be applied, is stored in MODCON<7:4>. Modulo Addressing is enabled for Y Data Space when YWM is set to any value other than '1111' and the YMODEN bit (MODCON<14>) is set.

FIGURE 3-10: MODULO ADDRESSING OPERATION EXAMPLE



```
MOV     #0x1100, W0
MOV     W0, XMODSRT    ;set modulo start address
MOV     #0x1163, W0
MOV     W0, MODEND     ;set modulo end address
MOV     #0x8001, W0
MOV     W0, MODCON     ;enable W1, X AGU for modulo

MOV     #0x0000, W0    ;W0 holds buffer fill value

MOV     #0x1110, W1    ;point W1 to buffer

DO      AGAIN, #0x31   ;fill the 50 buffer locations
MOV     W0, [W1++]     ;fill the next location
AGAIN:  INC W0, W0     ;increment the fill value
```

3.2.9 INTERFACING PROGRAM AND DATA MEMORY SPACES

The dsPIC33CH128MP508 family architecture uses a 24-bit wide Program Space (PS) and a 16-bit wide Data Space (DS). The architecture is also a modified Harvard scheme, meaning that data can also be present in the Program Space. To use this data successfully, it must be accessed in a way that preserves the alignment of information in both spaces.

Aside from normal execution, the architecture of the dsPIC33CH128MP508 family devices provides two methods by which Program Space can be accessed during operation:

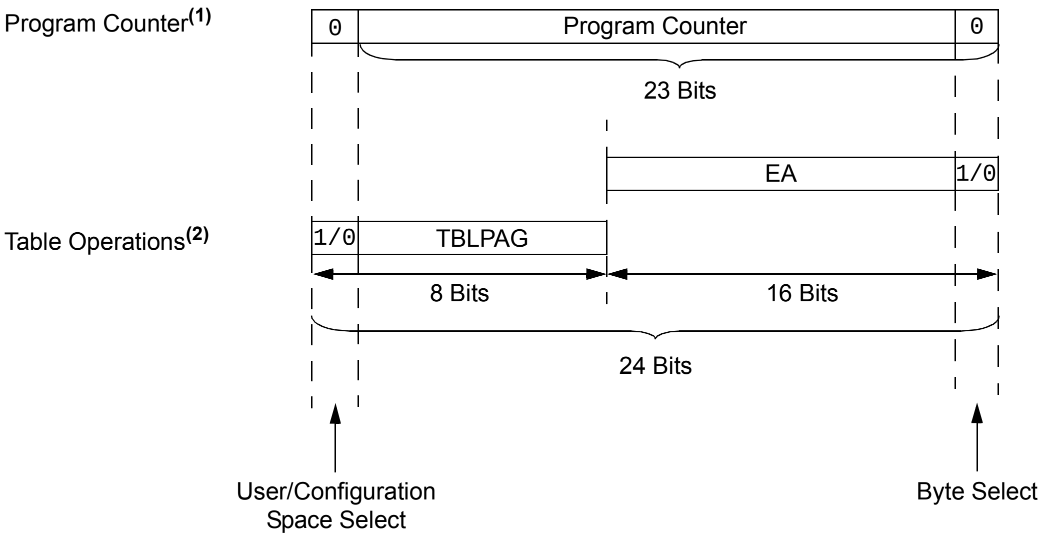
- Using table instructions to access individual bytes or words anywhere in the Program Space
- Remapping a portion of the Program Space into the Data Space (Program Space Visibility)

Table instructions allow an application to read or write to small areas of the program memory. This capability makes the method ideal for accessing data tables that need to be updated periodically. It also allows access to all bytes of the program word. The remapping method allows an application to access a large block of data on a read-only basis, which is ideal for look-ups from a large table of static data. The application can only access the least significant word of the program word.

TABLE 3-22: PROGRAM SPACE ADDRESS CONSTRUCTION

Access Type	Access Space	Program Space Address				
		<23>	**<22:16>**	**<15>**	**<14:1>**	**<0>**
Instruction Access (Code Execution)	User	0	PC<22:1>			0
		0xxx xxxx xxxx xxxx xxxx xxx0				
TBLRD/TBLWT (Byte/Word Read/Write)	User	TBLPAG<7:0>		Data EA<15:0>		
		0xxx xxxx		xxxx xxxx xxxx xxxx		
	Configuration	TBLPAG<7:0>		Data EA<15:0>		
		1xxx xxxx		xxxx xxxx xxxx xxxx		

FIGURE 3-12: DATA ACCESS FROM PROGRAM SPACE ADDRESS GENERATION



Note 1: The Least Significant bit (LSb) of Program Space addresses is always fixed as '0' to maintain word alignment of data in the Program and Data Spaces.

2: Table operations are not required to be word-aligned. Table Read operations are permitted in the configuration memory space.

REGISTER 3-51: RPINR15: PERIPHERAL PIN SELECT INPUT REGISTER 15

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
QEIHOM1R7	QEIHOM1R6	QEIHOM1R5	QEIHOM1R4	QEIHOM1R3	QEIHOM1R2	QEIHOM1R1	QEIHOM1R0
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
QEINDX1R7	QEINDX1R6	QEINDX1R5	QEINDX1R4	QEINDX1R3	QEINDX1R2	QEINDX1R1	QEINDX1R0
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 **QEIHOM1R<7:0>:** Assign QEI Home 1 Input (QEIHOM1) to the Corresponding RPn Pin bits
See Table 3-30.

bit 7-0 **QEINDX1R<7:0>:** Assign QEI Index 1 Input (QEINDX1) to the Corresponding RPn Pin bits
See Table 3-30.

REGISTER 3-52: RPINR18: PERIPHERAL PIN SELECT INPUT REGISTER 18

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
U1DSRR7	U1DSRR6	U1DSRR5	U1DSRR4	U1DSRR3	U1DSRR2	U1DSRR1	U1DSRR0
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
U1RXR7	U1RXR6	U1RXR5	U1RXR4	U1RXR3	U1RXR2	U1RXR1	U1RXR0
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 **U1DSRR<7:0>:** Assign UART1 Data-Set-Ready ($\overline{\text{U1DSR}}$) to the Corresponding RPn Pin bits
See Table 3-30.

bit 7-0 **U1RXR<7:0>:** Assign UART1 Receive (U1RX) to the Corresponding RPn Pin bits
See Table 3-30.

REGISTER 3-90: RPOR22: PERIPHERAL PIN SELECT OUTPUT REGISTER 22

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	RP181R5[1]	RP181R4[1]	RP181R3[1]	RP181R2[1]	RP181R1[1]	RP181R0[1]
bit 15							bit 8

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	RP180R5[1]	RP180R4[1]	RP180R3[1]	RP180R2[1]	RP180R1[1]	RP180R0[1]
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15-14 **Unimplemented:** Read as ‘0’

bit 13-8 **RP181R<5:0>:** Peripheral Output Function is Assigned to RP181 Output Pin bits (see Table 3-33 for peripheral function numbers)

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **RP180R<5:0>:** Peripheral Output Function is Assigned to RP180 Output Pin bits (see Table 3-33 for peripheral function numbers)

Note 1: These are virtual output ports.

4.2.5.3 Move and Accumulator Instructions

Move instructions, and the DSP accumulator class of instructions, provide a greater degree of addressing flexibility than other instructions. In addition to the addressing modes supported by most MCU instructions, move and accumulator instructions also support Register Indirect with Register Offset Addressing mode, also referred to as Register Indexed mode.

Note: For the MOV instructions, the addressing mode specified in the instruction can differ for the source and destination EA. However, the 4-bit Wb (Register Offset) field is shared by both source and destination (but typically only used by one).

In summary, the following addressing modes are supported by move and accumulator instructions:

- Register Direct
- Register Indirect
- Register Indirect Post-Modified
- Register Indirect Pre-Modified
- Register Indirect with Register Offset (Indexed)
- Register Indirect with Literal Offset
- 8-Bit Literal
- 16-Bit Literal

Note: Not all instructions support all the addressing modes given above. Individual instructions may support different subsets of these addressing modes.

4.2.5.4 MAC Instructions

The dual source operand DSP instructions (CLR, ED, EDAC, MAC, MPY, MPY.N, MOVSAC and MSC), also referred to as MAC instructions, use a simplified set of addressing modes to allow the user application to effectively manipulate the Data Pointers through register indirect tables.

The two-source operand prefetch registers must be members of the set {W8, W9, W10, W11}. For data reads, W8 and W9 are always directed to the X RAGU, and W10 and W11 are always directed to the Y AGU. The Effective Addresses generated (before and after modification) must therefore, be valid addresses within X Data Space for W8 and W9, and Y Data Space for W10 and W11.

Note: Register Indirect with Register Offset Addressing mode is available only for W9 (in X space) and W11 (in Y space).

In summary, the following addressing modes are supported by the MAC class of instructions:

- Register Indirect
- Register Indirect Post-Modified by 2
- Register Indirect Post-Modified by 4
- Register Indirect Post-Modified by 6
- Register Indirect with Register Offset (Indexed)

4.2.5.5 Other Instructions

Besides the addressing modes outlined previously, some instructions use literal constants of various sizes. For example, BRA (branch) instructions use 16-bit signed literals to specify the branch destination directly, whereas the DISI instruction uses a 14-bit unsigned literal field. In some instructions, such as ULNK, the source of an operand or result is implied by the opcode itself. Certain operations, such as a NOP, do not have any operands.

4.3.7 SLAVE PROGRAM MEMORY CONTROL/STATUS REGISTERS

REGISTER 4-4: NVMCON: PROGRAM MEMORY SLAVE CONTROL REGISTER

R/SO-0[1]	R/W-0[1]	R/W-0[1]	R/W-0	R/C-0	R/C-0	R/W-0	R/C-0
WR	WREN	WRERR	NVMSIDL[2]	SFTSWP	P2ACTIV	RPDF	URERR
bit 15							bit 8

U-0	U-0	U-0	U-0	R/W-0[1]	R/W-0[1]	R/W-0[1]	R/W-0[1]
—	—	—	—	NVMOP3[3,4]	NVMOP2[3,4]	NVMOP1[3,4]	NVMOP0[3,4]
bit 7							bit 0

Legend:	C = Clearable bit	SO = Settable Only bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15 **WR:** Write Control bit[1]

1 = Initiates a PRAM memory program or erase operation; the operation is self-timed and the bit is cleared by hardware once the operation is complete
0 = Program or erase operation is complete and inactive

bit 14 **WREN:** Write Enable bit[1]

1 = Enables program/erase operations
0 = Inhibits program/erase operations

bit 13 **WRERR:** Write Sequence Error Flag bit[1]

1 = An improper program or erase sequence attempt, or termination has occurred (bit is set automatically on any set attempt of the WR bit)
0 = The program or erase operation completed normally

bit 12 **NVMSIDL:** PRAM Stop in Idle Control bit[2]

1 = PRAM voltage regulator goes into Standby mode during Idle mode
0 = PRAM voltage regulator is active during Idle mode

bit 11 **SFTSWP:** Soft Swap Status bit

1 = Panels have been successfully swapped using the BOOTSWP instruction
0 = Awaiting for panels to be successfully swapped using the BOOTSWP instruction

bit 10 **P2ACTIV:** Dual Boot Active Region Status bit

1 = Panel 2 PRAM is mapped into the active region
0 = Panel 1 PRAM is mapped into the active region

bit 9 **RPDF:** Row Programming Data Format bit

1 = Row data to be stored in PRAM is in compressed format
0 = Row data to be stored in PRAM is in uncompressed format

bit 8 **URERR:** Row Programming Data Underrun Error bit

1 = Indicates row programming operation has been terminated
0 = No data underrun error is detected

bit 7-4 **Unimplemented:** Read as ‘0’

Note 1: These bits can only be reset on a POR.

2: If this bit is set, there will be minimal power savings (I_{IDLE}) and upon exiting Idle mode, there is a delay (T_{VREG}) before PRAM memory becomes operational.

3: All other combinations of NVMOP<3:0> are unimplemented.

4: Execution of the PWRSAV instruction is ignored while any of the NVM operations are in progress.

5: Two adjacent words on a 4-word boundary are programmed during execution of this operation.

TABLE 4-29: SLAVE PPS INPUT CONTROL REGISTERS

Register	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RPCONL	—	—	—	—	IOLOCK	—	—	—	—	—	—	—	—	—	—	—
RPINR0	INT1R7	INT1R6	INT1R5	INT1R4	INT1R3	INT1R2	INT1R1	INT1R0	—	—	—	—	—	—	—	—
RPINR1	INT3R7	INT3R6	INT3R5	INT3R4	INT3R3	INT3R2	INT3R1	INT3R0	INT2R7	INT2R6	INT2R5	INT2R4	INT2R3	INT2R2	INT2R1	INT2R0
RPINR2	T1CKR7	T1CKR6	T1CKR5	T1CKR4	T1CKR3	T1CKR2	T1CKR1	T1CKR0	—	—	—	—	—	—	—	—
RPINR3	ICM1R7	ICM1R6	ICM1R5	ICM1R4	ICM1R3	ICM1R2	ICM1R1	ICM1R0	TCKI1R7	TCKI1R6	TCKI1R5	TCKI1R4	TCKI1R3	TCKI1R2	TCKI1R1	TCKI1R0
RPINR4	ICM2R7	ICM2R6	ICM2R5	ICM2R4	ICM2R3	ICM2R2	ICM2R1	ICM2R0	TCKI2R7	TCKI2R6	TCKI2R5	TCKI2R4	TCKI2R3	TCKI2R2	TCKI2R1	TCKI2R0
RPINR5	ICM3R7	ICM3R6	ICM3R5	ICM3R4	ICM3R3	ICM3R2	ICM3R1	ICM3R0	TCKI3R7	TCKI3R6	TCKI3R5	TCKI3R4	TCKI3R3	TCKI3R2	TCKI3R1	TCKI3R0
RPINR6	ICM4R7	ICM4R6	ICM4R5	ICM4R4	ICM4R3	ICM4R2	ICM4R1	ICM4R0	TCKI4R7	TCKI4R6	TCKI4R5	TCKI4R4	TCKI4R3	TCKI4R2	TCKI4R1	TCKI4R0
RPINR11	OCFBR7	OCFBR6	OCFBR5	OCFBR4	OCFBR3	OCFBR2	OCFBR1	OCFBR0	OCFAR7	OCFAR6	OCFAR5	OCFAR4	OCFAR3	OCFAR2	OCFAR1	OCFAR0
RPINR12	PCI9R7	PCI9R6	PCI9R5	PCI9R4	PCI9R3	PCI9R2	PCI9R1	PCI9R0	PCI8R7	PCI8R6	PCI8R5	PCI8R4	PCI8R3	PCI8R2	PCI8R1	PCI8R0
RPINR13	PCI11R7	PCI11R6	PCI11R5	PCI11R4	PCI11R3	PCI11R2	PCI11R1	PCI11R0	PCI10R7	PCI10R6	PCI10R5	PCI10R4	PCI10R3	PCI10R2	PCI10R1	PCI10R0
RPINR14	QEIB1R7	QEIB1R6	QEIB1R5	QEIB1R4	QEIB1R3	QEIB1R2	QEIB1R1	QEIB1R0	QEIA1R7	QEIA1R6	QEIA1R5	QEIA1R4	QEIA1R3	QEIA1R2	QEIA1R1	QEIA1R0
RPINR15	QEIHOM1R7	QEIHOM1R6	QEIHOM1R5	QEIHOM1R4	QEIHOM1R3	QEIHOM1R2	QEIHOM1R1	QEIHOM1R0	QEINDX1R7	QEINDX1R6	QEINDX1R5	QEINDX1R4	QEINDX1R3	QEINDX1R2	QEINDX1R1	QEINDX1R0
RPINR18	U1DSRR7	U1DSRR6	U1DSRR5	U1DSRR4	U1DSRR3	U1DSRR2	U1DSRR1	U1DSRR0	U1RXR7	U1RXR6	U1RXR5	U1RXR4	U1RXR3	U1RXR2	U1RXR1	U1RXR0
RPINR20	SCK1R7	SCK1R6	SCK1R5	SCK1R4	SCK1R3	SCK1R2	SCK1R1	SCK1R0	SDI1R7	SDI1R6	SDI1R5	SDI1R4	SDI1R3	SDI1R2	SDI1R1	SDI1R0
RPINR21	REFOIR7	REFOIR6	REFOIR5	REFOIR4	REFOIR3	REFOIR2	REFOIR1	REFOIR0	SS1R7	SS1R6	SS1R5	SS1R4	SS1R3	SS1R2	SS1R1	SS1R0
RPINR23	U1CTSR7	U1CTSR6	U1CTSR5	U1CTSR4	U1CTSR3	U1CTSR2	U1CTSR1	U1CTSR0	—	—	—	—	—	—	—	—
RPINR37	PCI17R7	PCI17R6	PCI17R5	PCI17R4	PCI17R3	PCI17R2	PCI17R1	PCI17R0	—	—	—	—	—	—	—	—
RPINR38	—	—	—	—	—	—	—	—	PCI18R7	PCI18R6	PCI18R5	PCI18R4	PCI18R3	PCI18R2	PCI18R1	PCI18R0
RPINR42	PCI13R7	PCI13R6	PCI13R5	PCI13R4	PCI13R3	PCI13R2	PCI13R1	PCI13R0	PCI12R7	PCI12R6	PWM12R5	PWM12R4	PWM12R3	PWM12R2	PWM12R1	PWM12R0
RPINR43	PCI15R7	PCI15R6	PCI15R5	PCI15R4	PCI15R3	PCI15R2	PCI15R1	PCI15R0	PCI14R7	PCI14R6	PCI14R5	PCI14R4	PCI14R3	PCI14R2	PCI14R1	PCI14R0
RPINR44	—	—	—	—	—	—	—	—	PCI16R7	PCI16R6	PCI16R5	PCI16R4	PCI16R3	PCI16R2	PCI16R1	PCI16R0
RPINR45	CLCINAR7	CLCINAR6	CLCINAR5	CLCINAR4	CLCINAR3	CLCINAR2	CLCINAR1	CLCINAR0	—	—	—	—	—	—	—	—
RPINR46	CLCINCR7	CLCINCR6	CLCINCR5	CLCINCR4	CLCINCR3	CLCINCR2	CLCINCR1	CLCINCR0	CLCINBR7	CLCINBR6	CLCINBR5	CLCINBR4	CLCINBR3	CLCINBR2	CLCINBR1	CLCINBR0
RPINR47	ADCTRGR7	ADCTRGR6	ADCTRGR5	ADCTRGR4	ADCTRGR3	ADCTRGR2	ADCTRGR1	ADCTRGR0	CLCINDR7	CLCINDR6	CLCINDR5	CLCINDR4	CLCINDR3	CLCINDR2	CLCINDR1	CLCINDR0

6.3 Slave Oscillator Configuration Registers

Table 6-2 lists the configuration settings that select the device's Slave core oscillator source and operating mode at a POR.

TABLE 6-2: CONFIGURATION BIT VALUES FOR CLOCK SELECTION FOR THE SLAVE

Oscillator Source	Oscillator Mode	S1FNOSC<2:0> Value	POSCMD<1:0> Value[(3)]	Notes
S0	Fast RC Oscillator (FRC)	000	xx	1
S1	Fast RC Oscillator with PLL (FRCPLL)	001	xx	1
S2	Primary Oscillator (EC)	010	00	1
S2	Primary Oscillator (XT)	010	01	
S2	Primary Oscillator (HS)	010	10	
S3	Primary Oscillator with PLL (ECPLL)	011	00	1
S3	Primary Oscillator with PLL (XTPLL)	011	01	
S3	Primary Oscillator with PLL (HSPLL)	011	10	
S4	Reserved	100	xx	1
S5	Low-Power RC Oscillator (LPRC)	101	xx	1
S6	Backup FRC (BFRC)	110	xx	1
S7	Fast RC Oscillator with ÷ N Divider (FRCDIVN)	111	xx	1, 2

Note 1: The OSCO pin function is determined by the S1OSCIOFNC Configuration bit. If both the Master core OSCIOFNC and Slave core S1OSCIOFNC bits are set, the Master core OSCIOFNC bit has priority.

2: This is the default oscillator mode for an unprogrammed (erased) device.

3: The POSCMD<1:0> bits are only available in the Master Oscillator Configuration register, FOSC. This setting configures the Primary Oscillator for use by either core.

REGISTER 6-15: PLLDIV: PLL OUTPUT DIVIDER REGISTER (SLAVE)

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	VCODIV<1:0>	
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-1
—	POST1DIV<2:0>(1,2)			—	POST2DIV<2:0>(1,2)		
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 15-10 **Unimplemented:** Read as ‘0’

bit 9-8 **VCODIV<1:0>:** PLL VCO Output Divider Select bits

11 = FVCO
10 = FVCO/2
01 = FVCO/3
00 = FVCO/4

bit 7 **Unimplemented:** Read as ‘0’

bit 6-4 **POST1DIV<2:0>:** PLL Output Divider #1 Ratio bits(1,2)

POST1DIV<2:0> can have a valid value, from 1 to 7 (POST1DIVx value should be greater than or equal to the POST2DIVx value). The POST1DIVx divider is designed to operate at higher clock rates than the POST2DIVx divider.

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **POST2DIV<2:0>:** PLL Output Divider #2 Ratio bits(1,2)

POST2DIV<2:0> can have a valid value, from 1 to 7 (POST2DIVx value should be less than or equal to the POST1DIVx value). The POST1DIVx divider is designed to operate at higher clock rates than the POST2DIVx divider.

Note 1: The POST1DIVx and POST2DIVx divider values must not be changed while the PLL is operating.

2: The default values for POST1DIVx and POST2DIVx are 4 and 1, respectively, yielding a 150 MHz system source clock.

7.2.1 SLEEP MODE

The following occurs in Sleep mode:

- The system clock source is shut down. If an on-chip oscillator is used, it is turned off.
- The device current consumption is reduced to a minimum, provided that no I/O pin is sourcing current.
- The Fail-Safe Clock Monitor does not operate, since the system clock source is disabled.
- The LPRC clock continues to run in Sleep mode if the WDT is enabled.
- The WDT, if enabled, is automatically cleared prior to entering Sleep mode.
- Some device features or peripherals can continue to operate. This includes items such as the Input Change Notification on the I/O ports or peripherals that use an External Clock input.
- Any peripheral that requires the system clock source for its operation is disabled.

The device wakes up from Sleep mode on any of the these events:

- Any interrupt source that is individually enabled
- Any form of device Reset
- A WDT time-out

On wake-up from Sleep mode, the processor restarts with the same clock source that was active when Sleep mode was entered.

For optimal power savings, the internal regulator and the Flash regulator can be configured to go into standby when Sleep mode is entered by clearing the VREGS (RCON<8>) bit.

7.2.2 IDLE MODE

The following occurs in Idle mode:

- The CPU stops executing instructions.
- The WDT is automatically cleared.
- The system clock source remains active. By default, all peripheral modules continue to operate normally from the system clock source, but can also be selectively disabled (see **Section 7.4 "Peripheral Module Disable"**).
- If the WDT or FSCM is enabled, the LPRC also remains active.

The device wakes from Idle mode on any of these events:

- Any interrupt that is individually enabled
- Any device Reset
- A WDT time-out

On wake-up from Idle mode, the clock is reapplied to the CPU and instruction execution will begin (2-4 clock cycles later), starting with the instruction following the `PWRSAV` instruction or the first instruction in the ISR.

All peripherals also have the option to discontinue operation when Idle mode is entered to allow for increased power savings. This option is selectable in the control register of each peripheral; for example, the SIDL bit in the Timer1 Control register (T1CON<13>).

7.2.3 INTERRUPTS COINCIDENT WITH POWER SAVE INSTRUCTIONS

Any interrupt that coincides with the execution of a `PWRSAV` instruction is held off until entry into Sleep or Idle mode has completed. The device then wakes up from Sleep or Idle mode.

REGISTER 9-10: PWMEVTy: PWM EVENT OUTPUT CONTROL REGISTER y[5]

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
EVTyOEN	EVTyPOL	EVTySTRD	EVTySYNC	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
EVTySEL3	EVTySEL2	EVTySEL1	EVTySEL0	—	EVTyPGS2[2]	EVTyPGS1[2]	EVTyPGS0[2]
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 **EVTyOEN:** PWM Event Output Enable bit
1 = Event output signal is output on PWMEVTy pin
0 = Event output signal is internal only

bit 14 **EVTyPOL:** PWM Event Output Polarity bit
1 = Event output signal is active-low
0 = Event output signal is active-high

bit 13 **EVTySTRD:** PWM Event Output Stretch Disable bit
1 = Event output signal pulse width is not stretched
0 = Event output signal is stretched to 8 PWM clock cycles minimum[1]

bit 12 **EVTySYNC:** PWM Event Output Sync bit
1 = Event output signal is synchronized to the system clock
0 = Event output is not synchronized to the system clock
Event output signal pulse will be two system clocks when this bit is set and EVTySTRD = 1.

bit 11-8 **Unimplemented:** Read as ‘0’

bit 7-4 **EVTySEL<3:0>:** PWM Event Selection bits
1111 = High-resolution error event signal
1110-1010 = Reserved
1001 = ADC Trigger 2 signal
1000 = ADC Trigger 1 signal
0111 = STEER signal (available in Push-Pull Output modes only)[4]
0110 = CAHALF signal (available in Center-Aligned modes only)[4]
0101 = PCI Fault active output signal
0100 = PCI current-limit active output signal
0011 = PCI feed-forward active output signal
0010 = PCI Sync active output signal
0001 = PWM Generator output signal[3]
0000 = Source is selected by the PGTRGSEL<2:0> bits

bit 3 **Unimplemented:** Read as ‘0’

Note 1: The event signal is stretched using the peripheral clock because different PGs may be operating from different clock sources. The leading edge of the event pulse is produced in the clock domain of the PWM Generator. The trailing edge of the stretched event pulse is produced in the peripheral clock domain.
2: No event will be produced if the selected PWM Generator is not present.
3: This is the PWM Generator output signal prior to output mode logic and any output override logic.
4: This signal should be the PGx_clk domain signal prior to any synchronization into the system clock domain.
5: ‘y’ denotes a common instance (A-F).

REGISTER 9-17: PGxyPCIL: PWM GENERATOR xy PCI REGISTER LOW (x = PWM GENERATOR #; y = F, CL, FF OR S)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
TSYNCDIS	TERM2	TERM1	TERM0	AQPS	AQSS2	AQSS1	AQSS0
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
SWTERM	PSYNC	PPS	PSS4	PSS3	PSS2	PSS1	PSS0
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 **TSYNCDIS:** Termination Synchronization Disable bit

1 = Termination of latched PCI occurs immediately
0 = Termination of latched PCI occurs at PWM EOC

bit 14-12 **TERM<2:0>:** Termination Event Selection bits

111 = Selects PCI Source #9
110 = Selects PCI Source #8
101 = Selects PCI Source #1 (PWM Generator output selected by the PWMPCI<2:0> bits)
100 = PGxTRIGC trigger event
011 = PGxTRIGB trigger event
010 = PGxTRIGA trigger event
001 = Auto-Terminate: Terminate when PCI source transitions from active to inactive
000 = Manual Terminate: Terminate on a write of ‘1’ to the SWTERM bit location

bit 11 **AQPS:** Acceptance Qualifier Polarity Select bit

1 = Inverted
0 = Not inverted

bit 10-8 **AQSS<2:0>:** Acceptance Qualifier Source Selection bits

111 = SWPCI control bit only (qualifier forced to ‘0’)
110 = Selects PCI Source #9
101 = Selects PCI Source #8
100 = Selects PCI Source #1 (PWM Generator output selected by the PWMPCI<2:0> bits)
011 = PWM Generator is triggered
010 = LEB is active
001 = Duty cycle is active (base PWM Generator signal)
000 = No acceptance qualifier is used (qualifier forced to ‘1’)

bit 7 **SWTERM:** PCI Software Termination bit

A write of ‘1’ to this location will produce a termination event. This bit location always reads as ‘0’.

bit 6 **PSYNC:** PCI Synchronization Control bit

1 = PCI source is synchronized to PWM EOC
0 = PCI source is not synchronized to PWM EOC

bit 5 **PPS:** PCI Polarity Select bit

1 = Inverted
0 = Not inverted

10.4 Input Capture Mode

Input Capture mode is used to capture a timer value from an independent timer base, upon an event, on an input pin or other internal trigger source. The input capture features are useful in applications requiring frequency (time period) and pulse measurement. Figure 10-6 depicts a simplified block diagram of Input Capture mode.

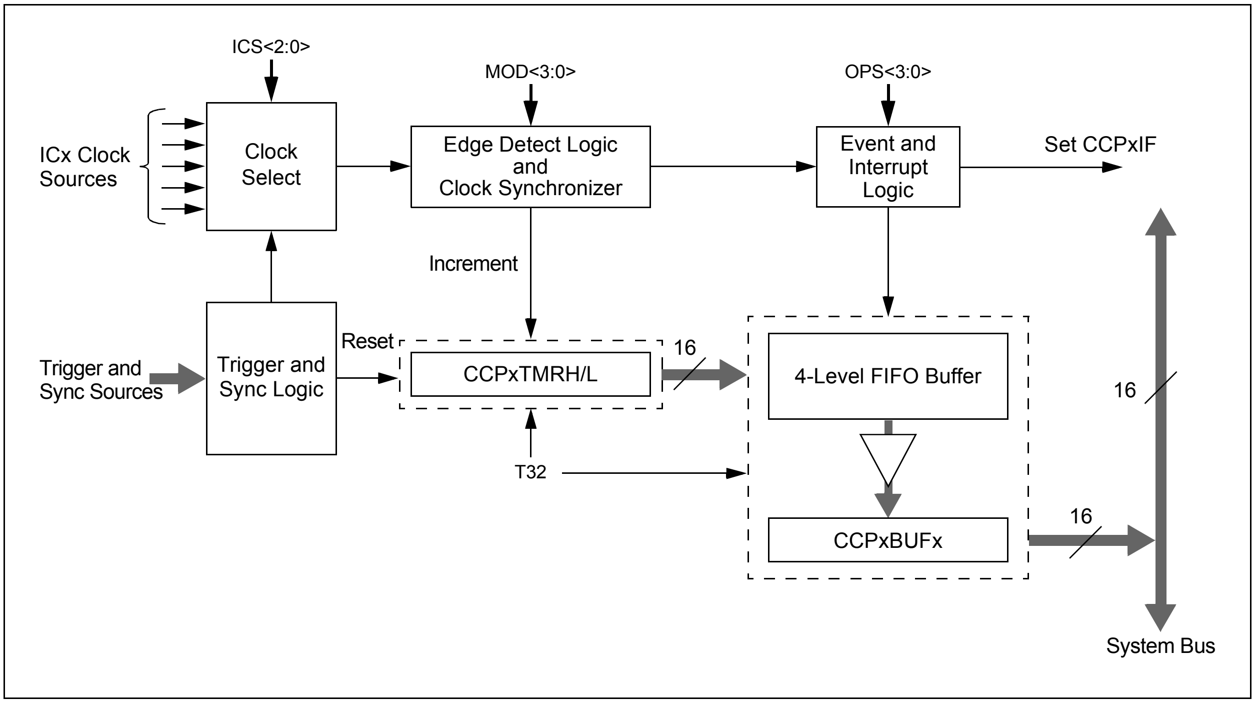
Input Capture mode uses a dedicated 16/32-bit, synchronous, up counting timer for the capture function. The timer value is written to the FIFO when a capture event occurs. The internal value may be read (with a synchronization delay) using the CCPxTMRH/L register.

To use Input Capture mode, the CCSEL bit (CCPxCON1L<4>) must be set. The T32 and the MOD<3:0> bits are used to select the proper Capture mode, as shown in Table 10-4.

TABLE 10-4: INPUT CAPTURE x MODES

MOD<3:0> (CCPxCON1L<3:0>)	T32 (CCPxCON1L<5>)	Operating Mode
0000	0	Edge Detect (16-bit capture)
0000	1	Edge Detect (32-bit capture)
0001	0	Every Rising (16-bit capture)
0001	1	Every Rising (32-bit capture)
0010	0	Every Falling (16-bit capture)
0010	1	Every Falling (32-bit capture)
0011	0	Every Rising/Falling (16-bit capture)
0011	1	Every Rising/Falling (32-bit capture)
0100	0	Every 4th Rising (16-bit capture)
0100	1	Every 4th Rising (32-bit capture)
0101	0	Every 16th Rising (16-bit capture)
0101	1	Every 16th Rising (32-bit capture)

FIGURE 10-6: INPUT CAPTURE x BLOCK DIAGRAM



REGISTER 14-4: SPIxSTATL: SPIx STATUS REGISTER LOW

U-0	U-0	U-0	HS/R/C-0	HSC/R-0	U-0	U-0	HSC/R-0
—	—	—	FRMERR	SPIBUSY	—	—	SPITUR(1)
bit 15							bit 8

HSC/R-0	HS/R/C-0	HSC/R-1	U-0	HSC/R-1	U-0	HSC/R-0	HSC/R-0
SRMT	SPIROV	SPIRBE	—	SPITBE	—	SPITBF	SPIRBF
bit 7							bit 0

Legend:	C = Clearable bit	U = Unimplemented, read as ‘0’	
R = Readable bit	W = Writable bit	HSC = Hardware Settable/Clearable bit	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	HS = Hardware Settable bit

bit 15-13 **Unimplemented:** Read as ‘0’

bit 12 **FRMERR**: SPIx Frame Error Status bit
1 = Frame error is detected
0 = No frame error is detected

bit 11 **SPIBUSY:** SPIx Activity Status bit
1 = Module is currently busy with some transactions
0 = No ongoing transactions (at time of read)

bit 10-9 **Unimplemented:** Read as ‘0’

bit 8 **SPITUR:** SPIx Transmit Underrun Status bit(1)
1 = Transmit buffer has encountered a Transmit Underrun condition
0 = Transmit buffer does not have a Transmit Underrun condition

bit 7 **SRMT:** Shift Register Empty Status bit
1 = No current or pending transactions (i.e., neither SPIxTXB or SPIxTXSR contains data to transmit)
0 = Current or pending transactions

bit 6 **SPIROV:** SPIx Receive Overflow Status bit
1 = A new byte/half-word/word has been completely received when the SPIxRXB was full
0 = No overflow

bit 5 **SPIRBE:** SPIx RX Buffer Empty Status bit
1 = RX buffer is empty
0 = RX buffer is not empty

Standard Buffer Mode:
Automatically set in hardware when SPIxBUF is read from, reading SPIxRXB. Automatically cleared in hardware when SPIx transfers data from SPIxRXSR to SPIxRXB.

Enhanced Buffer Mode:
Indicates RXELM<5:0> = 000000.

bit 4 **Unimplemented:** Read as ‘0’

Note 1: SPITUR is cleared when SPIEN = 0. When IGNTUR = 1, SPITUR provides dynamic status of the Transmit Underrun condition, but does not stop RX/TX operation and does not need to be cleared by software.

21.2 Device Calibration and Identification

The PGAx and current source modules on the dsPIC33CH128MP508 family devices require Calibration Data registers to improve performance of the module over a wide operating range. These Calibration registers are read-only and are stored in configuration memory space. Prior to enabling the module, the calibration data must be read (TBLPAG and Table Read instruction) and loaded into their respective SFR registers. The device calibration addresses are shown in Table 21-4.

The dsPIC33CH128MP508 devices have two Identification registers, near the end of configuration memory space, that store the Device ID (DEVID) and Device Revision (DEVREV). These registers are used to determine the mask, variant and manufacturing information about the device. These registers are read-only and are shown in Register 21-32 and Register 21-33.

TABLE 21-4: DEVICE CALIBRATION ADDRESSES[1]

Calibration Name	Address	Bits 23-16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PGA1CAL	0xF8001C	—	—	—	—	—	—	—	—	—	PGA1 Calibration Data							
PGA2CAL	0xF8001E	—	—	—	—	—	—	—	—	—	PGA2 Calibration Data							
PGA3CAL	0xF80020	—	—	—	—	—	—	—	—	—	PGA3 Calibration Data							
ISRCCAL	0xF80012	—	—	—	—	—	—	—	—	—	—	—	Current Source Calibration Data					

Note 1: The calibration data must be copied into its respective registers prior to enabling the module.

TABLE 24-6: DC CHARACTERISTICS: OPERATING CURRENT (I_{DD}) (MASTER SLEEP/SLAVE RUN)

DC CHARACTERISTICS	Master (Sleep) + Slave (Run)		Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ T_A ≤ +85°C for Industrial -40°C ≤ T_A ≤ +125°C for Extended			
Parameter No.	**Typ.**	**Max.**	**Units**	**Conditions**		
Operating Current (I_{DD})[1]						
DC20a	7.2	9.0	mA	-40°C	3.3V	10 MIPS (N = 1, N2 = 5, N3 = 2, M = 50, F_{VCO} = 400 MHz, F_{PLLO} = 40 MHz)
	7.3	12.6	mA	+25°C		
	7.6	18.9	mA	+85°C		
	11.6	25.6	mA	+125°C		
DC21a	9.0	10.9	mA	-40°C	3.3V	20 MIPS (N = 1, N2 = 5, N3 = 1, M = 50, F_{VCO} = 400 MHz, F_{PLLO} = 80 MHz)
	9.2	14.6	mA	+25°C		
	9.4	20.8	mA	+85°C		
	13.4	27.5	mA	+125°C		
DC22a	13.1	15.2	mA	-40°C	3.3V	40 MIPS (N = 1, N2 = 3, N3 = 1, M = 60, F_{VCO} = 480 MHz, F_{PLLO} = 160 MHz)
	13.2	19.0	mA	+25°C		
	13.4	25.1	mA	+85°C		
	17.3	31.5	mA	+125°C		
DC23a	18.6	21.2	mA	-40°C	3.3V	70 MIPS (N = 1, N2 = 2, N3 = 1, M = 70, F_{VCO} = 560 MHz, F_{PLLO} = 280 MHz)
	18.8	25.0	mA	+25°C		
	18.8	31.1	mA	+85°C		
	22.8	37.0	mA	+125°C		
DC24a	23.0	26.1	mA	-40°C	3.3V	90 MIPS (N = 1, N2 = 2, N3 = 1, M = 90, F_{VCO} = 720 MHz, F_{PLLO} = 360 MHz)
	23.2	30.0	mA	+25°C		
	23.2	35.8	mA	+85°C		
	27.1	41.4	mA	+125°C		
DC25a	23.5	26.6	mA	-40°C	3.3V	100 MIPS (N = 1, N2 = 1, N3 = 1, M = 50, F_{VCO} = 400 MHz, F_{PLLO} = 400 MHz)
	23.7	30.4	mA	+25°C		
	23.7	36.4	mA	+85°C		
	27.6	41.9	mA	+125°C		

Note 1: I_{DD} is primarily a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption. The test conditions for all I_{DD} measurements are as follows:

- Oscillator is switched to EC+PLL mode in software
- CLKO is configured as an I/O input pin in the Configuration Word
- All I/O pins are configured as output low
- $\overline{\text{MCLR}}$ = V_{DD}, WDT and FSCM are disabled
- CPU, SRAM, program memory and data memory are operational
- No peripheral modules are operating or being clocked (all defined PMDx bits are set)
- CPU is executing `while(1)` statement
- JTAG is disabled