mTouch Cap Library Help

Table of Contents

1 Introduction

Introduction

The Capacitive mTouch^{TM} Software Library provides the API's to develop capacitive touch applications using the Charge Time Measurement Unit (CTMU) and Capacitive Voltage Divider (CVD) technique on PIC18F, PIC24F and PIC32MX Microcontrollers (MCUs).

The software stack is developed using ‘C’ language and can be compiled by Microchip C18, C30 and C32 compilers for PIC18F, PIC24F and PIC32MX Microcontrollers.

Users of the mTouch^{TM} Software Library can select the PIC microcontroller used for the application and configure the CTMU or CVD Demos as required for the Application. The API's helps the user to integrate the mTouch Capacitive Library with the end application. This library is also designed to operate with other libraries developed by Microchip.

The CTMU has a constant current source that can be used for relative capacitance measurement, absolute capacitance measurement and accurate time measurement. This library will use the relative capacitance measurement for capacitive touch sensing application. Refer to the CTMU Family Reference Manual (DS39724) for more details of CTMU.

The CVD technique resides in successive charging and discharging cycles of ADC sample and holds capacitor and the external capacity of the sensor, while measuring the voltage left on the sample and hold capacitor after each cycle. This library contains the implementation of the CVD technique. Refer to the Capacitive Touch Using Only ADC (CVD) – AN1298 for more details.

The Capacitive mTouch^{TM} Software library is also implemented for PIC16F CVD Framework.

The Help file for PIC16F is available in the following location:

....\Microchip\Help\mTouch CVD Framework Documentation.

Hardware Setup for testing

The PIC18F, PIC24F and PIC32 enhanced Capacitive Touch Eval kit (DM183026-2) is used for testing the Capacitive mTouch^{TM} Software Library and its functionality.

2 Software License Agreement

MICROCHIP IS WILLING TO LICENSE THE ACCOMPANYING SOFTWARE AND DOCUMENTATION TO YOU ONLY ON THE CONDITION THAT YOU ACCEPT ALL OF THE FOLLOWING TERMS. TO ACCEPT THE TERMS OF THIS LICENSE, CLICK "I ACCEPT" AND PROCEED WITH THE DOWNLOAD OR INSTALL. IF YOU DO NOT ACCEPT THESE LICENSE TERMS, CLICK "I DO NOT ACCEPT," AND DO NOT DOWNLOAD OR INSTALL THIS SOFTWARE. BY DOWNLOADING AND INSTALLING THE SOFTWARE, LICENSEE AGREES TO BE BOUND BY THE TERMS OF THIS AGREEMENT.

NON-EXCLUSIVE SOFTWARE LICENSE AGREEMENT FOR ACCOMPANYING MICROCHIP SOFTWARE AND DOCUMENTATION

This Nonexclusive Software License Agreement ("Agreement") is a contract between you, your heirs, successors and assigns ("Licensee") and Microchip Technology Incorporated, a Delaware corporation, with a principal place of business at 2355 W. Chandler Blvd., Chandler, AZ 85224-6199, and its subsidiary, Microchip Technology (Barbados) II Incorporated (collectively, "Microchip") for the accompanying Microchip software including, but not limited to, Graphics Library Software, IrDA Stack Software, MCHPFSUSB Stack Software, Memory Disk Drive File System Software, mTouch(TM) Capacitive Library Software, Smart Card Library Software, TCP/IP Stack Software, MiWi(TM) DE Software, and/or any PC programs and any updates thereto (collectively, the "Software"), and accompanying documentation, including images and any other graphic resources provided by Microchip ("Documentation").

The Software and Documentation are licensed under this Agreement and not sold. U.S. copyright laws, international copyright treaties, and other intellectual property laws and treaties protect the Software and Documentation. Microchip reserves all rights not expressly granted to Licensee in this Agreement.

(1) Definitions. As used in this Agreement, the following capitalized terms will have the meanings defined below:

a. "Microchip Products" means Microchip microcontrollers and Microchip digital signal controllers.

b. "Licensee Products" means Licensee products that use or incorporate Microchip Products.

c. "Object Code" means the Software computer programming code that is in binary form (including related documentation, if any), and error corrections, improvements, modifications, and updates.

d. "Source Code" means the Software computer programming code that may be printed out or displayed in human readable form (including related programmer comments and documentation, if any), and error corrections, improvements, modifications, and updates.

e. "Third Party" means Licensee’s agents, representatives, consultants, clients, customers, or contract manufacturers.

f. "Third Party Products" means Third Party products that use or incorporate Microchip Products.

(2) Software License Grant. Microchip grants strictly to Licensee a non-exclusive, non-transferable, worldwide license to:

a. use the Software in connection with Licensee Products and/or Third Party Products;

b. if Source Code is provided, modify the Software, provided that no Open Source Components (defined in Section 5 below) are incorporated into such Software in such a way that would affect Microchip’s right to distribute the Software with the limitations set forth herein and provided that Licensee clearly notifies Third Parties regarding such modifications;

c. distribute the Software to Third Parties for use in Third Party Products, so long as such Third Party agrees to be bound by this Agreement (in writing or by "click to accept") and this Agreement accompanies such distribution;

d. sublicense to a Third Party to use the Software, so long as such Third Party agrees to be bound by this Agreement (in writing or by "click to accept");

e. with respect to the TCP/IP Stack Software, Licensee may port the ENC28J60.c, ENC28J60.h, ENCX24J600.c, and ENCX24J600.h driver source files to a non-Microchip Product used in conjunction with a Microchip ethernet controller;

f. with respect to the MiWi (TM) DE Software, Licensee may only exercise its rights when the Software is embedded on a Microchip Product and used with a Microchip radio frequency transceiver or UBEC UZ2400 radio frequency transceiver which are integrated into Licensee Products or Third Party Products.

For purposes of clarity, Licensee may NOT embed the Software on a non-Microchip Product, except as described in this Section.

(3) Documentation License Grant. Microchip grants strictly to Licensee a non-exclusive, non-transferable, worldwide license to use the Documentation in support of Licensee's authorized use of the Software

(4) Third Party Requirements. Licensee acknowledges that it is Licensee’s responsibility to comply with any third party license terms or requirements applicable to the use of such third party software, specifications, systems, or tools. Microchip is not responsible and will not be held responsible in any manner for Licensee’s failure to comply with such applicable terms or requirements.

(5) Open Source Components. Notwithstanding the license grant in Section 1 above, Licensee further acknowledges that certain components of the Software may be covered by so-called "open source" software licenses ("Open Source Components"). Open Source Components means any software licenses approved as open source licenses by the Open Source Initiative or any substantially similar licenses, including without limitation any license that, as a condition of distribution of the software licensed under such license, requires that the distributor make the software available in source code format. To the extent required by the licenses covering Open Source Components, the terms of such license will apply in lieu of the terms of this Agreement, and Microchip hereby represents and warrants that the licenses granted to such Open Source Components will be no less broad than the license granted in Section 1. To the extent the terms of the licenses applicable to Open Source Components prohibit any of the restrictions in this Agreement with respect to such Open Source Components, such restrictions will not apply to such Open Source Component.

(6) Licensee Obligations.

a. Licensee will ensure Third Party compliance with the terms of this Agreement, and will be liable for any breach of this Agreement committed by such Third Party.

b. Licensee will not: (i) engage in unauthorized use, modification, disclosure or distribution of Software or Documentation, or its derivatives; (ii) use all or any portion of the Software, Documentation, or its derivatives except in conjunction with Microchip Products or Third Party Products; or (iii) reverse engineer (by disassembly, decompilation or otherwise) Software or any portion thereof.

c. Licensee may not remove or alter any Microchip copyright or other proprietary rights notice posted in any portion of the Software or Documentation.

d. Licensee will defend, indemnify and hold Microchip and its subsidiaries harmless from and against any and all claims, costs, damages, expenses (including reasonable attorney's fees), liabilities, and losses, including without limitation product liability claims, directly or indirectly arising from or related to: (i) the use, modification, disclosure or distribution of the Software, Documentation, or any intellectual property rights related thereto; (ii) the use, sale and distribution of Licensee Products or Third Party Products; and (iii) breach of this Agreement. THIS SECTION 3(d) STATES THE SOLE AND EXCLUSIVE LIABILITY OF THE PARTIES FOR INTELLECTUAL PROPERTY INFRINGEMENT.

(7) Confidentiality. Licensee agrees that the Software (including but not limited to the Source Code, Object Code and library files) and its derivatives, Documentation and underlying inventions, algorithms, know-how and ideas relating to the Software and the Documentation are proprietary information belonging to Microchip and its licensors ("Proprietary Information"). Except as expressly and unambiguously allowed herein, Licensee will hold in confidence and not use or disclose any Proprietary Information and will similarly bind its employees and Third Party(ies) in writing. Proprietary Information will not include information that: (i) is in or enters the public domain without breach of this Agreement and through no fault of the receiving party; (ii) the receiving party was legally in possession of prior to receiving it; (iii) the receiving party can demonstrate was developed by the receiving party independently and without use of or reference to the disclosing party's Proprietary Information; or (iv) the receiving party receives from a third party without restriction on disclosure. If Licensee is required to disclose Proprietary Information by law, court order, or government agency, License will give Microchip prompt

notice of such requirement in order to allow Microchip to object or limit such disclosure. Licensee agrees that the provisions of this Agreement regarding unauthorized use and nondisclosure of the Software, Documentation and related Proprietary Rights are necessary to protect the legitimate business interests of Microchip and its licensors and that monetary damage alone cannot adequately compensate Microchip or its licensors if such provisions are violated. Licensee, therefore, agrees that if Microchip alleges that Licensee or Third Party has breached or violated such provision then Microchip will have the right to petition for injunctive relief, without the requirement for the posting of a bond, in addition to all other remedies at law or in equity.

(8) Ownership of Proprietary Rights. Microchip and its licensors retain all right, title and interest in and to the Software and Documentation ("Proprietary Rights") including, but not limited to all patent, copyright, trade secret and other intellectual property rights in the Software, Documentation, and underlying technology and all copies and derivative works thereof (by whomever produced). Further, copies and derivative works will be considered works made for hire with ownership vesting in Microchip on creation. To the extent such modifications and derivatives do not qualify as a "work for hire," Licensee hereby irrevocably transfers, assigns and conveys the exclusive copyright thereof to Microchip, free and clear of any and all liens, claims or other encumbrances, to the fullest extent permitted by law. Licensee and Third Party use of such modifications and derivatives is limited to the license rights described in Sections this Agreement.

(9) Termination of Agreement. Without prejudice to any other rights, this Agreement terminates immediately, without notice by Microchip, upon a failure by Licensee or Third Party to comply with any provision of this Agreement. Upon termination, Licensee and Third Party will immediately stop using the Software, Documentation, and derivatives thereof, and immediately destroy all such copies.

(10) Warranty Disclaimers. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY, TITLE, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE. MICROCHIP AND ITS LICENSORS ASSUME NO RESPONSIBILITY FOR THE ACCURACY, RELIABILITY OR APPLICATION OF THE SOFTWARE OR DOCUMENTATION. MICROCHIP AND ITS LICENSORS DO NOT WARRANT THAT THE SOFTWARE WILL MEET REQUIREMENTS OF LICENSEE OR THIRD PARTY, BE UNINTERRUPTED OR ERROR-FREE. MICROCHIP AND ITS LICENSORS HAVE NO OBLIGATION TO CORRECT ANY DEFECTS IN THE SOFTWARE. LICENSEE AND THIRD PARTY ASSUME THE ENTIRE RISK ARISING OUT OF USE OR PERFORMANCE OF THE SOFTWARE AND DOCUMENTATION PROVIDED UNDER THIS AGREEMENT.

(11) Limited Liability. IN NO EVENT WILL MICROCHIP OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER CONTRACT, NEGLIGENCE, STRICT LIABILITY, CONTRIBUTION, BREACH OF WARRANTY, OR OTHER LEGAL OR EQUITABLE THEORY FOR ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES INCLUDING BUT NOT LIMITED TO INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS. The aggregate and cumulative liability of Microchip and its licensors for damages hereunder will in no event exceed $1000 or the amount Licensee paid Microchip for the Software and Documentation, whichever is greater. Licensee acknowledges that the foregoing limitations are reasonable and an essential part of this Agreement.

LICENSEE ACKNOWLEDGES AND AGREES THAT IT IS SOLELY RESPONSIBLE FOR THE LICENSEE PRODUCTS AND THIRD PARTY PRODUCTS, INCLUDING DETERMINING WHETHER SUCH PRODUCTS INFRINGE A PATENT, COPYRIGHT OR OTHER PROPRIETARY RIGHT OF ANY THIRD PARTY. LICENSEE AGREES THAT MICROCHIP HAS NO OBLIGATION TO INDEMNIFY OR DEFEND LICENSEE IN THE EVENT THAT A THIRD PARTY MAKES A CLAIM REGARDING LICENSEE PRODUCTS OR THIRD PARTY PRODUCTS.

(12) General. THIS AGREEMENT WILL BE GOVERNED BY AND CONSTRUED UNDER THE LAWS OF THE STATE OF ARIZONA AND THE UNITED STATES WITHOUT REGARD TO CONFLICTS OF LAWS PROVISIONS. Licensee agrees that any disputes arising out of or related to this Agreement, Software or Documentation will be brought in the courts of the State of Arizona. The parties agree to waive their rights to a jury trial in actions relating to this Agreement. If either the Microchip or Licensee employs attorneys to enforce any rights arising out of or relating to this Agreement, the prevailing party will be entitled to recover its reasonable attorneys' fees, costs and other expenses. This Agreement will constitute the entire agreement between the parties with respect to the subject matter hereof. It will not be modified except by a written agreement signed by an authorized representative of the Microchip. Microchip’s authorized representatives will have the right to reasonably inspect Licensee's premises and to audit Licensee's records and inventory of Licensee Products in order to ensure Licensee's adherence to the terms of this Agreement. If any provision of this Agreement will be held by a court of

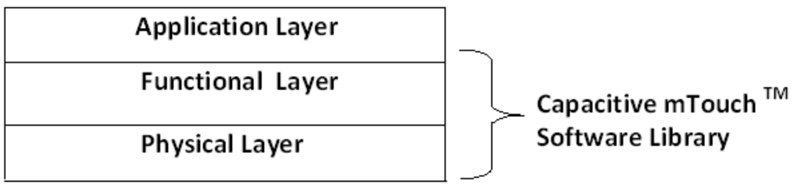
3 Software Stack - Layers

The Capacitive mTouch software library is implemented in three layers:

1) Physical Layer: This configures the hardware, based on the PIC device chosen.

2) Functional Layer: This defines all the APIs on the sensing algorithm.

3) Application Layer: Demonstrates how to use the APIs



The layered approach provides the advantage of configuring the stack for the application. For example, the physical layer can be configured based on the sensing technology and PIC MCU used.

Details about each layer and files included for each of them are mentioned in the subsequent sections.

3.1 Physical Layer

This is the layer where CTMU, ADC and Timers are configured as per the application needs. The input for configuration comes from the upper layers.

Following table shows the files of physical layer and their scope/description:

File Name	**Description**
mTouchCap_PIC18_CTMU_Physical.c/h **mTouchCap_PIC24_CTMU_Physical.c/h** **mTouchCap_PIC32MX_CVD_Physical.c/h**	These files have control and status handling routines for CTMU and CVD methods. This forms the Physical layer of the software stack. Functions in this layer take commands from the upper layer to configure the hardware registers(SFRs) controlling the CTMU and CVD methods. Example for CTMU routines of Physical layer includes InitCTMU, ChargeTimeInit, ReadCTMU, Current_trim_config, StablizeChannelData etc. Example for CVD routines of Physical layer includes InitCVD, ReadCVD, StablizeChannelData etc.
mTouchCap_Adc.c/h	This file comprises all the ADC configurations. The CTMU and CVD work is in conjunction with the ADC module to provide time or charge measurement. Example for CTMU/CVD routines of Physical layer includes OpenADC, SetChannelADC, ReadADCBuff, ADC_InitializeInterrupt, CloseADC etc.
mTouchCap_Timers.c/h	This file includes routines to handle Timer peripheral used by software stack. Example for CTMU/CVD routines of Physical layer includes TickInit, ISR routine, etc.

3.2 Functional Layer

The Functional layer of Capacitive mTouch software library implements all the APIs of the stack. It provides an interface to the Application layer and the physical layer. It also contains the functions used in the API's routines.

All the API functions are implemented in this layer.

Note: Users of the ‘library’ must take care of the pre-conditions mentioned for each API before invoking them.

Following table provides the recommended files and their scope/description:

File Name	Description
mTouchCap_CtmuAPI.c/h **mTouchCap_CvdAPI.c/h**	These files include all the API definitions/functions. As the name implies, these are the application programming interfaces to physical layer from application layer. CTMU routines of Functional layer include: mTouchCapAPI_SetUpCTMU_Default (see page 22), mTouchCapAPI_CTMU_SetupCurrentSource (see page 14), mTouchCapAPI_AutoAdjustChannel (see page 13) , mTouchCapAPI_CTMU_GetChannelReading (see page 14), mTouchCapAPI_ScanChannelIterative (see page 18), mTouchCapAPI_getChannelTouchStatus (see page 15), mTouchCapAPI_GetStatusMatrixButton (see page 17), mTouchCapAPI_GetStatusSlider2Ch (see page 17), mTouchCapAPI_GetStatusSlider4Ch (see page 18), mTouchCapAPI_SetUpChannelDirectKey (see page 19), mTouchCapAPI_SetUpChannelMatrixKey (see page 20), mTouchCapAPI_SetUpChannelSlider2Ch (see page 21), mTouchCapAPI_SetUpChannelSlider4Ch (see page 22), mTouchCapAPI_GetStatusDirectButton (see page 16) etc.

All the APIs such as Direct Key Sensing, Matrix Key sensing, Slider Sensing, Channel Setup for each channel etc are implemented in this layer.

3.3 Application Layer

The Application layer demonstrates the use of API's and the under laying physical functions. It also comprises the configuration options for the application. Users can configure the library to suit the application that they are designing.

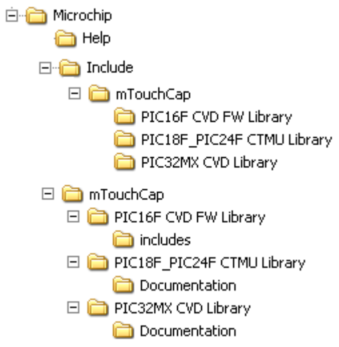
The following table provides the recommended files and their scope/description in Application layer:

File Name	**Description**
mTouchCap_PICXXF_DirectKey.c/h **mTouchCap_PICXXF_MatrixKey.c/h** **mTouchCap_PICXXF_2ChSlider.c/h** **mTouchCap_PICXXF_4ChSlider.c/h** **mTouchCap_PICXXF_Combo.c/h** **mTouchCap_PICXXF_GUI.c/h** **mTouchCap_PIC24F_DA210Graphics.c/h** **mTouchCap_PIC24F_Gesture.c/h** **mTouchCap_PIC24F_Proximity.c/h** **mTouchCap_PIC24F_LowPower.c/h** **mTouchCap_PIC24F_AN1317.c/h** **mTouchCap_PICXXXX_DirectKeyDemo.c/h** **mTouchCap_PICXXXX_MatrixKeyDemo.c/h** **mTouchCap_PICXXXX_2ChSliderDemo.c/h** **mTouchCap_PICXXXX_4ChSliderDemo.c/h** **Note:** **PICXXF - PIC18F/PIC24F** **PICXXXX- PIC32MX**	These files include the 'main' function of the application. It also demonstrates the 'API usage' (function calling with parameters) and preconditions (if any). These files call the APIs that are interfaced from 'Functional layer'. There will be a separate folder for each PIC18, PIC24 and PIC32 demo. The following demos are available: mTouchCap_DirectKey (see page 24): Demonstrates the functionalities and usage of the APIs related to applications with Direct Key. mTouchCap_MatrixKey (see page 24): Demonstrates the functionalities and usage of the APIs related to applications with Matrix Key. mTouchCap_2ChSlider (see page 25): Demonstrates the functionality and usage of the APIs related to applications with 2-channel sliders . mTouchCap_4ChSlider (see page 25): Demonstrates the functionalities and usage of the APIs related to applications with 4-channel sliders . mTouchCapCombo (see page 26): Demonstrates the functionalities and usage of APIs related to any combination of applications which use multiple types of Cap Touch Sensors (Ex: Sliders using 2 channels and Sliders using 4 channels). mTouchCap_GUI (see page 26): Demonstrates the usage of mTouch Diagnostic Tool in MPLAB for any combination of Cap Touch Sensor boards. mTouchCap_DA210Graphics (see page 27): Demonstrates the functionalities and usage of the APIs for a Graphics-applications on DA210 demo board. mTouchCap_Gesture (see page 28): Demonstrates the functionalities and usage of the APIs related to applications for Gesture detection. mTouchCap_Proximity (see page 29): Demonstrates the functionalities and usage of the APIs related to applications for Proximity detection. mTouchCap_AN1317 (see page 30): Demonstrate the conducted noise immune code using CTMU for capacitive touch application.
mTouchCap_Config.h	This file allows all types of application specific configurations of the Capacitive mTouch library. There is a separate "*mTouchCap_Config.h*" file for each PIC18, PIC24 and PIC32 demo.
mTouchCap_HardwareProfile. h	This file comprises the entire hardware configuration necessary for any application. This file is located in each demo folder. It configures system hardware related settings like system clock, ports, Max ADC channels.
MPLAB Project files	The entire MPLAB 8 project files i.e. files with extensions 'mcw', 'mcp' and MPLAB X project folder named as 'MPLAB.X' for different application demos.
mTouchCap_Display.c/h	This includes LED display handling example routines used for Cap Touch Eval board demos. This can be changed to suit user application.
mTouchCap_DirectKeys.c/h	This includes Direct key-board handling example routines used for this demo. This may be changed to suit the user application.

mTouchCap_MatrixKeys.c/h	This includes Matrix key-board handling example routines used for this demo. This file may be changed to suit the user application.
mTouchCap_4Ch_Slider.c/h	This includes 4-Channel slider board handling example routines used for this demo. This file may be changed to suit the user application.
mTouchCap_2Ch_Slider.c/h	This includes 2-Channel slider board handling example routines used for this demo. This file may be changed to suit the user application.

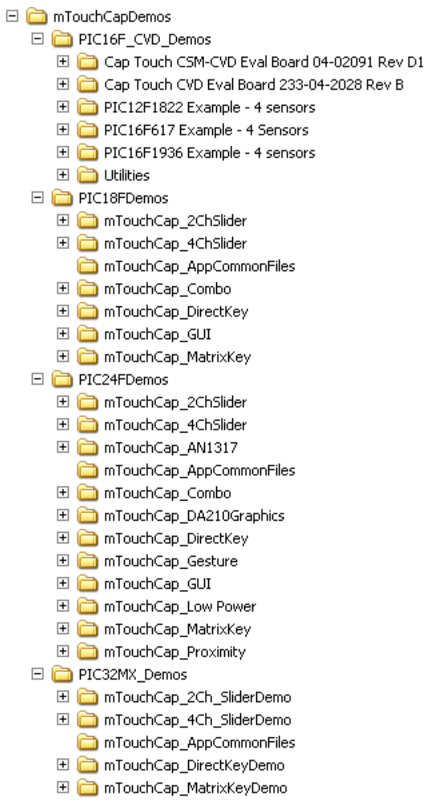
4 Folder Structure and Files

The folder and file structure of mTouchCap Software Library is shown below:



```
Microchip
    Help
    Include
        mTouchCap
            PIC16F CVD FW Library
            PIC18F_PIC24F CTMU Library
            PIC32MX CVD Library
    mTouchCap
        PIC16F CVD FW Library
            includes
        PIC18F_PIC24F CTMU Library
            Documentation
        PIC32MX CVD Library
            Documentation
```

The folder and file structure of mTouchCapDemos are as follows:



```
mTouchCapDemos
    PIC16F_CVD_Demos
        Cap Touch CSM-CVD Eval Board 04-02091 Rev D1
        Cap Touch CVD Eval Board 233-04-2028 Rev B
        PIC12F1822 Example - 4 sensors
        PIC16F617 Example - 4 Sensors
        PIC16F1936 Example - 4 sensors
        Utilities
    PIC18FDemos
        mTouchCap_2ChSlider
        mTouchCap_4ChSlider
        mTouchCap_AppCommonFiles
        mTouchCap_Combo
        mTouchCap_DirectKey
        mTouchCap_GUI
        mTouchCap_MatrixKey
    PIC24FDemos
        mTouchCap_2ChSlider
        mTouchCap_4ChSlider
        mTouchCap_AN1317
        mTouchCap_AppCommonFiles
        mTouchCap_Combo
        mTouchCap_DA210Graphics
        mTouchCap_DirectKey
        mTouchCap_Gesture
        mTouchCap_GUI
        mTouchCap_Low Power
        mTouchCap_MatrixKey
        mTouchCap_Proximity
    PIC32MX_Demos
        mTouchCap_2Ch_SliderDemo
        mTouchCap_4Ch_SliderDemo
        mTouchCap_AppCommonFiles
        mTouchCap_DirectKeyDemo
        mTouchCap_MatrixKeyDemo
```

You can add code and modules to the demo sub directories that will use and interact with the library. For example, you could add a folder named "Your Applications Directory" to the mTouchCapDemos folder that contains your application source code. The library specific folders are the following:

• The ..\Microchip folder will contain the library components.

• The Help sub-folder under ..\Microchip folder will contain this document (mTouch Software Library Help.chm file).

• The ..\mTouchCap sub-folder under the ..\Microchip folder is where the C files, documentation related to mTouch stack are located.

- The ..\mTouchCap sub-folder under the Include folder is where the Header files related to the mTouch stack are located.

5 API - mTouch Software Library

Functions

	Name	Description
◆	mTouchCapAPI_AutoAdjustChannel (see page 13)	This API is used to automatically adjust the voltage reading(charge level) on a channel. This will read the ADC value after charging the channel, and then adjust the CTMU current source Trim bits to read the value defined as per AUTO_ADJUST_BAND_PERCENT.
◆	mTouchCapAPI_CTMU_GetChannelReading (see page 14)	This API is used to get the channel reading. It initializes the CTMU and ADC module for the corresponding channel passed. It reads ADC data from the channel and returns.
◆	mTouchCapAPI_CTMU_SetupCurrentSource (see page 14)	This API sets the current source and trim level for a particular channel.
◆	mTouchCapAPI_getChannelTouchStatus (see page 15)	This API will determine if the channel which is associated with a particular key is touched or not. It will output the pressed or unpressed status of the channel based on the Decode method which is associated with the channel.
◆	mTouchCapAPI_GetStatusDirectButton (see page 16)	This API will provide the status of the Direct key passed which will be used by the application to perform the related task.
◆	mTouchCapAPI_GetStatusMatrixButton (see page 17)	This API will provide the status of the Matrix key passed which will be used by the application to perform the related task.
◆	mTouchCapAPI_GetStatusSlider2Ch (see page 17)	This API gets the percentage level of a particular 2-channel slider passed.The output is ratio-metrically calculated from 0% to 100% proportional to the finger on the slider **Parameters** **Parameters** | **Description** Button | Slider - Obje
◆	mTouchCapAPI_GetStatusSlider4Ch (see page 18)	This API gets the percentage level of a particular 4-channel slider passed.The output is ratio-metrically calculated from 0% to 100% proportional to the finger on the slider **Parameters** **Parameters** | **Descr** Button | Slider
◆	mTouchCapAPI_ScanChannelIterative (see page 18)	This API is used for scanning the channels, one at a time. This should be called in the Timer tick function, preferably in an interrupt. This will return the Averaged ADC value based on the SampleCount passed.
◆	mTouchCapAPI_SetUpChannelDirectKey (see page 19)	This API will setup the channel associated with the Direct key . The channel number, filter type and decode method are stored in the structure associated with the Direct Key.
◆	mTouchCapAPI_SetUpChannelMatrixKey (see page 20)	This API will setup the channels of the Row and Column associated with the Matrix key. The channel number of the Row and Column, filter type and decode method are stored in the structure associated with the corresponding channel.
◆	mTouchCapAPI_SetUpChannelSlider2Ch (see page 21)	This API will setup the 2 channels associated with the 2-channel Slider. The 2 channel numbers, filter type and decode method are stored in the structure associated with the corresponding 2-Channel Slider

◆	mTouchCapAPI_SetUpChannelSlider4Ch (see page 22)	This API will setup the 4 channels associated with the 4-channel Slider. The 4 channel numbers, filter type and decode method are stored in the structure associated with the corresponding 4-Channel Slider
◆	mTouchCapAPI_SetUpCTMU_Default (see page 22)	This API is for those who want to use the cap-touch application without having to tweak much. The API sets up the channel in a predefined default method with known configuration settings.

5.1 mTouchCapAPI_AutoAdjustChannel Function

File

mTouchCap_CtmuAPI.h

C

```
CHAR mTouchCapAPI_AutoAdjustChannel(WORD ChannelNum, WORD AdcValueToAchieve);
```

Side Effects

None

Returns

CHAR ChannelAdjResult

- -1 : FAILED
- 1 : PASSED

Description

This API is used to automatically adjust the voltage reading(charge level) on a channel. This will read the ADC value after charging the channel, and then adjust the CTMU current source Trim bits to read the value defined as per AUTO_ADJUST_BAND_PERCENT.

Parameters

Parameters	Description
WORD ChannelNum	CHANNEL_AN0 , CHANNEL_AN1 , CHANNEL_AN2 , CHANNEL_AN3 , CHANNEL_AN4 , CHANNEL_AN5 , CHANNEL_AN6 , CHANNEL_AN7 , CHANNEL_AN8 , CHANNEL_AN9 , CHANNEL_AN10 , CHANNEL_AN11 , CHANNEL_AN12 , CHANNEL_AN13 , CHANNEL_AN14 , CHANNEL_AN15 , CHANNEL_AN16 , CHANNEL_AN17 , CHANNEL_AN18 , CHANNEL_AN19 , CHANNEL_AN20 , CHANNEL_AN21 , CHANNEL_AN22 , CHANNEL_AN23, CHANNEL_AN24 , CHANNEL_AN25 , CHANNEL_AN26 , CHANNEL_AN27 //Check the availability of channels in the PIC MCU being used
WORD AdcValueToAchieve	ADC Value to be achieved

Function

CHAR mTouchCapAPI_AutoAdjustChannel (WORD ChannelNum, WORD AdcValueToAchieve)

PreCondition   :   Channel setup is complete.

5.2 mTouchCapAPI_CTMU_GetChannelReading Function

File

mTouchCap_CtmuAPI.h

C

```
WORD mTouchCapAPI_CTMU_GetChannelReading(WORD ChannelNum);
```

Side Effects

None

Returns

ChannelData :ADC value (Range for 10 bit ADC is 0 : 0x3FF)

Description

This API is used to get the channel reading. It initializes the CTMU and ADC module for the corresponding channel passed. It reads ADC data from the channel and returns.

Parameters

Parameters	Description
WORD ChannelNum	Channel number (must have enabled in "mTouchCAp_Config.h") CHANNEL_AN0 , CHANNEL_AN1 , CHANNEL_AN2 , CHANNEL_AN3 , CHANNEL_AN4 , CHANNEL_AN5 , CHANNEL_AN6 , CHANNEL_AN7 , CHANNEL_AN8 , CHANNEL_AN9 , CHANNEL_AN10 , CHANNEL_AN11 , CHANNEL_AN12 , CHANNEL_AN13 , CHANNEL_AN14 , CHANNEL_AN15 , CHANNEL_AN16 , CHANNEL_AN17 , CHANNEL_AN18 , CHANNEL_AN19 , CHANNEL_AN20 , CHANNEL_AN21 , CHANNEL_AN22 , CHANNEL_AN23, CHANNEL_AN24 , CHANNEL_AN25 , CHANNEL_AN26 , CHANNEL_AN27, //Check the availability of channels in the PIC MCU being used

Function

WORD mTouchCapAPI_CTMU_GetChannelReading(WORD ChannelNum)

PreCondition	:	Channel setup is complete

5.3 mTouchCapAPI_CTMU_SetupCurrentSource Function

File

mTouchCap_CtmuAPI.h

C

```
void mTouchCapAPI_CTMU_SetupCurrentSource(BYTE CurrentSourceRange, BYTE TrimValue);
```

Side Effects

None

Returns

None

Description

This API sets the current source and trim level for a particular channel.

Parameters

Parameters	Description
BYTE CurrentSourceRange	Current source range • CURRENT_RANGE_100XBASE_CURRENT //Current source Range is 100*Base current (55uA) • CURRENT_RANGE_10XBASE_CURRENT //Current source Range is 10*Base current (5.5uA) • CURRENT_RANGE_BASE_CURRENT //Current source Range is Base current (0.55uA) • CURRENT_SRC_DISABLED //Current source disabled
BYTE TrimValue	Trim settings

Function

void mTouchCapAPI_CTMU_SetupCurrentSource (BYTE CurrentSourceRange, BYTE TrimValue)

PreCondition : None

5.4 mTouchCapAPI_getChannelTouchStatus Function

File

mTouchCap_CtmuAPI.h

C

```
BYTE mTouchCapAPI_getChannelTouchStatus(WORD ChIndex, BYTE Decode_Method);
```

Side Effects

None

Returns

TouchStatus-Whether the key associated with the Channel is pressed or not KEY_NOT_PRESSED, KEY_PRESSED

Description

This API will determine if the channel which is associated with a particular key is touched or not. It will output the pressed or unpressed status of the channel based on the Decode method which is associated with the channel.

Parameters

Parameters	Description
WORD ChIndex	The Channel number.
BYTE Decode_Method	The type of Decode Method associated with that channel DECODE_METHOD_MOST_PRESSED, DECODE_METHOD_MULTIPLE_PRESS, DECODE_METHOD_PRESS_AND_RELEASE, DECODE_METHOD_PRESS_ASSERT

Function

BYTE mTouchCapAPI_getChannelTouchStatus(WORD ChIndex, BYTE Decode_Method)

PreCondition : None

5.5 mTouchCapAPI_GetStatusDirectButton Function

File

mTouchCap_CtmuAPI.h

C

```
BYTE mTouchCapAPI_GetStatusDirectButton(DirectKey * Button);
```

Side Effects

None

Returns

The Touch Status of the particular key. 0 = KEY_NOT_PRESSED 1 = KEY_PRESSED

Description

This API will provide the status of the Direct key passed which will be used by the application to perform the related task.

Parameters

Parameters	Description
DirectKey * Button	Object of the structure associated with the Direct Key

Function

BYTE mTouchCapAPI_GetStatusDirectButton(DirectKey *Button)

PreCondition : The Channel associatd with the Direct key should have been set up.

5.6 mTouchCapAPI_GetStatusMatrixButton Function

File

mTouchCap_CtmuAPI.h

C

```
BYTE mTouchCapAPI_GetStatusMatrixButton(MatrixKey * Button);
```

Side Effects

None

Returns

The Touch Status of the particular key. 0 = KEY_NOT_PRESSED 1 = KEY_PRESSED

Description

This API will provide the status of the Matrix key passed which will be used by the application to perform the related task.

Parameters

Parameters	Description
MatrixKey * Button	Object of the Structure associated with the Matrix Key

Function

BYTE mTouchCapAPI_GetStatusMatrixButton (MatrixKey *Button)

PreCondition : The Channels of the Row and Column associatd with the Matrix key should have been already set up

5.7 mTouchCapAPI_GetStatusSlider2Ch Function

File

mTouchCap_CtmuAPI.h

C

```
SHORT mTouchCapAPI_GetStatusSlider2Ch(Slider2Ch* Slider);
```

Side Effects

None

Returns

SliderLevel gives the Slider percent level of the touch.

Description

This API gets the percentage level of a particular 2-channel slider passed.The output is ratio-metrically calculated from 0% to 100% proportional to the finger on the slider

Parameters

Parameters	Description
Button	Slider - Object of the 2-channel slide

Function

SHORT mTouchCapAPI_GetStatusSlider2Ch (Slider2Ch *Slider)

PreCondition : 2-channel Slider setup is complete.

5.8 mTouchCapAPI_GetStatusSlider4Ch Function

File

mTouchCap_CtmuAPI.h

C

```
SHORT mTouchCapAPI_GetStatusSlider4Ch(Slider4Ch* Slider);
```

Side Effects

None

Returns

SliderLevel gives the Slider percent level of the touch.

Description

This API gets the percentage level of a particular 4-channel slider passed.The output is ratio-metrically calculated from 0% to 100% proportional to the finger on the slider

Parameters

Parameters	Description
Button	Slider - Object of the 4-channel slide

Function

SHORT mTouchCapAPI_GetStatusSlider4Ch (Slider4Ch *Slider)

PreCondition : 4-channel Slider setup is complete.

5.9 mTouchCapAPI_ScanChannelIterative Function

File

mTouchCap_CtmuAPI.h

C

```
WORD mTouchCapAPI_ScanChannelIterative(WORD ChannelNum, BYTE SampleCount);
```

Side Effects

None

Returns

RawData :Averaged ADC Value

Description

This API is used for scanning the channels, one at a time. This should be called in the Timer tick function, preferably in an interrupt. This will return the Averaged ADC value based on the SampleCount passed.

Parameters

Parameters	Description
WORD ChannelNum	Channel number CHANNEL_AN0 , CHANNEL_AN1 , CHANNEL_AN2 , CHANNEL_AN3 , CHANNEL_AN4 , CHANNEL_AN5 , CHANNEL_AN6 , CHANNEL_AN7 , CHANNEL_AN8 , CHANNEL_AN9 , CHANNEL_AN10 , CHANNEL_AN11 , CHANNEL_AN12 , CHANNEL_AN13 , CHANNEL_AN14 , CHANNEL_AN15 , CHANNEL_AN16 , CHANNEL_AN17 , CHANNEL_AN18 , CHANNEL_AN19 , CHANNEL_AN20 , CHANNEL_AN21, CHANNEL_AN22 , CHANNEL_AN23, CHANNEL_AN24 , CHANNEL_AN25 , CHANNEL_AN26 , CHANNEL_AN27, //Check the availability of channels in the PIC MCU being used
BYTE SampleCount	Count of Samples to be taken per scan

Function

WORD mTouchCapAPI_ScanChannelIterative (WORD ChannelNum, BYTE SampleCount)

PreCondition : Channel setup is complete.

5.10 mTouchCapAPI_SetUpChannelDirectKey Function

File

mTouchCap_CtmuAPI.h

C

```
BYTE mTouchCapAPI_SetUpChannelDirectKey(DirectKey * Button, BYTE Channel_number, WORD
Trip_Value, WORD Decode_Method, WORD Filter_Method);
```

Side Effects

None

Returns

SetUpStatus - Status of the Direct key(TRUE or FALSE).

Description

This API will setup the channel associated with the Direct key . The channel number, filter type and decode method are stored in the structure associated with the Direct Key.

Parameters

Parameters	Description
DirectKey * Button	Object of the Direct key structure Channel number : channel number of the object Button associated with corresponding direct key. Trip Value: Default Trip value for the channel specified by the channel number.
WORD Decode_Method	The Decode method associated with the Direct Key. DECODE_METHOD_MOST_PRESSED, DECODE_METHOD_MULTIPLE_PRESS, DECODE_METHOD_PRESS_AND_RELEASE, DECODE_METHOD_PRESS_ASSERT
WORD Filter_Method	The filter method associated with the Direct Key. FILTER_METHOD_SLOWAVERAGE=0, FILTER_METHOD_GATEDAVERAGE, FILTER_METHOD_FASTAVERAGE

Function

CHAR mTouchCapAPI_SetUpChannelDirectKey(DirectKey *Button,CHAR Channel_number, WORD Trip_Value, WORD Decode_Method, WORD Filter_Method)

PreCondition : None

5.11 mTouchCapAPI_SetUpChannelMatrixKey Function

File

mTouchCap_CtmuAPI.h

C

```
BYTE mTouchCapAPI_SetUpChannelMatrixKey(MatrixKey * Button, BYTE Row_Channel_Number, BYTE
Col_Channel_number, WORD Trip_Value, WORD Decode_Method, WORD Filter_Method);
```

Side Effects

None

Returns

SetUpStatus - Status of the Matrix key(TRUE or FALSE).

Description

This API will setup the channels of the Row and Column associated with the Matrix key. The channel number of the Row and Column, filter type and decode method are stored in the structure associated with the corresponding channel.

Parameters

Parameters	Description
MatrixKey * Button	Object of the Matrix key structure
BYTE Col_Channel_number	channel number of the Column associated with corresponding matrix key. Trip Value - Default trip value for the channel associated with corresponding matrix key

WORD Decode_Method	The Decode method associated with the corresponding Matrix key
WORD Filter_Method	The filter method associated with the corresponding Matrix key
Row_Channel_number	channel number of the Row associated with corresponding matrix key.

Function

BYTE mTouchCapAPI_SetUpChannelMatrixKey(MatrixKey *Button,BYTE Row_Channel_number,BYTE Col_Channel_number, WORD Trip_Value, WORD Decode_Method, WORD Filter_Method)

PreCondition : None

5.12 mTouchCapAPI_SetUpChannelSlider2Ch Function

File

mTouchCap_CtmuAPI.h

C

```
BYTE mTouchCapAPI_SetUpChannelSlider2Ch(Slider2Ch * Slider, BYTE Slider_Channel1_Number,
BYTE Slider_Channel2_Number, WORD Trip_Value, BYTE Decode_Method, BYTE Filter_Method);
```

Side Effects

None

Returns

SetUpStatus - Status of the 2-channel slider(TRUE or FALSE).

Description

This API will setup the 2 channels associated with the 2-channel Slider. The 2 channel numbers, filter type and decode method are stored in the structure associated with the corresponding 2-Channel Slider

Parameters

Parameters	Description
Slider2Ch * Slider	Object of the Matrix key structure Channel number - channel number of the object Slider associated with corresponding 2-channel slider. Trip Value - Trip value for the channels associated with the 2-channel slider
BYTE Filter_Method	One of the filter method for the 2-channel slider Decode_Method -The Decode method for the 2-channel slider

Function

BYTE mTouchCapAPI_SetUpChannelSlider2Ch(Slider2Ch *Slider, BYTE Slider_Channel1_number,BYTE Slider_Channel2_number, WORD Trip_Value, BYTE Decode_Method, BYTE Filter_Method)

5.13 mTouchCapAPI_SetUpChannelSlider4Ch Function

File

mTouchCap_CtmuAPI.h

C

```
BYTE mTouchCapAPI_SetUpChannelSlider4Ch(Slider4Ch * Slider, BYTE Slider_Channel1_Number,
BYTE Slider_Channel2_Number, BYTE Slider_Channel3_Number, BYTE Slider_Channel4_Number, WORD
Trip_Value, BYTE Decode_Method, BYTE Filter_Method);
```

Side Effects

None

Returns

SetUpStatus - Status of the 4-channel slider(TRUE or FALSE).

Description

This API will setup the 4 channels associated with the 4-channel Slider. The 4 channel numbers, filter type and decode method are stored in the structure associated with the corresponding 4-Channel Slider

Parameters

Parameters	Description
Slider4Ch * Slider	object of the 4-channel Slider structure Channel number : channel number of the object Slider associated with 4-channel slider. Trip Value: Trip value for the channels associated with the 4-channel slider.
BYTE Filter_Method	The filter method for the 4-channel slider. Decode_Method:The Decode method for the 4-channel slider.

Function

BYTE mTouchCapAPI_SetUpChannelSlider4Ch(Slider4Ch *Slider, BYTE Slider_Channel1_Number,BYTE Slider_Channel2_Number, BYTE Slider_Channel3_Number,BYTE Slider_Channel4_Number,WORD Trip_Value, BYTE Decode_Method, BYTE Filter_Method)

PreCondition : None

5.14 mTouchCapAPI_SetUpCTMU_Default Function

File

mTouchCap_CtmuAPI.h

C

```
void mTouchCapAPI_SetUpCTMU_Default(WORD ChannelNum);
```

Side Effects

None

Returns

None

Description

This API is for those who want to use the cap-touch application without having to tweak much. The API sets up the channel in a predefined default method with known configuration settings.

Parameters

Parameters	Description
WORD ChannelNum	Channel Number

Function

void mTouchCapAPI_SetUpCTMU_Default(WORD ChannelNum)

PreCondition : None

6 mTouchCap Demos

6.1 mTouchCap_DirectKey

****

mTouch Capacitive Touch Library Demonstration for DirectKeys

****

This demo contains the information related to the mTouch Capacitive Touch Software Library used for Directkeys. This demo uses 8 CTMU or CVD channels to perform 8 Directkeys functionality. Each key is directly connected to a CTMU or CVD Channel. When a touch is detected on the Key, a corresponding LED will light-up to indicate the touch.

****

mTouch Capacitive Touch Library Testing Details for Directkeys

****

The mTouch Capacitive Touch Library software is tested using enhanced mTouch Capacitive Touch Evaluation Kit boards (DM183026-2).

The mTouch Capacitive Touch Library software can be customized to customer's application specific boards using Set-up details mentioned in ReadMe.txt file located in the demo folder.

Note: Refer ReadMe.txt file for further details on DirectKeys Demo.

6.2 mTouchCap_MatrixKey

****

mTouch Capacitive Touch Library Demonstration for MatrixKeys

****

This demo contains the information related to the mTouch Capacitive Touch Software Library used for MatrixKeys. This demo uses 7 CTMU or CVD channels to perform a 4x3 MatrixKeys functionality. Each Matrix key is multiplexed with 2 CTMU or CVD Channels. When a touch is detected on the Key, a corresponding LED will light-up to indicate the touch.

****

mTouch Capacitive Touch Library Testing Details for MatrixKeys

****

The mTouch Capacitive Touch Library software is tested using enhanced mTouch Capacitive Touch Evaluation Kit boards (DM183026-2).

The mTouch Capacitive Touch Library software can be customized to customer's application specific boards using Set-up details mentioned in ReadMe.txt file located in the demo folder.

Note: Refer ReadMe.txt file for further details on MatrixKeys Demo.

6.3 mTouchCap_2ChSlider

****

mTouch Capacitive Touch Library Demonstration for 2 Channel-Slider

****

This Demo contains the information related to the mTouch Capacitive Touch Software Library used for a 2 Channel-Slider. This demo uses 2 CTMU or CVD channels to perform slider functionality. The output is ratio-metrically calculated from 0% to 100% proportional to the position of a finger on the slider.

****

mTouch Capacitive Touch Library Testing Details for 2 Channel-Slider

****

The mTouch Capacitive Touch Library software is tested using enhanced mTouch Capacitive Touch Evaluation Kit boards (DM183026-2).

The mTouch Capacitive Touch Library software can be customized to customer's application specific boards using Set-up details mentioned in ReadMe.txt file located in the demo folder.

Note: Refer ReadMe.txt file for further details on 2 Channel-Slider Demo.

6.4 mTouchCap_4ChSlider

****

mTouch Capacitive Touch Library Demonstration for 4 Channel-Slider

****

This demo contains the information related to the mTouch Capacitive Touch Software Library used for a 4 Channel-Slider. This demo uses 4 CTMU or CVD channels to perform slider functionality. The output is ratio-metrically calculated from 0% to

100% proportional to the position of a finger on the slider.

****

mTouch Capacitive Touch Library Testing Details for 4 Channel-Slider

****

The mTouch Capacitive Touch Library software is tested using enhanced mTouch Capacitive Touch Evaluation Kit boards (DM183026-2).

The mTouch Capacitive Touch Library software can be customized to customer's application specific boards using Set-up details mentioned in ReadMe.txt file located in the demo folder.

Note: Refer ReadMe.txt file for further details on 4 Channel-Slider Demo.

6.5 mTouchCap_Combo

****

mTouch Capacitive Touch Library Demonstration for Combo

****

This demo contains the information related to the mTouch Capacitive Touch Software Library used for the demonstration of different applications like Direct keys, Matrix keys, and sliders. In this demo, we can select Directkey, MatrixKey, 2-ch Slider and 4-ch Slider demo by enabling the macro for each of these demo which is defined in config.h. The explanation for each demo is given in the readme.txt which is available in the individual demo folder.

****

mTouch Capacitive Touch Library Testing Details

****

The mTouch Capacitive Touch Library software is tested using PIC24F CTMU Evaluation boards(DM183026-2).

The mTouch Capacitive Touch Library software can be customized to customer's application specific boards using Set-up details mentioned in ReadMe.txt file located in the demo folder.

Note: Refer ReadMe.txt file for further details on Combo Demo.

6.6 mTouchCap_GUI

****

mTouch Capacitive Touch Library Demonstration for mTouch Diagnostic Tool

****

This demonstrates the usage of mTouch Diagnostic Tool in MPLAB for any combination of Cap Touch Sensor boards.

****

mTouch Capacitive Touch Library testing Details for GUI Demo

****

The mTouch Capacitive Touch Library software is tested using PIC18F and PIC24F CTMU Evaluation boards(DM183026-2)

****

mTouch Capacitive Touch Library Set-Up Details for GUI Demo

****

The setup details for the selected Demo Boards are given in their respective Readme.txt file. The setup details for GUI (MPLAB Diagnostic Tool) related testing is given below.

Step 1:Select the "mTouch diagnostic Tool" option in the "Tools" tab from the MPLAB IDE project window. As soon as the option is selected, there will be a new "mTouch Diagnostic Tool" tab that will be added very next to the "Tools" tab in the MPLAB GUI.

Step 2:Select the "Settings" option in the mTouch Diagnostic Tool window.Within the "Settings" option

the below changes have to be done.

a). Select the "Board" Tab and choose the "Custom" option from the Select Board dropdown menu.

b). Select the "Sensor Count" value based on the total number of Channels used by the application.

c). Select the "Communication" tab and select the USB option for communicating with the

CTMU Eval Board.

d). Select the "Logging" tab and mention the location to save the log history.

Note: Refer ReadMe.txt file for further details on GUI Demo.

6.7 mTouchCap_DA210Graphics

****

mTouch Capacitive Touch Library Demonstration for DA210 Graphics

****

This demo contains information related to the mTouch Capacitive Touch Software Library for Cap Touch on the DA210 Demo Board, with integration with the Microchip Graphics Library.

****

mTouch Capacitive Touch Library testing Details for DA210

****

The mTouch Capacitive Touch Library software is tested using the PIC24FJ256DA210 Development Board, and Graphics Display Powertip 4.3" 480 x 272 Board.

****

Cap Touch Demonstration

****

The purpose of this demo is to interact with the graphics display objects on the display screen, and be able to change the display using the Cap Touch keys located on the DA210 Demo Board.

This is done by reading the cap touch keys, and then creating events that are fed to the graphics library code to change the display based on the cap touch key pressed.

The mTouch Capacitive Touch Library software can be customized to customer's application specific boards using Set-up details mentioned in ReadMe.txt file located in the demo folder.

Note: Refer ReadMe.txt file for further details on DA210 Graphics Demo.

6.8 mTouchCap_Gesture

****

mTouch Capacitive Touch Library Demonstration for Gesture

****

This demo contains the information related to the mTouch Capacitive Touch Software Library used for Gesture detection using CTMU on PIC24F Microcontroller. The purpose of this demo is to show the presence of swiping gesture on the Cap Touch keys.

****

mTouch Capacitive Touch Library Testing Details for Gesture

****

The mTouch Capacitive Touch Library software is tested using PIC24F CTMU Evaluation boards(DM183026-2). The Gesture Demo uses the Direct Key plug-in board.

The mTouch Capacitive Touch Library software can be customized to customer's application specific boards using Set-up details mentioned in ReadMe.txt file located in the demo folder.

Note: Refer ReadMe.txt file for further details on Gesture Demo.

6.9 mTouchCap_Proximity

****

mTouch Capacitive Touch Library Demonstration for Proximity

****

This demo contains information related to the mTouch Capacitive Touch Software Library for Proximity functionality demonstration. The purpose is to demonstrate proximity detection (detect fingers, hand or other objects approaching the matrix keypad.)

Note: In general, the Proximity mode can detect a person's hand within 1-2 inches. However, the PIC24F CTMU Evaluation board running the Proximity demo software may be sensitive to the surface material on which it is placed or objects in its vicinity, so results may vary.

The demo operates in 2 modes; Proximity and Matrix key.

****

mTouch Capacitive Touch Library testing Details for Proximity

****

The mTouch Capacitive Touch Library software is tested using PIC24F CTMU Evaluation boards(DM183026 -2)

The mTouch Capacitive Touch Library software can be customized to customer's application specific boards using Set-up details mentioned in ReadMe.txt file located in the demo folder.

Note: Refer ReadMe.txt file for further details on Proximity Demo.

6.10 mTouchCap_LowPower

****

mTouch Capacitive Touch Library Demonstration for LowPower

****

This demo is used to demonstrate the functionality of implementing a Low Power application

using the Direct Key Demo board.

****

mTouch Capacitive Touch Library testing Details for LowPower

****

The mTouch Capacitive Touch Library software is tested using PIC24F CTMU Evaluation board(DM183026-2)

The mTouch Capacitive Touch Library software can be customized to customer's application specific boards using Set-up details mentioned in ReadMe.txt file located in the demo folder.

Note: Refer ReadMe.txt file for further details on LowPower Demo.

6.11 mTouchCap_AN1317

****

mTouch Capacitive Touch Library Demonstration for AN1317

****

This demo includes the conducted noise immunity code using CTMU for capacitive touch application. For the details of implementation refer to AN1317

****

mTouch Capacitive Touch Library Testing Details for AN1317 Demo

****

The details of the demo is mentioned in ReadMe.txt file located in the demo folder.

Note: Refer ReadMe.txt file for further details on AN1317 Demo.

7 Limitations:

The known limitations of mTouch^{TM} software library version 1.31 are listed below:

- The array size of some of the variables used in mTouch stack is equal to maximum number of ADC Channels available in the device.
- The mTouch software Library supports PIC18F and PIC24F devices that have CTMU module.
- The current version of mTouch software Library supports CVD technique for PIC16F and PIC32MX Microcontrollers (MCUs).

Index