MICROCHIP

Section 33. Charge Time Measurement Unit (CTMU)

HIGHLIGHTS

This section of the manual contains the following major topics:

Note:	This family reference manual section is meant to serve as a complement to device data sheets. Depending on the device variant, this manual section may not apply to all dsPIC33E/PIC24E devices. Please consult the note at the beginning of the **"Charge Time Measurement Unit (CTMU)"** chapter in the current device data sheet to check whether this document supports the device you are using. Device data sheets and family reference manual sections are available for download from the Microchip Worldwide Web site at: http://www.microchip.com

33.1 INTRODUCTION

The Charge Time Measurement Unit (CTMU) is a flexible analog module that provides accurate differential time measurement between pulse sources and asynchronous pulse generation. By working with other on-chip analog modules, the CTMU can be used to precisely measure time, capacitance, relative changes in capacitance or generate output pulses with a specific time delay. The CTMU is ideal for interfacing with capacitive-based sensors.

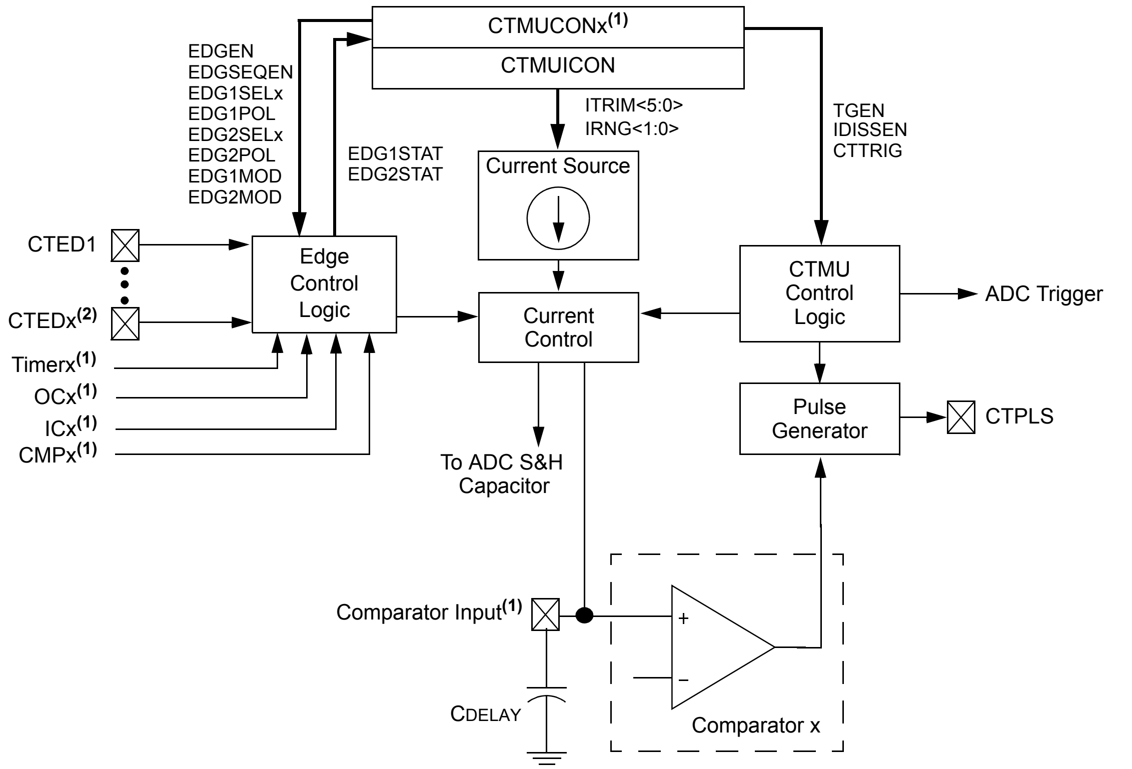
The CTMU module includes the following key features:

- Up to 16 channels available for capacitive or time measurement input
- On-chip precision current source
- Four-edge input trigger sources
- Polarity control for each edge source
- Control of edge sequence
- Control of response to edges
- High precision time measurement
- Time delay of external or internal signal asynchronous to system clock

The CTMU module works in conjunction with the Analog-to-Digital Converter (ADC) module to provide input channels for time or charge measurement. When configured for time delay, and depending on the device variant, the CTMU is connected to one of the analog comparators. The level-sensitive input edge sources can be selected from the following sources: two external inputs, Timers, Output Compare, Input Capture, and Comparator modules. For device-specific information on available input sources, refer to the specific device data sheet.

A block diagram of the CTMU is illustrated in Figure 33-1.

Figure 33-1: CTMU Block Diagram



Note 1: Refer to the specific device data sheet to determine the inputs that are available on your device.

2: Refer to the specific device data sheet to determine the number of available edge trigger inputs for your device.

33.2 REGISTERS

Depending on the device variant, there are up to three control registers available for the CTMU: CTMUCON1, CTMUCON2, and CTMUICON.

The CTMUCON1 and CTMUCON2 registers (Register 33-1 and Register 33-2) contain control bits for configuring the CTMU module edge source selection, edge source polarity selection, edge sequencing, analog-to-digital trigger, analog circuit capacitor discharge and enables. The CTMUICON register (Register 33-3) has bits for selecting the current source range and current source trim.

Register 33-1: CTMUCON1: CTMU Control Register 1(1)

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 7							bit 0

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

bit 15 **CTMUEN:** CTMU Enable bit
1 = Module is enabled
0 = Module is disabled

bit 14 **Unimplemented:** Read as ‘0’

bit 13 **CTMUSIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12 **TGEN:** Time Generation Enable bit
1 = Enables edge delay generation
0 = Disables edge delay generation

bit 10 **EDGEN:** Edge Enable bit
1 = Edges are not blocked
0 = Edges are blocked

bit 10 **EDGSEQEN:** Edge Sequence Enable bit
1 = Edge 1 event must occur before Edge 2 event can occur
0 = No edge sequence is needed

bit 9 **IDISSEN:** Current Discharge Enable bit
1 = Analog current source output is grounded
0 = Analog current source output is not grounded

bit 8 **CTTRIG:** Trigger Control bit
1 = Trigger output is enabled
0 = Trigger output is disabled

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

Note 1: Refer to the specific device data sheet to determine whether this register is available on your particular device.

Register 33-2: CTMUCON2: CTMU Control Register 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
EDG1MOD	EDG1POL	EDG1SEL<3:0>(1)				EDG2STAT	EDG1STAT
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
EDG2MOD	EDG2POL	EDG2SEL<3:0>(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 **EDG1MOD:** Input mode selection bit
1 = Input is edge-sensitive
0 = Input is level-sensitive

bit 14 **EDG1POL:** Edge 1 Polarity Select bit
1 = Edge 1 programmed for a positive level response
0 = Edge 1 programmed for a negative level response

bit 13-10 **EDG1SEL<3:0>:** Edge 1 Source Select bits(1)
1111 = Edge 1 Source 15 selected
1110 = Edge 1 Source 14 selected
1101 = Edge 1 Source 13 selected
1100 = Edge 1 Source 12 selected
1011 = Edge 1 Source 11 selected
1010 = Edge 1 Source 10 selected
1001 = Edge 1 Source 9 selected
1000 = Edge 1 Source 8 selected
0111 = Edge 1 Source 7 selected
0110 = Edge 1 Source 6 selected
0101 = Edge 1 Source 5 selected
0100 = Edge 1 Source 4 selected
0011 = Edge 1 Source 3 selected
0010 = Edge 1 Source 2 selected
0001 = Edge 1 Source 1 selected
0000 = Edge 1 Source 0 selected

bit 9 **EDG2STAT:** Edge 2 Status bit
1 = Edge 2 event has occurred
0 = Edge 2 event has not occurred

bit 8 **EDG1STAT:** Edge 1 Status bit
1 = Edge 1 event has occurred
0 = Edge 1 event has not occurred

bit 7 **EDG2MOD:** Input Mode Selection bit
1 = Input is edge-sensitive
0 = Input is level-sensitive

bit 6 **EDG2POL:** Edge 2 Polarity Select bit
1 = Edge 2 programmed for a positive level response
0 = Edge 2 programmed for a negative level response

Note 1: Refer to the specific device data sheet for edge source types and assignments.

Register 33-2: CTMUCON2: CTMU Control Register 2 (Continued)

bit 5-2	**EDG2SEL<3:0>:** Edge 2 Source Select bits(1) 1111 = Edge 2 Source 15 selected 1110 = Edge 2 Source 14 selected 1101 = Edge 2 Source 13 selected 1100 = Edge 2 Source 12 selected 1011 = Edge 2 Source 11 selected 1010 = Edge 2 Source 10 selected 1001 = Edge 2 Source 9 selected 1000 = Edge 2 Source 8 selected 0111 = Edge 2 Source 7 selected 0110 = Edge 2 Source 6 selected 0101 = Edge 2 Source 5 selected 0100 = Edge 2 Source 4 selected 0011 = Edge 2 Source 3 selected 0010 = Edge 2 Source 2 selected 0001 = Edge 2 Source 1 selected 0000 = Edge 2 Source 0 selected
bit 1-0	**Unimplemented:** Read as '0'

Note 1: Refer to the specific device data sheet for edge source types and assignments.

Register 33-3: CTMUICON: CTMU Current Control Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ITRIM<5:0>						IRNG<1:0>(1)	
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 7							bit 0

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

bit 15-10 **ITRIM<5:0>:** Current Source Trim bits
011111 = Maximum positive change from nominal current
•
•
•
000001 = Minimum positive change from nominal current
000000 = Nominal current output specified by IRNG<1:0>
111111 = Minimum negative change from nominal current
•
•
•
100001 = Maximum negative change from nominal current

bit 9-8 **IRNG<1:0>:** Current Source Range Select bits(1)
11 = 100 × base current
10 = 10 × base current
01 = Base current level (0.55 μA nominal)
00 = Reserved

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

Note 1: Refer to the specific device data sheet to determine which setting is available on your particular device.

33

Charge Time Measurement Unit (CTMU)

33.3 CTMU OPERATION

The CTMU works by using a constant current source to charge a circuit. The type of circuit depends on the type of measurement is made. In the case of capacitance measurement, the current and the amount of time the current is applied to the circuit is fixed. The amount of voltage read by the ADC is then a measurement of the capacitance of the circuit. In the case of time measurement, the current and the capacitance of the circuit is fixed and charge time varies. In this case, the voltage read by the ADC is then representative of the amount of time elapsed from the time the current source starts and stops charging the circuit.

If the CTMU is used as a time delay, both capacitance and current source are fixed, and the voltage supplied to the comparator circuit also fixed. The delay of a signal is determined by the amount of time it takes the voltage to charge to the comparator threshold voltage.

33.3.1 Theory of Operation

The operation of the CTMU is based on the equation for charge, as shown in Equation 33-1.

Equation 33-1:

$$I = C \cdot \frac{dV}{dT}$$

The amount of charge measured in coulombs in a circuit is defined as current in amperes (I) multiplied by the amount of time in seconds that the current flows (t). Charge is also defined as the capacitance in farads (C) multiplied by the voltage of the circuit (V), as shown in Equation 33-2.

Equation 33-2:

$$I \cdot t = C \cdot V$$

The CTMU module provides a constant current source. The ADC is used to measure (V) in the equation, leaving two unknowns: capacitance (C) and time (t). Equation 33-2 can be used to calculate capacitance or time either using the known fixed capacitance of the circuit, as shown in Equation 33-3, or using a fixed time that the current source is applied to the circuit, as shown in Equation 33-4.

Equation 33-3:

$$t = \frac{(C \cdot V)}{I}$$

Equation 33-4:

$$C = \frac{(I \cdot t)}{V}$$

33.3.2 Current Source

At the core of the CTMU is a precision current source, designed to provide a constant reference for measurements. The level of current is user-selectable across three ranges or a total of two orders of magnitude, with the ability to trim the output in ±2% increments (nominal). The current range is selected by the IRNG<1:0> bits (CTMUICON<9:8>), with a value of '`00`' representing the lowest range.

Current trim is provided by the ITRIM<5:0> bits (CTMUICON<15:10>). These six bits allow trimming of the current source in steps of approximately 2% per step. One half of the range adjusts the current source positively and the other half reduces the current source. A value of '`000000`' is the neutral position (no change). A value of '`000000`' is the neutral position (no change). A value of '`100001`' (see **Note 1**) is the maximum negative adjustment (approximately -62%) and '`011111`' (see **Note 2**) is the maximum positive adjustment (approximately +62%).

> **Note 1:** The value '`100001`' = -31 & 0x3F →ITRIM = -31 * Delta I, approximately -62% of nominal.
>
> **2:** The value '`011111`' = +31 & 0x3F →ITRIM = +31 * Delta I, approximately +62% of nominal.

33.3.3 Edge Selection and Control

CTMU measurements are controlled by edge events occurring on the module's input channels. Each channel, referred to as Edge 1 and Edge 2, can be configured to receive input pulses from one of the edge input pins (CTEDx), and from some peripherals such as Timers, Output Compare Module, Input Capture, and Comparators. Refer to the specific device data sheet to determine which setting is available on each particular device. The input channels are level-sensitive, responding to the instantaneous level on the channel rather than a transition between levels. The inputs are selected using the EDG1SEL and EDG2SEL bit pairs (CTMUCON2<5:2> and CTMUCON2<13:9>).

In addition to source, each channel can be configured for event polarity using the EDGE2POL and EDGE1POL bits (CTMUCON2<14:6>). The input channels can also be filtered for an edge event sequence (Edge 1 occurring before Edge 2) by setting the EDGSEQEN bit (CTMUCON1<10>).

33.3.4 Edge Status

The CTMUCON register also contains two status bits, EDG2STAT and EDG1STAT (CTMUCON2<9:8>). Their primary function is to show if an edge response has occurred on the corresponding channel. The CTMU automatically sets a bit when an edge response is detected on its channel. The level-sensitive nature of the input channels also means that the status bits become set immediately if the channel's configuration changes and remains in the new state.

The module uses the edge status bits to control the current source output to external analog modules like ADC. Current is supplied to external modules only when one (but not both) of the status bits is set, and shuts current off when both the bits are either set or cleared. This allows the CTMU to measure current only during the interval between edges. After both status bits are set, it is necessary to clear them before another measurement is taken. If possible, both the bits should be cleared simultaneously to avoid re-enabling the CTMU current source.

In addition to being set by the CTMU hardware, the edge status bits can also be set by software. This is also the user's application to manually enable or disable the current source. Setting either one (but not both) of the bits enables the current source. Setting or clearing both bits at once disables the source.

33.3.5 Interrupts

The CTMU sets its interrupt flag, CTMUIF (IFS4<13>), whenever the current source is enabled and disabled. An interrupt is generated only if the corresponding interrupt enable bit, CTMUIE (IEC4<13>), is also set. If edge sequencing is not enabled (that is Edge 1 must occur before Edge 2), it is necessary to monitor the edge status bits and determine which edge occurred last and caused the interrupt.

33.4 CTMU MODULE INITIALIZATION

The following sequence is a general guideline that can be used to initialize the CTMU module:

1. Select the current source range using the IRNG<1:0> bits (CTMUICON<9:8>).
2. Adjust the current source trim using the ITRIM<5:0> bits (CTMUICON<15:10>).
3. Configure the edge input sources for Edge 1 and Edge 2 by setting the EDG1SEL and EDG2SEL bits (CTMUCON2<13:10> and CTMUCON2<5:2>).
4. Configure the input polarities for the edge inputs using the EDG1POL and EDG2POL bits (CTMUCON2<14:6>). The default configuration is for negative edge polarity (high-to-low transitions).
5. Enable edge sequencing using the EDGSEQEN bit (CTMUCON1<10>). By default, edge sequencing is disabled.
6. Select the operating mode (Measurement or Time Delay) with the TGEN bit (CTMUCON1<12>). The default mode is Time/Capacitance Measurement.
7. Configure the module to automatically trigger an analog-to-digital conversion when the second edge event has occurred using the CTTRIG bit (CTMUCON1<8>). The conversion trigger is disabled by default.
8. Discharge the connected circuit by setting the IDISSEN bit (CTMUCON1<9>); after waiting a sufficient time for the circuit to discharge, clear IDISSEN.
9. Disable the module by clearing the CTMUEN bit (CTMUCON1<15>).
10. Clear the Edge Status bits, EDG2STAT and EDG1STAT (CTMUCON2<9:8>).
11. Enable both edge inputs by setting the EDGEN bit (CTMUCON1<11>).
12. Enable the module by setting the CTMUEN bit (CTMUCON1<15>).

Depending on the type of measurement or pulse generation being performed, one or more additional modules may also need to be initialized and configured with the CTMU module:

- Edge Source Generation: In addition to the external edge input pins, other peripherals such as Timers, Output Compare, Input Capture, and Comparators can be used as edge sources for the CTMU
- Capacitance or Time Measurement: The CTMU module uses the ADC to measure the voltage across a capacitor that is connected to one of the analog input channels.
- Pulse Generation: When generating system clock independent output pulses, the CTMU module uses one analog Comparator module and the associated comparator voltage reference

For specific information on initializing these modules, refer to the applicable section of the *“dsPIC33E/PIC24E Family Reference Manual”* for the appropriate module.

33.5 CALIBRATING THE CTMU MODULE

The CTMU requires calibration for precise measurements of capacitance and time, and also for accurate time delay. If the application only requires measurement of a relative change in capacitance or time, calibration is usually not necessary. An example of this type of application would include a capacitive touch switch, in which the touch circuit has a baseline capacitance, and the added capacitance of the human body changes the overall capacitance of a circuit.

If actual capacitance or time measurement is required, two hardware calibrations must take place: the current source needs calibration to set it to a precise current, and the circuit being measured needs calibration to measure and/or nullify all other capacitance other than that to be measured.

33.5.1 Current Source Calibration

The current source onboard the CTMU module has a range of ±62% nominal for each of three current ranges. Therefore, for precise measurements, it is possible to measure and adjust this current source by placing a high precision resistor, R_{CAL}, onto a special analog channel. An example circuit is illustrated in Figure 33-2. The current source measurement is performed using the following steps:

1. Initialize the ADC.
2. Initialize the CTMU by configuring the module for Pulse Generation mode (TGEN = 1).
3. Enable the current source by setting EDG1STAT (CTMUCON1<8>).
4. Issue settling time delay.
5. Perform analog-to-digital conversion.
6. Calculate the current source current using $I = V/R_{CAL}$, where R_{CAL} is a high precision resistance and V is measured by performing an analog-to-digital conversion.

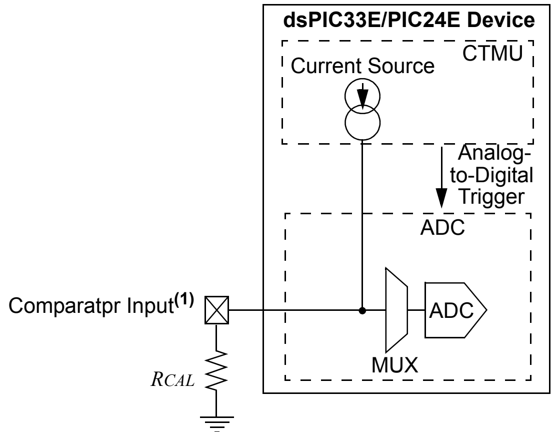
CTMU charge pump calibration should only be done on the special pin that connects to the Comparator module associated with the CTMU. Use of any other ANx pin will add around 2.5 kΩ of series resistance to the measurement from the analog multiplexer in front of the ADC. This effect can be ignored only for the smallest current settings. For larger current settings, the uncertainty in the extra series resistance overwhelms the overall resistance accuracy from the calibration resistor.

The CTMU current source may be trimmed with the trim bits in CTMUICON using an iterative process to get an exact desired current. Alternatively, the nominal value without adjustment may be used and it may be stored by the software for use in all subsequent capacitive or time measurements.

When the module is configured for pulse generation delay by setting the TGEN bit (CTMUCON1<12>), the internal current source is connected to one of the inputs of a Comparator module. Figure 33-2 illustrates the external connections for current source calibration and the relationship of the different analog modules required.

To calculate the value for R_{CAL}, the nominal current must be chosen, and then the resistance can be calculated. For example, if the ADC reference voltage is 3.3V, use 70% of full scale, or 2.31V as the desired approximate voltage to be read by the ADC. If the range of the CTMU current source is selected to be 0.55 µA, the resistor value needed is calculated as $R_{CAL} = 2.31\text{V}/0.55\ \mu\text{A}$, for a value of 4.2 MΩ. Similarly, if the current source is chosen to be 5.5 µA, R_{CAL} will be 420,000Ω, and 42,000Ω if the current source is set to 55 µA.

Figure 33-2: CTMU Current Source Calibration Circuit



Note 1: Refer to the **"Comparator"** and **"CTMU"** chapters in the specific device data sheet to determine which comparator and comparator input pin is used by the CTMU module.

A value of 70% full-scale voltage is chosen to ensure that the ADC was in a range that is well above the noise floor. If an exact current is chosen that is to incorporate the trimming bits from CTMUICON, the resistor value of RCAL may need to be adjusted accordingly. RCAL may be adjusted to allow for available resistor values. RCAL should be of the highest precision available, the amount of precision required for the circuit that the CTMU will be used to measure. A recommended minimum precision will be 0.1% tolerance.

The following examples show typical methods for performing a CTMU calibrations. Example 33-1 shows one approach for the actual current calibration. This method manually triggers the ADC and it is performed to demonstrate the entire stepwise process. It is also possible to automatically trigger the conversion by setting the CTTRIG bit (CTMUCON1<8>).

Note:	A complete MPLAB® project based on this calibration routine is available on the Microchip web site at: http://www.microchip.com/CodeExamplesByFunc.aspx. From the landing page, scroll down and select **Touch Sense (mTouch)** as the application.

Example 33-1: Current Calibration Routine

```
typedef enum _PLIB_CTMU_CURRENT_RANGE
{
    PLIB_CTMU_CurrentRange_Base      = 0x1,
    PLIB_CTMU_CurrentRange_10xBase   = 0x2,
    PLIB_CTMU_CurrentRange_100xBase  = 0x3,
    PLIB_CTMU_CurrentRange_1000xBase = 0x0, // PIC32
    PLIB_CTMU_CurrentRange_Disabled  = 0x0  // Others
} PLIB_CTMU_CURRENT_RANGE;

// TO DO: Setup Device

int main(void)
{
    char                ButtonMeasString[133];// UART output string
    unsigned long int  ADC_Sum;              // For averaging multiple ADC measurements
    unsigned short int iAvg,                 // Averaging index
                       Naverages = 1024,     // Number of averages < 2^22 (22=32-10 bits of ADC)
                       Log2Naverages = 10;   // Right shift equal to 1/Naverages
    short int          iTrim;                // Current trim index
    unsigned short int VmeasADC, VavgADC;    // Measured Voltages, 65536 = Full Scale

    // Fosc= Fin*M/(N1*N2), Fcy=Fosc/2
    // Fosc= 8.0*40/(2*2)= 80.000 MHz for 8.00 input clock
    /****************** Clock definitions *********************************/
    PLLFBD = PLLFBD_VALUE;                   // M=40
    CLKDIVbits.PLLPOST=0;                    // N1=2
    CLKDIVbits.PLLPRE=0;                     // N2=2
    OSCTUN=0;                                // Tune FRC oscillator, if FRC is used

    // TO DO: Clock switch to incorporate PLL
    // TO DO: Configure UART2 for diagnostic dumps to ProfiLab
    // TO DO: Setup Pins

    // CTMU Setup
    CTMUCON1bits.TGEN = 1;    // Enable direct output to PGED1/AN5/C1IN1-/RP35/RB3 pin
//  CTMUICONbits.IRNG = PLIB_CTMU_CurrentRange_Base;      // Current Range
//  CTMUICONbits.IRNG = PLIB_CTMU_CurrentRange_10xBase;   // Current Range
    CTMUICONbits.IRNG = PLIB_CTMU_CurrentRange_100xBase;  // Current Range
//  Turn on CTMU after setting current trim below

    // Comparator Setup
    PMD3bits.CMPMD = 1; // Disable comparator

    // ADC Setup
    AD1CON1bits.AD12B = 0; // 10-bit mode
    AD1CON2 = 0x0; // VR+ = AVDD, V- = AVSS, Don't scan, Converts CH0 only

    // ADC clock derived from peripheral buss clock
    // Tadc = 10 * Tcy = 10 * 12.5 ns = 125 ns > 117.6 ns required
    // Tadc = (    9       +1)*Tcy
    // Tadc = (AD1CON3<7:0>+1)*Tcy
    AD1CON3bits.ADCS = 9;

    AD1CHS0bits.CH0SA = 5;                   // Select channel AN5 on PGED1/AN5/C1IN1-/RP35/RB3

    // TO DO: set ANSELx to assign ADC pins

    IEC0bits.AD1IE = 0;                      // Disable ADC interrupts

    AD1CON1bits.ADON = 1;                    // Turn on ADC

// Sweep over all possible current trim values
    for ( iTrim = -31; iTrim < 32; iTrim++ )
    {
        CTMUICONbits.ITRIM = iTrim & 0x3F;   // Set current trim value
        CTMUCON1bits.CTMUEN = 1;             // Turn on CTMU
        // TO DO: Wait 1 ms for CTMU to warm up

        ADC_Sum = 0;
        for ( iAvg = 0; iAvg < Naverages; iAvg++ )
        {
            AD1CON1bits.SAMP = 1;            // Manual sampling start
            CTMUCON1bits.IDISSEN = 1;        // Ground charge pump
            // TO DO: Wait 500 us
            CTMUCON1bits.IDISSEN = 0;        // End drain of circuit
```

33

Charge Time Measurement Unit (CTMU)

Example 33-1: Current Calibration Routine (Continued)

```
            CTMUCON2bits.EDG1STAT = 1;    // Begin charging the circuit
            // TO DO: Charge Delay based on current range

            AD1CON1bits.SAMP = 0;         // Begin analog-to-digital conversion
            CTMUCON2bits.EDG1STAT = 0;    // Stop charging circuit
            while (!AD1CON1bits.DONE)     // Wait for ADC conversion
            {
                //Do Nothing
            }
            AD1CON1bits.DONE = 0;         // ADC conversion done, clear flag
            VmeasADC = ADC1BUF0;          // Get the value from the ADC
            ADC_Sum += VmeasADC;          // Update averaging sum

        }//end for ( iAvg = 0; iAvg < Naverages; iAvg++ )
        CTMUCON1bits.CTMUEN = 0;          // Turn off CTMU

        VavgADC = ADC_Sum >> (Log2Naverages-5); // Full scale = 2^10<<5 = 32768

        // TO DO: Format text string of iTrim and VavgADC, transmit string

    }//end for ( iTrim = -31; iTrim < 32; iTrim++ )

    // CTMU already turned off
    AD1CON1bits.ADON = 1;                 // Turn off ADC
    U2MODEbits.UARTEN = 0;                // Disable UAR

}//end main()
```


33.5.2 Capacitance Calibration

A small amount of capacitance from the internal ADC sample capacitor and stray capacitance from the circuit board traces and pads that affect the precision of capacitance measurements. A measurement of the stray capacitance can be taken to ensure the desired capacitance to be measured is removed. The measurement is then performed using the following steps:

1. Initialize the ADC and the CTMU.
2. Set EDG1STAT (= `1`).
3. Wait for a fixed delay of time t.
4. Clear EDG1STAT.
5. Perform an analog-to-digital conversion.
6. Calculate the stray and analog-to-digital sample capacitances using Equation 33-5.

Equation 33-5:

$$C_{OFFSET} = C_{STRAY} + C_{AD} = \frac{(I \cdot t)}{V}$$

where I is known from the current source measurement step, t is a fixed delay and V is measured by performing an analog-to-digital conversion.

This measured value is then stored and used for calculations of time measurement, or subtracted for capacitance measurement. For calibration, it is expected that the capacitance of $C_{STRAY} + C_{AD}$ is approximately known. C_{AD} is approximately 4 pF.

An iterative process must be used to adjust the time t, that the circuit is charged to obtain a reasonable voltage reading from the ADC. The value of **t** may be determined by setting C_{OFFSET} to a theoretical value, then solving for t. For example, if C_{STRAY} is theoretically calculated to be 11 pF, and V is expected to be 70% of V_{DD}, or 2.31V, t would be equal to Equation 33-6 or 63 µs.

Equation 33-6: :

$$(4\ pF + 11\ pF) \bullet \frac{2.31V}{0.55\mu A}$$

A charge delay of 63 µs represents 2520 instructions when the system is operating at 40 MHz (2520 = 40 MHz * 63 µs). If the CTMU charge pump is set to 100 x base current (55 µA), the charge time is cut by a factor of 100, or 25.2 instructions. This can easily be implemented by 25 `NOP` instructions in the code.

Example 33-2 shows one typical method for measuring capacitance once the CTMU's charge pump has been calibrated.

Note:	A complete MPLAB project based on this calibration routine is available on the Microchip web site at: http://www.microchip.com/CodeExamplesByFunc.aspx. From the landing page, scroll down and select **Touch Sense (mTouch)** as the application.

Example 33-2: Capacitance Calibration Routine

```
typedef enum _PLIB_CTMU_CURRENT_RANGE
{
    PLIB_CTMU_CurrentRange_Base      = 0x1,
    PLIB_CTMU_CurrentRange_10xBase   = 0x2,
    PLIB_CTMU_CurrentRange_100xBase  = 0x3,
    PLIB_CTMU_CurrentRange_1000xBase = 0x0, // PIC32
    PLIB_CTMU_CurrentRange_Disabled  = 0x0  // Others
} PLIB_CTMU_CURRENT_RANGE;

// TO DO: Device Setup

int main(void)
{
    char                ButtonMeasString[133]; // UART output string
    unsigned long int   ADC_Sum;               // For averaging multiple ADC measurements
    unsigned short int iAvg,                   // Averaging index
                        Naverages = 1024,      // Number of averages < 2^22 (22=32-10 bits of ADC)
                        Log2Naverages = 10;    // Right shift equal to 1/Naverages
    short int           iTrim;                 // Current trim index
    unsigned short int VmeasADC, VavgADC;      // Measured Voltages, 65536 = Full Scale

    // TO DO: Clock switch to incorporate PLL
    // TO DO: Configure UART2 for diagnostic dumps to ProfiLab
    // TO DO: Setup Pins

    // CTMU Setup
    CTMUCON1bits.TGEN = 1;             // Enable direct output to PGED1/AN5/C1IN1-/RP35/RB3 pin
    CTMUICONbits.IRNG = PLIB_CTMU_CurrentRange_100xBase;     // Current Range
//  Turn on CTMU after setting current trim below

    // Comparator Setup
    PMD3bits.CMPMD = 1;                  // Disable comparator

    // ADC Setup
    AD1CON1bits.AD12B = 0;               // 10-bit mode
    AD1CON2 = 0x0;                       // VR+ = AVDD, V- = AVSS, Don't scan, Converts CH0 only

    // ADC clock derived from peripheral buss clock
    // Tadc = 10 * Tcy = 10 * 12.5 ns = 125 ns > 117.6 ns required
    // Tadc = (    9       +1)*Tcy
    // Tadc = (AD1CON3<7:0>+1)*Tcy
    AD1CON3bits.ADCS = 9;

    AD1CHS0bits.CH0SA = 5;               // Select channel AN5 on PGED1/AN5/C1IN1-/RP35/RB3

    // TO DO: set ANSELx to assign ADC pins

    IEC0bits.AD1IE = 0;                  // Disable ADC interrupts

    AD1CON1bits.ADON = 1;                // Turn on ADC

    // Sweep over all possible current trim values
    for ( iTrim = -31; iTrim < 32; iTrim++ )
    {
        CTMUICONbits.ITRIM = iTrim & 0x3F; // Set current trim value
        CTMUCON1bits.CTMUEN = 1;           // Turn on CTMU
        // TO DO: Wait 1 msec for CTMU to warm up

        ADC_Sum = 0;
        for ( iAvg = 0; iAvg < Naverages; iAvg++ )
        {
            AD1CON1bits.SAMP = 1;     // Manual sampling start
            CTMUCON1bits.IDISSEN = 1; // Ground charge pump
            Delay10us(50);
            CTMUCON1bits.IDISSEN = 0; // End drain of circuit
```


Example 33-2: Capacitance Calibration Routine (Continued

```
            CTMUCON2bits.EDG1STAT = 1;       // Begin charging the circuit
            // Use Equation 33-6 to solve for charge time with current = 55 uA
            // Charge time = 0.63 us, or 25.2 NOPs at 40 MHz system clock
            // TO DO: Wait 25 NOPs
            AD1CON1bits.SAMP = 0;            // Begin analog-to-digital conversion
            CTMUCON2bits.EDG1STAT = 0;       // Stop charging circuit
            while (!AD1CON1bits.DONE)        // Wait for ADC conversion
            {
                //Do Nothing
            }
            AD1CON1bits.DONE = 0;            // ADC conversion done, clear flag
            VmeasADC = ADC1BUF0;             // Get the value from the ADC
            ADC_Sum += VmeasADC;             // Update averaging sum

        }//end for ( iAvg = 0; iAvg < Naverages; iAvg++ )
        CTMUCON1bits.CTMUEN = 0;             // Turn off CTMU

        VavgADC = ADC_Sum >> (Log2Naverages-5); // Full scale = 2^10<<5 = 32768

        // TO DO: Format text string with iTrim and VavgADC, transmit string

        if ( VavgADC > 23000 )               // Stop, ADC voltage too high
        {
            break;
        }

    }//end for ( iTrim = -31; iTrim < 32; iTrim++ )

    // CTMU already turned off
    AD1CON1bits.ADON = 1;                    // Turn off ADC
    U2MODEbits.UARTEN = 0;                   // Disable UAR

}//end main()
```

33.6 MEASURING CAPACITANCE WITH THE CTMU

The two methods of measuring capacitance with the CTMU are:

- Absolute Capacitance Measurement: The actual capacitance value is required
- Relative Charge Measurement: The actual capacitance value is not required instead an indication of change in capacitance is required.

33.6.1 Absolute Capacitance Measurement

For absolute capacitance measurements, both the current and capacitance calibration steps found in **33.5 "Calibrating the CTMU Module"** should be followed. Capacitance measurements are then performed using the following steps:

1. Initialize the ADC.
2. Initialize the CTMU.
3. Set EDG1STAT.
4. Wait for a fixed delay, *T*.
5. Clear EDG1STAT.
6. Perform an analog-to-digital conversion.
7. Calculate the total capacitance, $C_{TOTAL} = (I * T)/V$, where I is known from the current source measurement step (**33.5.1 "Current Source Calibration"**), T is a fixed delay and V is measured by performing an analog-to-digital conversion
8. Subtract the stray and analog-to-digital capacitance (C_{OFFSET} from **33.5.2 "Capacitance Calibration"**) from C_{TOTAL} to determine the measured capacitance.

33.6.2 Relative Charge Measurement

An application may not require precise capacitance measurements. For example, when detecting a valid press of a capacitance-based switch, detecting a relative change of capacitance is of interest. In this application, when the switch is open (or not touched), the total capacitance is the capacitance of the combination of the board traces, the ADC, etc. A larger voltage will be measured by the ADC. When the switch is closed (or is touched), the total capacitance is larger due to the addition of the capacitance of the human body to the above listed capacitances, and a smaller voltage will be measured by the ADC.

Detecting capacitance changes can be done with the CTMU using these steps:

1. Initialize the ADC and the CTMU.
2. Set EDG1STAT.
3. Wait for a fixed delay.
4. Clear EDG1STAT.
5. Perform an analog-to-digital conversion.

The voltage measured by performing the analog-to-digital conversion is an indication of the relative capacitance. In this case, no calibration of the current source or circuit capacitance measurement is required.

A sample software routine for a capacitive touch switch is shown in Example 33-3. In this example, the prior calibration of the 8-Key Direct Sensor Daughter Board can be used to equalize the measured voltages by adjusting charge time for variations in each button circuit's capacitance. Alternatively, a fixed charge time can be used if the software supports separate trigger levels for each button. For simplicity, the routine only checks the second button (9) on the 8-Key Direct Sensor Daughter Board.

Note:	A more advanced version that measures all eight buttons is available on the Microchip web site at: http://www.microchip.com/CodeExamplesByFunc.aspx. From the landing page, scroll down and select **Touch Sense (mTouch)** as the application.

Example 33-3: Routine for Capacitive Touch Switch

```
#define NUM_DIRECT_KEYS  8
static unsigned short int ButtonADCChannels[NUM_DIRECT_KEYS] = {1,2,3,4,5,6,7,8};

// TO DO: Device Setup

int main(void)
{
    char                ButtonMeasString[133]; // UART output string
    unsigned long int   ADC_Sum;               // For averaging multiple ADC measurements
    unsigned short int iAvg,                   // Averaging index
                        Naverages = 32,        // Number of averages < 2^22 (22=32-10 bits of ADC)
                        Log2Naverages = 5;     // Right shift equal to 1/Naverages
    short int           iTrim,                 // Current trim index
                        iButton,               // Button index
                        iChan,                 // ADC channel index
                        CurrentButtonStatus;   // Bit field of buttons that are pressed
    unsigned short int VmeasADC, VavgADC;      // Measured Voltages, 65536 = Full Scale
    unsigned short int ButtonVmeasADC[NUM_DIRECT_KEYS]; // Report out all voltages at once

    // TO DO: Clock switch to incorporate PLL
    // TO DO: Configure UART2 for diagnostic dumps to ProfiLab
    // TO DO: Setup Pins

    // CTMU Setup
    CTMUICONbits.IRNG = 0x3;                   // Current Range
    CTMUICONbits.ITRIM  = -25 & 0x3F;          // Set current trim value
    CTMUCON1bits.CTMUEN = 1;                   // Turn on CTMU
    DelayMs(1);                                // Wait 1 ms

    // Comparator Setup
    PMD3bits.CMPMD = 1;                        // Disable comparator

    // ADC Setup
    AD1CON1bits.AD12B = 0;                     // 10-bit mode
    AD1CON2 = 0x0;                     // VR+ = AVDD, V- = AVSS, Don't scan, Converts CH0 only

    // ADC clock derived from peripheral buss clock
    // Tadc = 10 * Tcy = 10 * 12.5 ns = 125 ns > 117.6 ns required
    // Tadc = (    9       +1)*Tcy
    // Tadc = (AD1CON3<7:0>+1)*Tcy
    AD1CON3bits.ADCS = 9;

    // TO DO: set ANSELx to assign ADC pins

    IEC0bits.AD1IE = 0;                        // Disable ADC interrupts

    AD1CON1bits.ADON = 1;                      // Turn on ADC

    while (1)
    {
        CurrentButtonStatus = 0;
        for ( iButton = 0; iButton < NUM_DIRECT_KEYS; iButton++ )
        {
            iChan = ButtonADCChannels[iButton];
            AD1CHS0bits.CH0SA = iChan;

            ADC_Sum = 0;
            for ( iAvg = 0; iAvg < Naverages; iAvg++ )
            {
                AD1CON1bits.SAMP = 1;          // Manual sampling start
                CTMUCON1bits.IDISSEN = 1;      // Ground charge pump
                // TO DO: Wait 500 us for CTMU to warm up
                CTMUCON1bits.IDISSEN = 0;      // End drain of circuit

                CTMUCON2bits.EDG1STAT = 1;     // Begin charging the circuit
                // TO DO: Wait for charge accumulation
                AD1CON1bits.SAMP = 0;          // Begin analog-to-digital conversion
                CTMUCON2bits.EDG1STAT = 0;     // Stop charging circuit
```

33

Charge Time Measurement Unit (CTMU)

Example 33-3: Routine for Capacitive Touch Switch (Continued)

```
                AD1CON1bits.SAMP = 0;      // Begin analog-to-digital conversion
                CTMUCON2bits.EDG1STAT = 0; // Stop charging circuit
                while (!AD1CON1bits.DONE)  // Wait for ADC conversion
                {
                    //Do Nothing
                }
                AD1CON1bits.DONE = 0; // ADC conversion done, clear flag
                VmeasADC = ADC1BUF0;  // Get the value from the ADC
                ADC_Sum += VmeasADC;  // Update averaging sum

            }//end for ( iAvg = 0; iAvg < Naverages; iAvg++ )

            if ( Log2Naverages-5 > 0 )
            {
                VavgADC = ADC_Sum >> (Log2Naverages-5); // Full scale = 2^10<<5 = 32768
            }
            else
            {
                VavgADC = ADC_Sum << (5-Log2Naverages); // Full scale = 2^10<<5 = 32768
            }

            if ( VavgADC < 12000 )                      // Button is being pressed
            {
                CurrentButtonStatus += 1<<iButton;
            }
            ButtonVmeasADC[iButton] = VavgADC;

        }//end for ( iButton = 0; iButton < NUM_DIRECT_KEYS; iButton++ )

    }//end while(1)

}//end main()
```


33.7 MEASURING TIME WITH THE CTMU MODULE

Time can be precisely measured after the ratio (C/I) is measured from the current and capacitance calibration step by following these steps:

1. Initialize the ADC and the CTMU.
2. Set EDG1STAT.
3. Set EDG2STAT.
4. Perform an analog-to-digital conversion.
5. Calculate the time between edges as $T = (C/I) * V$, where I is calculated in the current calibration step (**33.5.1 "Current Source Calibration"**), C is calculated in the capacitance calibration step (**33.5.2 "Capacitance Calibration"**) and V is measured by performing the analog-to-digital conversion.

It is assumed that the time measured is small enough that the capacitance C_{OFFSET} provides a valid voltage to the ADC. For the smallest time measurement, set the ADC Channel Select register (AD1CHS0) to the ADC input signal labeled Open. This minimizes any stray capacitance by keeping the total circuit capacitance close to the ADC (4-5 pF). To measure longer time intervals, an external capacitor may be connected to an ADC channel, and this channel is selected when making a time measurement.

33.8 CREATING A DELAY WITH THE CTMU MODULE

A unique feature of the CTMU module is to generate system clock independent output pulses based on an external capacitor value. This is accomplished using the internal comparator voltage reference module, the Comparator module input pin, and an external capacitor. The pulse is output onto the CTPLS pin. To enable this mode, you need to set the TGEN bit (CTMUCON1<12>).

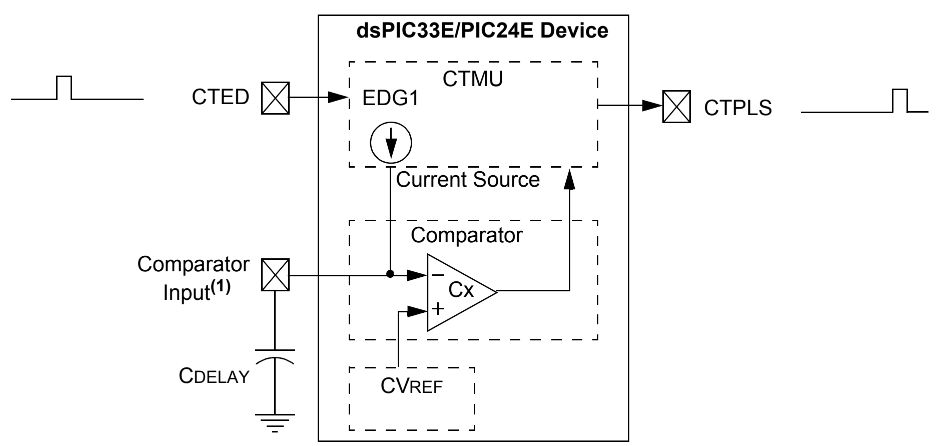
An example circuit is illustrated in Figure 33-3. C_{PULSE} is chosen by the user to determine the output pulse width on CTPLS. The pulse width is calculated by $T = (C_{PULSE}/I) * V$, where I is known from the current source measurement step (**33.5.1 "Current Source Calibration"**) and V is the internal reference voltage (CVREF).

An example use of this feature is for interfacing with variable capacitive-based sensors like a humidity sensor. As the humidity varies, the pulse-width output on CTPLS will vary. The CTPLS output pin can be connected to an input capture pin and the varying pulse width is measured to determine the humidity in the application.

Follow these steps to use this feature:

1. Initialize the Comparator module.
2. Initialize the comparator voltage reference.
3. Initialize the CTMU and enable time delay generation by setting the TGEN bit.
4. Set EDG1STAT.
5. When C_{PULSE} charges to the value of the voltage reference trip point, an output pulse is generated on CTPLS.

Figure 33-3: Typical Connections and Internal Configuration for Pulse Delay Generation



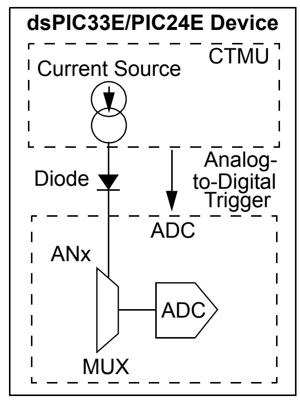
Note 1: Refer to the **"Comparator"** and **"CTMU"** chapters in the specific device data sheet to determine which comparator and comparator input pin is used by the CTMU module.

33.9 MEASURING ON-CHIP TEMPERATURE WITH THE CTMU

The CTMU module can be used to measure the internal temperature of the device through an internal diode that is available for such purposes. When EDGE 1 is not equal to EDGE 2 and TGEN = 0, the current is steered into the temperature sensing diode. The voltage across the diode is available as an input to the ADC module. Refer to the specific device data sheet for information on the analog channel on which the diode is connected.

Figure 33-4 illustrates how this module can be used for temperature measurement. As the temperature rises, the voltage across the diode will drop by about 300 mV over a 150ºC range. Selecting a higher current drive strength will raise the voltage value by a few 100 mV.

Figure 33-4: CTMU Temperature Measurement Circuit



At 25ºC, the forward voltage of the temperature diode is 0.83V. The rate of change between the forward voltage of the diode and its temperature is 1.87 mV per degree Celsius. The formula shown in Equation 33-7 can be used to calculate the forward voltage.

Equation 33-7: Voltage versus Temperature

$$Vf\,(\text{in mV}) = 783.24mV + 1.87mV \cdot T$$

Where:

Vf = Forward voltage of temperature diode

T = Temperature in degrees Celsius

33.10 OPERATION DURING SLEEP OR IDLE MODE

33.10.1 Sleep Mode

When the device enters Sleep mode, the CTMU module current source is always disabled. If the CTMU is performing an operation that depends on the current source when Sleep mode is invoked, the operation may not terminate correctly. Capacitance and time measurements may return erroneous values.

33.10.2 Idle Mode

The behavior of the CTMU in Idle mode is determined by the CTMUSIDL bit (CTMUCON1<13>). If CTMUSIDL is cleared, the module will continue to operate in Idle mode. If CTMUSIDL is set, the CTMU module's current source is disabled when the device enters Idle mode. If the CTMU module is performing an operation when Idle mode is invoked, the results will be similar to those with Sleep mode.

33.11 EFFECTS OF A RESET ON CTMU

Upon Reset, all registers of the CTMU are cleared. This leaves the CTMU module disabled, its current source is turned off and all configuration options return to their default settings. The module needs to be reinitialized following any Reset.

If the CTMU is in the process of taking a measurement at the time of Reset, the measurement will be lost. A partial charge may exist on the circuit that was being measured, and should be properly discharged before the CTMU makes subsequent attempts to make a measurement. The circuit is discharged by setting and then clearing the IDISSEN bit (CTMUCON1<9>) while the ADC is connected to the appropriate channel.

33.12 REGISTER MAPS

A summary of the registers associated with the dsPIC33E/PIC24E CTMU is given in Table 33-1.

Table 33-1: CTMU Register Map

File Name	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
CTMUCON1(1)	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG	—	—	—	—	—	—	—	—	0000
CTMUCON2(1)	EDG1MOD	EDG1POL	EDG1SEL<3:0>				EDG2STAT	EDG1STAT	EDG2MOD	EDG2POL	EDG2SEL<3:0>				—	—	0000
CTMUICON	ITRIM<5:0>						IRNG<1:0>		—	—	—	—	—	—	—	—	0000

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

Note 1: Refer to the **"CTMU"** chapter in the specific device data sheet to determine whether this register is available on your particular device.

33.13 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 dsPIC33E/PIC24E device family, but the concepts are pertinent and could be used with modification and possible limitations. The current application notes related to the Charge Time Measurement Unit (CTMU) module are:

Title	Application Note #
Microchip CTMU for Capacitive Touch Applications	AN1250
Water-Resistant Capacitive Sensing	AN1286
mTouch™ Conducted Noise Immunity Techniques for the CTMU	AN1317
Techniques for Robust Touch Sensing Design	AN1334

Note:	Visit the Microchip web site (www.microchip.com) for additional application notes and code examples for the dsPIC33E/PIC24E family of devices.

33.14 REVISION HISTORY

Revision A (May 2011)

This is the initial released version of this document.

Revision B (February 2012)

This revision includes the following updates:

- Added Note 1 and Note 2 to **33.3.2 "Current Source"**
- Updated the last sentence of the first paragraph in **33.3.4 "Edge Status"**
- Updated **33.5.2 "Capacitance Calibration"**
- Updated **33.6.2 "Relative Charge Measurement"**
- Added Note 1 to the CTMU Current Source Calibration Circuit (see Figure 33-2)
- Removed Setup for CTMU Calibration Routines (formerly Example 37-1)
- Updated the code in the Current Calibration Routine (see Example 33-1)
- Updated the code in the Capacitance Calibration Routine (see Example 33-2)
- Updated the code in the Routine for Capacitive Touch Switch Routine (see Example 33-3)
- Minor updates to text and formatting were incorporated throughout the document

NOTES:

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2009

Trademarks

Printed on recycled paper.

ISBN: 978-1-62076-037-6

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Hangzhou
Tel: 86-571-2819-3187
Fax: 86-571-2819-3189

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Osaka
Tel: 81-66-152-7160
Fax: 81-66-152-9310

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-330-9305

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

11/29/11