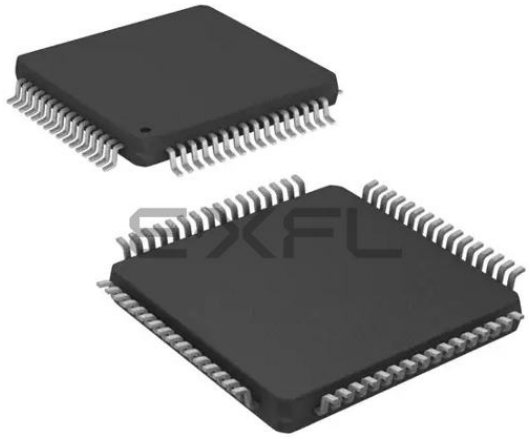
Microchip Technology - DSPIC33FJ64GS406-I/PT Datasheet



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
Speed	40 MIPs
Connectivity	I²C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, QEI, POR, PWM, WDT
Number of I/O	58
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 16x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/dspic33fj64gs406-i-pt

TO OUR VALUED CUSTOMERS

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

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

Most Current Data Sheet

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

http://www.microchip.com

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

Errata

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

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

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

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

Customer Notification System

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

TABLE 4-1: CPU CORE REGISTER MAP

File Name	SFR Addr	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	All Resets
WREG0	0000	Working Register 0																0000
WREG1	0002	Working Register 1																0000
WREG2	0004	Working Register 2																0000
WREG3	0006	Working Register 3																0000
WREG4	0008	Working Register 4																0000
WREG5	000A	Working Register 5																0000
WREG6	000C	Working Register 6																0000
WREG7	000E	Working Register 7																0000
WREG8	0010	Working Register 8																0000
WREG9	0012	Working Register 9																0000
WREG10	0014	Working Register 10																0000
WREG11	0016	Working Register 11																0000
WREG12	0018	Working Register 12																0000
WREG13	001A	Working Register 13																0000
WREG14	001C	Working Register 14																0000
WREG15	001E	Working Register 15																0800
SPLIM	0020	Stack Pointer Limit Register																xxxx
ACCAL	0022	ACCAL																xxxx
ACCAH	0024	ACCAH																xxxx
ACCAU	0026	ACCA<39>	ACCA<39>	ACCA<39>	ACCA<39>	ACCA<39>	ACCA<39>	ACCA<39>	ACCA<39>	ACCAU								xxxx
ACCBL	0028	ACCBL																xxxx
ACCBH	002A	ACCBH																xxxx
ACCBU	002C	ACCB<39>	ACCB<39>	ACCB<39>	ACCB<39>	ACCB<39>	ACCB<39>	ACCB<39>	ACCB<39>	ACCBU								xxxx
PCL	002E	Program Counter Low Byte Register																0000
PCH	0030	—	—	—	—	—	—	—	—	Program Counter High Byte Register								0000
TBLPAG	0032	—	—	—	—	—	—	—	—	Table Page Address Pointer Register								0000
PSVPAG	0034	—	—	—	—	—	—	—	—	Program Memory Visibility Page Address Pointer Register								0000
RCOUNT	0036	REPEAT Loop Counter Register																xxxx
DCOUNT	0038	DCOUNT<15:0>																xxxx
DOSTARTL	003A	DOSTARTL<15:1>															0	xxxx
DOSTARTH	003C	—	—	—	—	—	—	—	—	—	—	DOSTARTH<5:0>						00xx
DOENDL	003E	DOENDL<15:1>															0	xxxx
DOENDH	0040	—	—	—	—	—	—	—	—	—	—	DOENDH						00xx
SR	0042	OA	OB	SA	SB	OAB	SAB	DA	DC	IPL2	IPL1	IPL0	RA	N	OV	Z	C	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 4-21: HIGH-SPEED PWM GENERATOR 5 REGISTER MAP

File Name	SFR Addr	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	All Resets
PWMCON5	04A0	FLTSTAT	CLSTAT	TRGSTAT	FLTIEN	CLIEN	TRGIEN	ITB	MDCS	DTC1	DTC0	DTCP	—	MTBS	CAM	XPRES	IUE	0000
IOCON5	04A2	PENH	PENL	POLH	POLL	PMOD1	PMOD0	OVRENH	OVRENL	OVRDAT1	OVRDAT0	FLTDAT1	FLTDAT0	CLDAT1	CLDAT0	SWAP	OSYNC	0000
FCLCON5	04A4	IFLTMOD	CLSRC4	CLSRC3	CLSRC2	CLSRC1	CLSRC0	CLPOL	CLMOD	FLTSRC4	FLTSRC3	FLTSRC2	FLTSRC1	FLTSRC0	FLTPOL	FLTMOD1	FLTMOD0	0000
PDC5	04A6	PDC5<15:0>																0000
PHASE5	04A8	PHASE5<15:0>																0000
DTR5	04AA	—	—	DTR5<13:0>														0000
ALTDTR5	04AA	—	—	ALTDTR5<13:0>														0000
SDC5	04AE	SDC5<15:0>																0000
SPHASE5	04B0	SPHASE5<15:0>																0000
TRIG5	04B2	TRGCMP<12:0>													—	—	—	0000
TRGCON5	04B4	TRGDIV3	TRGDIV2	TRGDIV1	TRGDIV0	—	—	—	—	DTM	—	TRGSTRT5	TRGSTRT4	TRGSTRT3	TRGSTRT2	TRGSTRT1	TRGSTRT0	0000
STRIG5	04B6	STRGCMP<12:0>													—	—	—	0000
PWMCAP5	04B8	PWMCAP<12:0>													—	—	—	0000
LEBCON5	04BA	PHR	PHF	PLR	PLF	FLTLEBEN	CLLEBEN	—	—	—	—	BCH	BCL	BPHH	BPHL	BPLH	BPLL	0000
LEBDLY5	04BC	—	—	—	—	LEB<8:0>									—	—	—	0000
AUXCON5	04BE	HRPDIS	HRDDIS	—	—	BLANKSEL3	BLANKSEL2	BLANKSEL1	BLANKSEL0	—	—	CHOPSEL3	CHOPSEL2	CHOPSEL1	CHOPSEL0	CHOPHEN	CHOPLEN	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 4-32: HIGH-SPEED 10-BIT ADC REGISTER MAP FOR dsPIC33FJ32GS610 AND dsPIC33FJ64GS610 DEVICES ONLY (CONTINUED)

File Name	SFR Addr	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	All Resets
ADCBUF22	036C	ADC Data Buffer 22																xxxx
ADCBUF23	036E	ADC Data Buffer 23																xxxx
ADCBUF24	0370	ADC Data Buffer 24																xxxx
ADCBUF25	0372	ADC Data Buffer 25																xxxx

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 4-41: PORTA REGISTER MAP FOR dsPIC33FJ32GS610 AND dsPIC33FJ64GS610 DEVICES

File Name	SFR Addr	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	All Resets
TRISA	02C0	TRISA<15:14>		—	—	—	TRISA<10:9>		—	TRISA<7:0>								C6FF
PORTA	02C2	RA<15:14>		—	—	—	RA<10:9>		—	RA<7:0>								xxxx
LATA	02C4	LATA<15:14>		—	—	—	LATA<10:9>		—	LATA<7:0>								0000
ODCA	02C6	ODCA<15:14>		—	—	—	ODCA<10:9>		—	—	—	ODCA<5:4>		—	—	ODCA<1:0>		0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 4-42: PORTA REGISTER MAP FOR dsPIC33FJ32GS608 AND dsPIC33FJ64GS608 DEVICES

File Name	SFR Addr	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	All Resets
TRISA	02C0	TRISA<15:14>		—	—	—	TRISA<10:9>		—	—	—	—	—	—	—	—	—	C600
PORTA	02C2	RA<15:14>		—	—	—	RA<10:9>		—	—	—	—	—	—	—	—	—	xxxx
LATA	02C4	LATA<15:14>		—	—	—	LATA<10:9>		—	—	—	—	—	—	—	—	—	0000
ODCA	02C6	ODCA<15:14>		—	—	—	ODCA<10:9>		—	—	—	—	—	—	—	—	—	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 4-43: PORTB REGISTER MAP

File Name	SFR Addr	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	All Resets
TRISB	02C8	TRISB<15:0>																FFFF
PORTB	02CA	RB<15:0>																xxxx
LATB	02CC	LATB<15:0>																0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 4-44: PORTC REGISTER MAP FOR dsPIC33FJ32GS610 AND dsPIC33FJ64GS610 DEVICES

File Name	SFR Addr	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	All Resets
TRISC	02D0	TRISC<15:12>				—	—	—	—	—	—	—	TRISC<4:1>				—	F01E
PORTC	02D2	RC<15:12>				—	—	—	—	—	—	—	RC<4:1>				—	xxxx
LATC	02D4	LATC<15:12>				—	—	—	—	—	—	—	LATC<4:1>				—	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 4-53: PORTF REGISTER MAP FOR dsPIC33FJ32GS406/606 AND dsPIC33FJ64GS406/606 DEVICES

File Name	SFR Addr	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	All Resets
TRISF	02E8	—	—	—	—	—	—	—	—	—	TRISF<6:0>							007F
PORTF	02EA	—	—	—	—	—	—	—	—	—	RF<6:0>							xxxx
LATF	02EC	—	—	—	—	—	—	—	—	—	LATF<6:0>							0000
ODCF	02EE	—	—	—	—	—	—	—	—	—	ODCF6	—	—	ODCF<3:1>			—	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 4-54: PORTG REGISTER MAP FOR dsPIC33FJ32GS610 AND dsPIC33FJ64GS610 DEVICES

File Name	SFR Addr	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	All Resets
TRISG	02F0	TRISG<15:12				—	—	TRISG<9:6>				—	—	TRISG<3:0>				F3CF
PORTG	02F2	RG<15:12>				—	—	RG<9:6>				—	—	RG<3:0>				xxxx
LATG	02F4	LATG<15:12>				—	—	LATG<9:6>				—	—	LATG<3:0>				0000
ODCG	02F6	ODCG<15:12>				—	—	ODCG<9:6>				—	—	ODCG<3:0>				0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 4-55: PORTG REGISTER MAP FOR dsPIC33FJ32GS608 AND dsPIC33FJ64GS608 DEVICES

File Name	SFR Addr	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	All Resets
TRISG	02F0	—	—	—	—	—	—	TRISG<9:6>				—	—	TRISG<3:0>				03CF
PORTG	02F2	—	—	—	—	—	—	RG<9:6>				—	—	RG<3:0>				xxxx
LATG	02F4	—	—	—	—	—	—	LATG<9:6>				—	—	LATG<3:0>				0000
ODCG	02F6	—	—	—	—	—	—	ODCG<9:6>				—	—	ODCG<3:0>				0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 4-56: PORTG REGISTER MAP FOR dsPIC33FJ32GS406/606 AND dsPIC33FJ64GS406/606 DEVICES

File Name	SFR Addr	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	All Resets
TRISG	02F0	—	—	—	—	—	—	TRISG<9:6>				—	—	TRISG<3:2>		—	—	03CC
PORTG	02F2	—	—	—	—	—	—	RG<9:6>				—	—	RG<3:2>		—	—	xxxx
LATG	02F4	—	—	—	—	—	—	LATG<9:6>				—	—	LATG<3:2>		—	—	0000
ODCG	02F6	—	—	—	—	—	—	ODCG<9:6>				—	—	ODCG<3:2>		—	—	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

EXAMPLE 5-2: LOADING THE WRITE BUFFERS

```
; Set up NVMCON for row programming operations
        MOV     #0x4001, W0               ;
        MOV     W0, NVMCON                ; Initialize NVMCON
; Set up a pointer to the first program memory location to be written
; program memory selected, and writes enabled
        MOV     #0x0000, W0               ;
        MOV     W0, TBLPAG                ; Initialize PM Page Boundary SFR
        MOV     #0x6000, W0               ; An example program memory address
; Perform the TBLWT instructions to write the latches
; 0th_program_word
        MOV     #LOW_WORD_0, W2           ;
        MOV     #HIGH_BYTE_0, W3          ;
        TBLWTL  W2, [W0]                  ; Write PM low word into program latch
        TBLWTH  W3, [W0++]                ; Write PM high byte into program latch
; 1st_program_word
        MOV     #LOW_WORD_1, W2           ;
        MOV     #HIGH_BYTE_1, W3          ;
        TBLWTL  W2, [W0]                  ; Write PM low word into program latch
        TBLWTH  W3, [W0++]                ; Write PM high byte into program latch
;  2nd_program_word
        MOV     #LOW_WORD_2, W2           ;
        MOV     #HIGH_BYTE_2, W3          ;
        TBLWTL  W2, [W0]                  ; Write PM low word into program latch
        TBLWTH  W3, [W0++]                ; Write PM high byte into program latch
        •
        •
        •
; 63rd_program_word
        MOV     #LOW_WORD_31, W2          ;
        MOV     #HIGH_BYTE_31, W3         ;
        TBLWTL  W2, [W0]                  ; Write PM low word into program latch
        TBLWTH  W3, [W0++]                ; Write PM high byte into program latch
```

EXAMPLE 5-3: INITIATING A PROGRAMMING SEQUENCE

```
        DISI    #5                        ; Block all interrupts with priority <7
                                          ; for next 5 instructions
        MOV     #0x55, W0
        MOV     W0, NVMKEY                ; Write the 55 key
        MOV     #0xAA, W1                 ;
        MOV     W1, NVMKEY                ; Write the AA key
        BSET    NVMCON, #WR               ; Start the erase sequence
        NOP                               ; Insert two NOPs after the
        NOP                               ; erase command is asserted
```

REGISTER 8-4: DMAxSTB: DMA CHANNEL x RAM START ADDRESS OFFSET REGISTER B

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STB<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
STB<7:0>							
bit 7							bit 0

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

bit 15-0 **STB<15:0>:** Secondary DMA RAM Start Address bits (source or destination)

REGISTER 8-5: DMAxPAD: DMA CHANNEL x PERIPHERAL ADDRESS REGISTER(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PAD<15:8>(2)							
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
PAD<7:0>(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 15-0 **PAD<15:0>:** Peripheral Address Register bits(2)

Note 1: If the channel is enabled (i.e., active), writes to this register may result in unpredictable behavior of the DMA channel and should be avoided.

2: See Table 8-1 for a complete list of peripheral addresses.

NOTES:

The Timer2/3/4/5 modules can operate in one of the following modes:

- Timer mode
- Gated Timer mode
- Synchronous Counter mode

In Timer and Gated Timer modes, the input clock is derived from the internal instruction cycle clock (FCY). In Synchronous Counter mode, the input clock is derived from the external clock input at the TxCK pin.

The timer modes are determined by the following bits:

- TCS (TxCON<1>): Timer Clock Source Control bit
- TGATE (TxCON<6>): Timer Gate Control bit

Timer control bit settings for different operating modes are given in the Table 13-1.

TABLE 13-1: TIMER MODE SETTINGS

Mode	TCS	TGATE
Timer	0	0
Gated Timer	0	1
Synchronous Counter	1	x

13.1 16-Bit Operation

To configure any of the timers for individual 16-bit operation:

1. Clear the T32 bit corresponding to that timer.
2. Select the timer prescaler ratio using the TCKPS<1:0> bits.
3. Set the Clock and Gating modes using the TCS and TGATE bits.
4. Load the timer period value into the PRx register.
5. If interrupts are required, set the interrupt enable bit, TxIE. Use the priority bits, TxIP<2:0>, to set the interrupt priority.
6. Set the TON bit.

13.2 32-Bit Operation

A 32-bit timer module can be formed by combining a Type B and a Type C 16-bit timer module. For 32-bit timer operation, the T32 control bit in the Type B Timer Control (TxCON<3>) register must be set. The Type C timer holds the most significant word (msw) and the Type B timer holds the least significant word (lsw) for 32-bit operation.

When configured for 32-bit operation, only the Type B Timerx Control (TxCON) register bits are required for setup and control while the Type C Timer Control register bits are ignored (except the TSIDL bit).

For interrupt control, the combined 32-bit timer uses the interrupt enable, interrupt flag and interrupt priority control bits of the Type C timer. The interrupt control and status bits for the Type B timer are ignored during 32-bit timer operation.

The timers that can be combined to form a 32-bit timer are listed in Table 13-2.

TABLE 13-2: 32-BIT TIMER

Type B Timer (lsw)	Type C Timer (msw)
Timer2	Timer3
TImer4	Timer5

A block diagram representation of the 32-bit timer module is shown in Figure 13-3. The 32-timer module can operate in one of the following modes:

- Timer mode
- Gated Timer mode
- Synchronous Counter mode

To configure the timer features for 32-bit operation:

1. Set the T32 control bit.
2. Select the prescaler ratio for Timer2 using the TCKPS<1:0> bits.
3. Set the Clock and Gating modes using the corresponding TCS and TGATE bits.
4. Load the timer period value. PR3 contains the most significant word of the value, while PR2 contains the least significant word.
5. If interrupts are required, set the interrupt enable bit, T3IE. Use the priority bits, T3IP<2:0>, to set the interrupt priority. While Timer2 controls the timer, the interrupt appears as a Timer3 interrupt.
6. Set the corresponding TON bit.

14.0 INPUT CAPTURE

Note 1: This data sheet summarizes the features of the dsPIC33FJ32GS406/606/608/610 and dsPIC33FJ64GS406/606/608/610 families of devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to **"Input Capture"** (DS70198) in the *"dsPIC33/PIC24 Family Reference Manual"*, which is available from the Microchip web site (www.microchip.com). The information in this data sheet supersedes the information in the FRM.

2: Some registers and associated bits described in this section may not be available on all devices. Refer to **Section 4.0 "Memory Organization"** in this data sheet for device-specific register and bit information.

The input capture module is useful in applications requiring frequency (period) and pulse measurement. The dsPIC33FJ32GS406/606/608/610 and dsPIC33FJ64GS406/606/608/610 devices support up to two input capture channels.

The input capture module captures the 16-bit value of the selected Time Base register when an event occurs at the ICx pin. The events that cause a capture event are listed below in three categories:

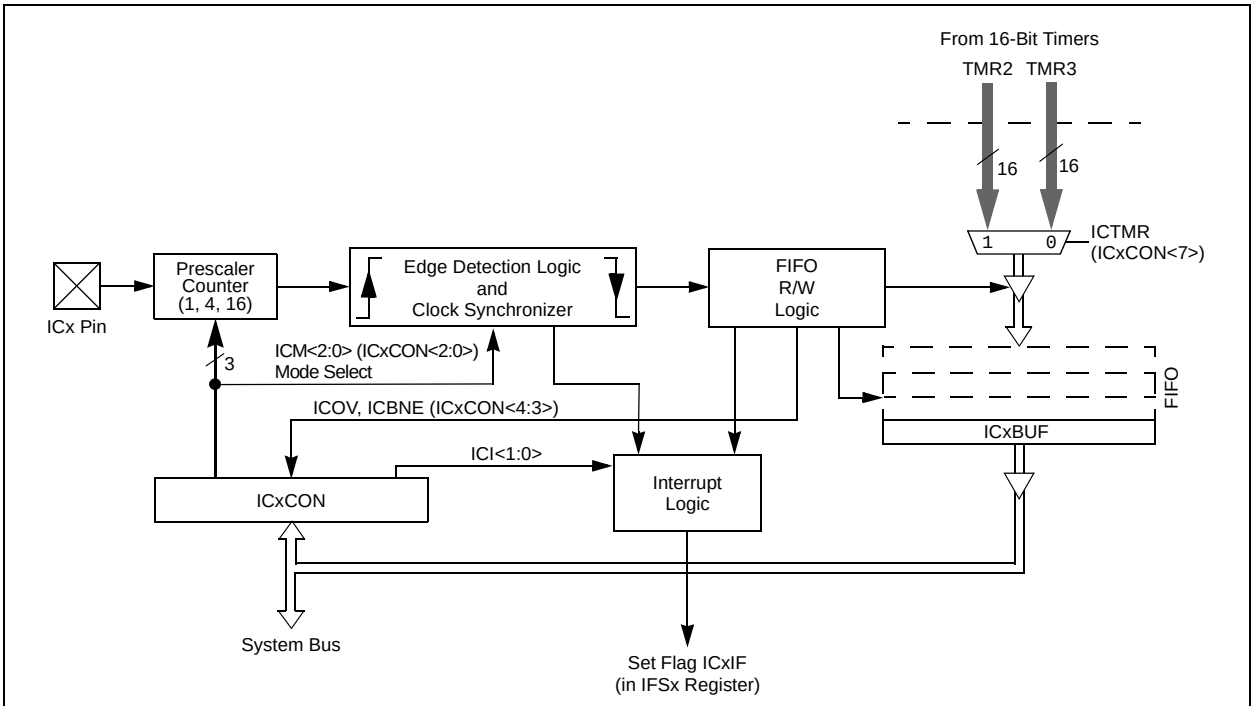
- Simple Capture Event modes:
  - Capture timer value on every falling edge of input at ICx pin
  - Capture timer value on every rising edge of input at ICx pin
- Capture Timer Value on Every Edge (rising and falling)
- Prescaler Capture Event modes:
  - Capture timer value on every 4th rising edge of input at ICx pin
  - Capture timer value on every 16th rising edge of input at ICx pin

Each input capture channel can select one of the two 16-bit timers (Timer2 or Timer3) for the time base. The selected timer can use either an internal or external clock.

Other operational features include:

- Device Wake-up from Capture Pin during CPU Sleep and Idle modes
- Interrupt on Input Capture Event
- 4-Word FIFO Buffer for Capture Values
  - Interrupt optionally generated after 1, 2, 3 or 4 buffer locations are filled
- Use of Input Capture to provide Additional Sources of External Interrupts

FIGURE 14-1: INPUT CAPTURE x BLOCK DIAGRAM



Note 1: An 'x' in a signal, register or bit name denotes the number of the input capture channel.

REGISTER 20-1: UxMODE: UARTx MODE REGISTER (CONTINUED)

bit 4 **URXINV:** Receive Polarity Inversion bit
1 = UxRX Idle state is '0'
0 = UxRX Idle state is '1'

bit 3 **BRGH:** High Baud Rate Enable bit
1 = BRG generates 4 clocks per bit period (4x baud clock, High-Speed mode)
0 = BRG generates 16 clocks per bit period (16x baud clock, Standard mode)

bit 2-1 **PDSEL<1:0>:** Parity and Data Selection bits
11 = 9-bit data, no parity
10 = 8-bit data, odd parity
01 = 8-bit data, even parity
00 = 8-bit data, no parity

bit 0 **STSEL:** Stop Bit Selection bit
1 = Two Stop bits
0 = One Stop bit

Note 1: Refer to **"UART"** (DS70188) in the *"dsPIC33/PIC24 Family Reference Manual"* for information on enabling the UART module for receive or transmit operation. That section of the manual is available on the Microchip web site: www.microchip.com.

2: This feature is only available for the 16x BRG mode (BRGH = 0).

21.0 ENHANCED CAN (ECAN™) MODULE

Note 1: This data sheet summarizes the features of the dsPIC33FJ32GS406/606/608/610 and dsPIC33FJ64GS406/606/608/610 families of devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to **"ECAN™"** (DS70185) in the *dsPIC33/PIC24 Family Reference Manual*, which is available from the Microchip web site (www.microchip.com). The information in this data sheet supersedes the information in the FRM.

2: Some registers and associated bits described in this section may not be available on all devices. Refer to **Section 4.0 "Memory Organization"** in this data sheet for device-specific register and bit information.

21.1 Overview

The Enhanced Controller Area Network (ECAN™) module is a serial interface, useful for communicating with other ECAN modules or microcontroller devices. This interface/protocol was designed to allow communications within noisy environments. The dsPIC33FJ64GS606/608/610 devices contain one ECAN module.

The ECAN module is a communication controller implementing the CAN 2.0 A/B protocol, as defined in the BOSCH CAN specification. The module supports CAN 1.2, CAN 2.0A, CAN 2.0B Passive and CAN 2.0B Active versions of the protocol. The module implementation is a full CAN system. The CAN specification is not covered within this data sheet. The reader can refer to the BOSCH CAN specification for further details.

The module features are as follows:

- Implementation of the CAN Protocol, CAN 1.2, CAN 2.0A and CAN 2.0B
- Standard and Extended Data Frames
- 0-8 Bytes Data Length
- Programmable Bit Rate, up to 1 Mbit/sec
- Automatic Response to Remote Transmission Requests
- Up to 8 Transmit Buffers with Application-Specified Prioritization and Abort Capability (each buffer can contain up to 8 bytes of data)
- Up to 32 Receive Buffers (each buffer can contain up to 8 bytes of data)
- Up to 16 Full (Standard/Extended Identifier) Acceptance Filters
- Three Full Acceptance Filter Masks
- DeviceNet™ Addressing Support
- Programmable Wake-up Functionality with Integrated Low-Pass Filter
- Programmable Loopback mode Supports Self-Test Operation
- Signaling via Interrupt Capabilities for all CAN Receiver and Transmitter Error States
- Programmable Clock Source
- Programmable Link to Input Capture module (IC2 for CAN1) for Time-Stamping and Network Synchronization
- Low-Power Sleep and Idle mode

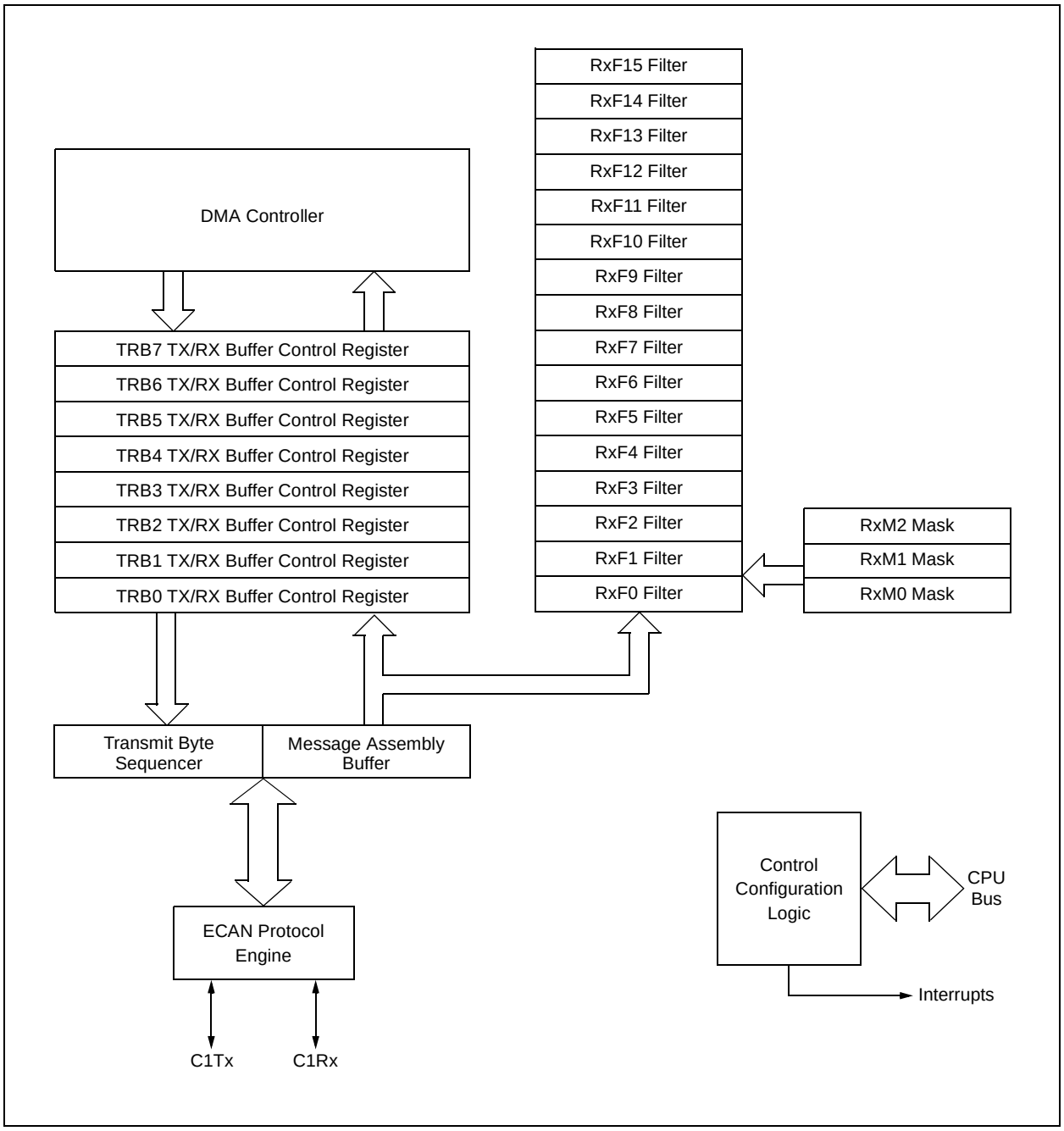
The CAN bus module consists of a protocol engine and message buffering/control. The CAN protocol engine handles all functions for receiving and transmitting messages on the CAN bus. Messages are transmitted by first loading the appropriate data registers. Status and errors can be checked by reading the appropriate registers. Any message detected on the CAN bus is checked for errors and then matched against filters to see if it should be received and stored in one of the receive registers.

21.2 Frame Types

The CAN module transmits various types of frames which include data messages, or remote transmission requests initiated by the user, as other frames that are automatically generated for control purposes. The following frame types are supported:

- Standard Data Frame: A standard data frame is generated by a node when the node wishes to transmit data. It includes an 11-bit Standard Identifier (SID), but not an 18-bit Extended Identifier (EID).
- Extended Data Frame: An extended data frame is similar to a standard data frame, but includes an Extended Identifier as well.
- Remote Frame: It is possible for a destination node to request the data from the source. For this purpose, the destination node sends a remote frame with an identifier that matches the identifier of the required data frame. The appropriate data source node sends a data frame as a response to this remote request.
- Error Frame: An error frame is generated by any node that detects a bus error. An error frame consists of two fields: an error flag field and an error delimiter field.
- Overload Frame: An overload frame can be generated by a node as a result of two conditions. First, the node detects a dominant bit during interframe space which is an illegal condition. Second, due to internal conditions, the node is not yet able to start reception of the next message. A node can generate a maximum of 2 sequential overload frames to delay the start of the next message.
- Interframe Space: Interframe space separates a proceeding frame (of whatever type) from a following data or remote frame.

FIGURE 21-1: ECANx MODULE BLOCK DIAGRAM



REGISTER 21-10: CxCFG2: ECANx BAUD RATE CONFIGURATION REGISTER 2

U-0	R/W-x	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	WAKFIL	—	—	—	SEG2PH2	SEG2PH1	SEG2PH0
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
SEG2PHTS	SAM	SEG1PH2	SEG1PH1	SEG1PH0	PRSEG2	PRSEG1	PRSEG0
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 **Unimplemented:** Read as ‘0’

bit 14 **WAKFIL:** Select ECAN Bus Line Filter for Wake-up bit
1 = Uses ECAN bus line filter for wake-up
0 = ECAN bus line filter is not used for wake-up

bit 13-11 **Unimplemented:** Read as ‘0’

bit 10-8 **SEG2PH<2:0>:** Phase Segment 2 bits
111 = Length is 8 x T_Q
•
•
•
000 = Length is 1 x T_Q

bit 7 **SEG2PHTS:** Phase Segment 2 Time Select bit
1 = Freely programmable
0 = Maximum of SEG1PHx bits or Information Processing Time (IPT), whichever is greater

bit 6 **SAM:** Sample of the ECAN Bus Line bit
1 = Bus line is sampled three times at the sample point
0 = Bus line is sampled once at the sample point

bit 5-3 **SEG1PH<2:0>:** Phase Segment 1 bits
111 = Length is 8 x T_Q
•
•
•
000 = Length is 1 x T_Q

bit 2-0 **PRSEG<2:0>:** Propagation Time Segment bits
111 = Length is 8 x T_Q
•
•
•
000 = Length is 1 x T_Q

REGISTER 21-19: CxFMSKSEL2: ECANx FILTER 15-8 MASK SELECTION REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F15MSK1	F15MSK0	F14MSK1	F14MSK0	F13MSK1	F13MSK0	F12MSK1	F12MSK0
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
F11MSK1	F11MSK0	F10MSK1	F10MSK0	F9MSK1	F9MSK0	F8MSK1	F8MSK0
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 **F15MSK<1:0>:** Mask Source for Filter 15 bits
11 = Reserved
10 = Acceptance Mask 2 registers contain mask
01 = Acceptance Mask 1 registers contain mask
00 = Acceptance Mask 0 registers contain mask

bit 13-12 **F14MSK<1:0>:** Mask Source for Filter 14 bits (same values as bits<15:14>)

bit 11-10 **F13MSK<1:0>:** Mask Source for Filter 13 bits (same values as bits<15:14>)

bit 9-8 **F12MSK<1:0>:** Mask Source for Filter 12 bits (same values as bits<15:14>)

bit 7-6 **F11MSK<1:0>:** Mask Source for Filter 11 bits (same values as bits<15:14>)

bit 5-4 **F10MSK<1:0>:** Mask Source for Filter 10 bits (same values as bits<15:14>)

bit 3-2 **F9MSK<1:0>:** Mask Source for Filter 9 bits (same values as bits<15:14>)

bit 1-0 **F8MSK<1:0>:** Mask Source for Filter 8 bits (same values as bits<15:14>)

REGISTER 22-7: ADCPC1: ADC CONVERT PAIR CONTROL REGISTER 1 (CONTINUED)

bit 12-8 **TRGSRC3<4:0>:** Trigger 3 Source Selection bits

Selects trigger source for conversion of analog channels AN7 and AN6.
11111 = Timer2 period match
11110 = PWM Generator 8 current-limit ADC trigger
11101 = PWM Generator 7 current-limit ADC trigger
11100 = PWM Generator 6 current-limit ADC trigger
11011 = PWM Generator 5 current-limit ADC trigger
11010 = PWM Generator 4 current-limit ADC trigger
11001 = PWM Generator 3 current-limit ADC trigger
11000 = PWM Generator 2 current-limit ADC trigger
10111 = PWM Generator 1 current-limit ADC trigger
10110 = PWM Generator 9 secondary trigger is selected
10101 = PWM Generator 8 secondary trigger is selected
10100 = PWM Generator 7 secondary trigger is selected
10011 = PWM Generator 6 secondary trigger is selected
10010 = PWM Generator 5 secondary trigger is selected
10001 = PWM Generator 4 secondary trigger is selected
10000 = PWM Generator 3 secondary trigger is selected
01111 = PWM Generator 2 secondary trigger is selected
01110 = PWM Generator 1 secondary trigger is selected
01101 = PWM secondary Special Event Trigger is selected
01100 = Timer1 period match
01011 = PWM Generator 8 primary trigger is selected
01010 = PWM Generator 7 primary trigger is selected
01001 = PWM Generator 6 primary trigger is selected
01000 = PWM Generator 5 primary trigger is selected
00111 = PWM Generator 4 primary trigger is selected
00110 = PWM Generator 3 primary trigger is selected
00101 = PWM Generator 2 primary trigger is selected
00100 = PWM Generator 1 primary trigger is selected
00011 = PWM Special Event Trigger is selected
00010 = Global software trigger is selected
00001 = Individual software trigger is selected
00000 = No conversion is enabled

bit 7 **IRQEN2:** Interrupt Request Enable 2 bit

1 = Enables IRQ generation when requested conversion of Channels AN5 and AN4 is completed
0 = IRQ is not generated

bit 6 **PEND2:** Pending Conversion Status 2 bit

1 = Conversion of Channels AN5 and AN4 is pending; set when selected trigger is asserted
0 = Conversion is complete

bit 5 **SWTRG2:** Software Trigger 2 bit

1 = Starts conversion of AN5 and AN4 (if selected by the TRGSRCx<4:0> bits)**(1)**
This bit is automatically cleared by hardware when the PEND2 bit is set.
0 = Conversion has not started

Note 1: The trigger source must be set as an individual software trigger prior to setting this bit to '1'. If other conversions are in progress, the conversion is performed when the conversion resources are available.

REGISTER 22-10: ADCPC4: ADC CONVERT PAIR CONTROL REGISTER 4 (CONTINUED)

bit 4-0 **TRGSRC8<4:0>:** Trigger 8 Source Selection bits

Selects trigger source for conversion of Analog Channels AN17 and AN16.
11111 = Timer2 period match
11110 = PWM Generator 8 current-limit ADC trigger
11101 = PWM Generator 7 current-limit ADC trigger
11100 = PWM Generator 6 current-limit ADC trigger
11011 = PWM Generator 5 current-limit ADC trigger
11010 = PWM Generator 4 current-limit ADC trigger
11001 = PWM Generator 3 current-limit ADC trigger
11000 = PWM Generator 2 current-limit ADC trigger
10111 = PWM Generator 1 current-limit ADC trigger
10110 = PWM Generator 9 secondary trigger selected
10101 = PWM Generator 8 secondary trigger selected
10100 = PWM Generator 7 secondary trigger selected
10011 = PWM Generator 6 secondary trigger selected
10010 = PWM Generator 5 secondary trigger selected
10001 = PWM Generator 4 secondary trigger selected
10000 = PWM Generator 3 secondary trigger selected
01111 = PWM Generator 2 secondary trigger selected
01110 = PWM Generator 1 secondary trigger selected
01101 = PWM secondary Special Event Trigger selected
01100 = Timer1 period match
01011 = PWM Generator 8 primary trigger selected
01010 = PWM Generator 7 primary trigger selected
01001 = PWM Generator 6 primary trigger selected
01000 = PWM Generator 5 primary trigger selected
00111 = PWM Generator 4 primary trigger selected
00110 = PWM Generator 3 primary trigger selected
00101 = PWM Generator 2 primary trigger selected
00100 = PWM Generator 1 primary trigger selected
00011 = PWM Special Event Trigger selected
00010 = Global software trigger selected
00001 = Individual software trigger selected
00000 = No conversion is enabled

Note 1: The trigger source must be set as an individual software trigger prior to setting this bit to ‘1’. If other conversions are in progress, the conversion is performed when the conversion resources are available.

25.0 INSTRUCTION SET SUMMARY

Note: This data sheet summarizes the features of the dsPIC33FJ32GS406/606/608/610 and dsPIC33FJ64GS406/606/608/610 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"dsPIC33/PIC24 Family Reference Manual"*. Please see the Microchip web site (www.microchip.com) for the latest *"dsPIC33F/PIC24H Family Reference Manual"* sections. The information in this data sheet supersedes the information in the FRM.

The dsPIC33F instruction set is identical to that of the dsPIC30F.

Most instructions are a single program memory word (24 bits). Only three instructions require two program memory locations.

Each single-word instruction is a 24-bit word, divided into an 8-bit opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into five basic categories:

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

Table 25-1 shows the general symbols used in describing the instructions.

The dsPIC33F instruction set summary in Table 25-2 lists all the instructions, along with the status flags affected by each instruction.

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

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

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

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

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

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

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

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

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

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

The MAC class of DSP instructions can use some of the following operands:

- The accumulator (A or B) to be used (required operand)
- The W registers to be used as the two operands
- The X and Y address space prefetch operations
- The X and Y address space prefetch destinations
- The accumulator write-back destination

The other DSP instructions do not involve any multiplication and can include:

- The accumulator to be used (required)
- The source or destination operand (designated as Wso or Wdo, respectively) with or without an address modifier
- The amount of shift specified by a W register, 'Wn', or a literal value

The control instructions can use some of the following operands:

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

TABLE B-3: MAJOR SECTION UPDATES (CONTINUED)

Section Name	Update Description
Section 27.0 "Electrical Characteristics" (Continued)	Updated the Timer1, Timer2, and Timer3 External Clock Timing Requirements (see Table 27-23, Table 27-24, and Table 27-25). Updated the Simple OC/PWM Mode Timing Requirements (see Table 27-28). Updated all SPI Timing specifications (see Figure 27-11-Figure 27-18 and Table 27-30-Table 27-37). Added Note 2 to the 10-bit High-Speed ADC Module Specifications (see Table 27-40). Added Note 2 to the 10-bit High-Speed ADC Module Timing Requirements (see Table 27-41). Added parameter DA08 to the DAC Module Specifications (see Table 27-43). Updated parameter DA16 in the DAC Output Buffer Specifications (see Table 27-44). Added DMA Read/Write Timing Requirements (see Table 27-49).
Section 28.0 "50 MIPS Electrical Characteristics"	Added new chapter with electrical specifications for 50 MIPS devices.
Section 29.0 "DC and AC Device Characteristics Graphs"	Added new chapter.

Revision E (October 2012)

This revision removes the Preliminary watermark and includes minor typographical and formatting changes throughout the data sheet.

Revision F (July 2014)

Changes CHOP bit to CHOPCLK in the High Speed PWM Register Map and CHOPCLK PWMCHOP Clock Generator Register (see Register 4-16 and Register 16-9).

Changes values in the Minimum Row Write Time and Maximum Row Write time equation examples (see Equation 5-2 and Equation 5-3).

Adds the Oscillator Delay table (see Table 6-2).

Updates TUN bit ranges in the OSCTUN: Oscillator Tuning Register (see Register 9-4).

Updates the Type C Timer Block Diagram (see Figure 13-2).

Adds Note 1 to the CxFCTRL: ECANx FIFO Control Register (see Register 21-4).

Adds Note 10 to the DC Characteristics: I/O Pin Input Specifications (see Table 27-9).

Updates values in the DC Characteristics: Program Memory Table (see Table 27-12).

Adds Register 29-7 through Register 29-12 to **Section 29.0 "DC and AC Device Characteristics Graphs"**

Also includes minor typographical and formatting changes throughout the data sheet.