Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

THIS SPEC IS OBSOLETE

Spec No: 002-05228

Spec Title: AN205228 - F2MC-8FX Family, MB95410H/470H series TSC_GPIO.LIB

Replaced by: NONE

AN205228

F²MC-8FX Family, MB95410H/470H Series TSC_GPIO.LIB

This application note describes Cypress TSC_GPIO library, which is based on latest Capacitance Touch Sensor (TSC) GPIO algorithm. Added to that is the descriptions of how to use TSC_GPIO library and some notices.

OBSOLETE

Contents

1 Introduction....1
1.1 Purpose....1
1.2 Document Overview....1
2 GPIO Solution....1
3 Library....4
3.1 Library Overview....4
3.2 Parameters Setup....5
3.3 Application Interface....8
3.4 Cypress TSC Performance....9
3.5 How to Add Cypress TSC_GPIO.lib....10
3.6 How to use Cypress TSC_GPIO.lib....13
4 LIB Usage Notice....15
5 Additional Information....16
5.1 Sample Code....17
6 Document History....19

1 Introduction

1.1 Purpose

This application note describes Cypress TSC_GPIO library, which is based on latest Capacitance Touch Sensor (TSC) GPIO algorithm. Added to that is the descriptions of how to use TSC_GPIO library and some notices.

1.2 Document Overview

The rest of document is organized as the following:

Chapter 2 explains the working principles of GPIO solution.

Chapter 3 explains how to use TSC library.

Chapter 4 explains LIB usage notice.

2 GPIO Solution

This chapter introduces working principles of GPIO solution.

The theory of capacitance touch sensor is to check capacitance increment. As follows, when finger not touch the

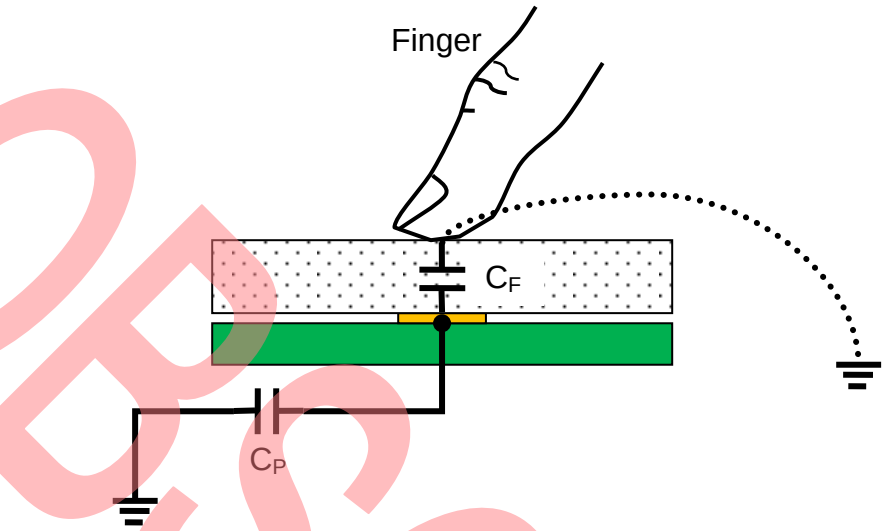
$$C = C_P \quad (2.1)$$

While the pad is touching, equation become

$$C = C_P + C_F \quad (2.2)$$

Where the increment is $\Delta C = C_F$.

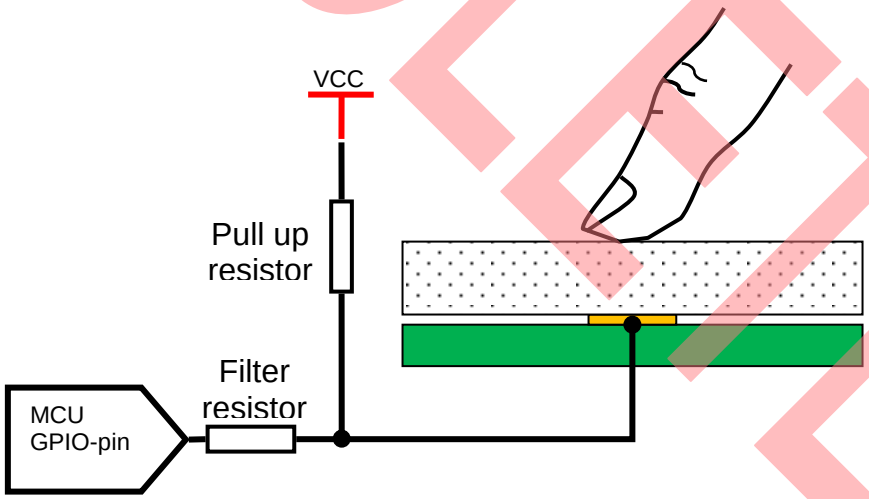
Figure 1. Diagram of Capacitance Touch Sensor Theory



According to the characteristics of capacitor, the deposited charge of capacitor increases along with the increase of capacitance.

Cypress uses GPIO method to check the capacitance change.

Figure 2. Diagram of GPIO algorithm



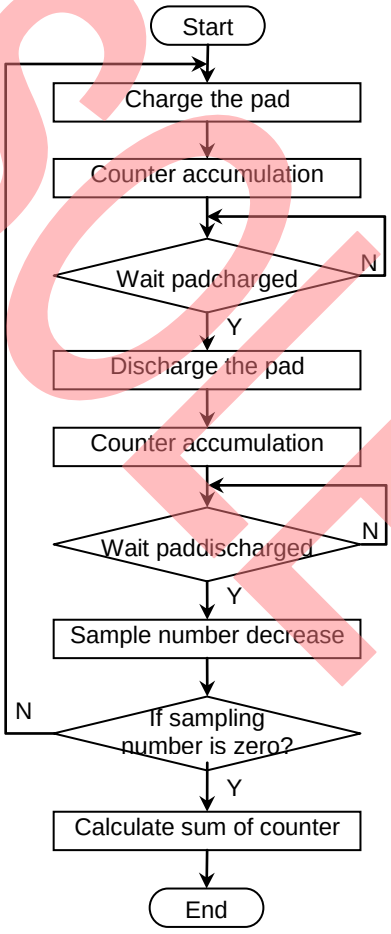
OBSOLETE

The method includes the following steps:

1. Set pin to high impedance to charge the pad;
2. Begin counter accumulation;
3. Wait until the pad is charged;
4. Set pin to 'L' to discharge the pad;
5. Wait until the pad is discharged;
6. Save counter value;
7. Sample number decrease;
8. If sample number isn't zero, start next sample loop;
9. If sample number is zero, calculate sum of counter, finish sample;

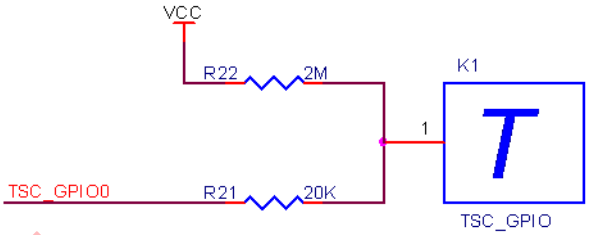
Flowchart and hardware connection are shown in Figure 3 and Figure 4.

Figure 3. Check Flowchart



OBSOLETE

Figure 4. Hardware Connection



3 Library

This chapter introduces how to use TSC library.

3.1 Library Overview

There are parameters which user need to setup, and 4 functions as API for user's situations. All the parameters and functions are introduced as follow.

Table 1. Parameters List

Name	Description	Remarks
TSCKEY_Buttonx_Used	Decide if TSCKEY Button x be Used	N/A
TSCKEY_IndividualKeyUsed	Decide if Individual Key will be Used	N/A
TSCKEY_SlideKeyUsed	Decide if Slide Key will be Used	N/A
TSCKEY_MinificationEnable	Decide if need minification of baseline value	N/A
TSCKEY_InterpolationEnable	Decide if need interpolation of slide position	N/A
TSCKEY_KeyNum	Total GPIO TSCKey number	N/A
TSCKEY_SlideKeyNum	TSCKey GPIO slide key number	N/A
TSCKEY_IIRShiftConstLx	Shift Const Level x for IIR filter	N/A
TSCKEY_IIRNumConstLx	Number of Shift Times Const Level x for IIR filter	N/A
TSCKEY_OffsetLx	Offset Const Level x for individual keys	N/A
TSCKEY_SubAvgShiftConst	Sub driver layer average shift const	N/A
TSCKEY_AvoidErrorConst	Avoid mistake in initialize process	N/A
TSCKEY_NoiseOffsetConst	Avoid mistake in initialize process	N/A
TSCKEY_DeltaBucketThrConst	Delta value threshold const	N/A
TSCKEY_PercentSumConst	Sum const of all keys	N/A
TSCKEY_MinificationConst	Baseline value minificateconst	N/A
TSCKEY_BalnUpdateConst	Basline update couterconst	N/A
TSCKEY_AmpShiftConst	Aamplify shift const	N/A
TSCKEY_DebounceConst	Debounceconst	N/A
TSCKEY_DebounceMinuConst	Debounceconst	N/A

OBSOLETE

Table 2: Functions List

Prototype	Function Description	Remarks
void TSCKey_Init (unsigned char Key_Num)	Initialize TSC module	N/A
void TSCKey_SampleUnit (unsigned char TSCKEY_SampleNum)	Bottom layer sample unit of TSC.	N/A
void TSCKey_Filter (unsigned char Key_Num, unsigned char Order_Shift)	Filter function of TSC sample value	N/A
unsigned int TSCKey_GetValue (unsigned char Key_Num)	Judgment touch status and coding key value	N/A

3.2 Parameters Setup

3.2.1 Port and Pin define

The driver layer function composed by assembly language, so the related ports and pins defined based on assembly language.

```
#pragma asm
/* \brief TSCKEY Data Port Define */
TSCKEY_Button0_Data         .equ   0x0000        /* TSCKey pin data  >> Port 5 */
TSCKEY_Button1_Data         .equ   0x0000        /* TSCKey pin data  >> Port 0 */
TSCKEY_Button2_Data         .equ   0x0000        /* TSCKey pin data  >> Port 0 */
TSCKEY_Button3_Data         .equ   0x0000        /* TSCKey pin data  >> Port 0 */
/* \brief TSCKEY Direction Port Define */
TSCKEY_Button0_Dir          .equ   0x0001        /* TSCKey pin dir   >> Port 5 */
TSCKEY_Button1_Dir          .equ   0x0001        /* TSCKey pin dir   >> Port 0 */
TSCKEY_Button2_Dir          .equ   0x0001        /* TSCKey pin dir   >> Port 0 */
TSCKEY_Button3_Dir          .equ   0x0001        /* TSCKey pin dir   >> Port 0 */
/* \brief TSCKEY Sample Pin Define */
TSCKEY_Button0_Bit          .equ   0             /* TSCKey pin       >> P52 */
TSCKEY_Button1_Bit          .equ   1             /* TSCKey pin       >> P00 */
TSCKEY_Button2_Bit          .equ   2             /* TSCKey pin       >> P01 */
TSCKEY_Button3_Bit          .equ   3             /* TSCKey pin       >> P02 */
/* \brief Timer used Define */
TSC_TimerDataUp             .equ   0xf94
TSC_TimerDataLow            .equ   0xf95
TSC_TimerEn                 .equ   0x37:7
#pragma endasm
```

OBSOLETE

3.2.2 Sample Parameters setup

The parameters:

```
#define TSCKEY_IndividualKeyUsed          TRUE   /* Use Individual Key or not */
#define TSCKEY_SlideKeyUsed               FALSE  /* Use Slide Key or not */
#define TSCKEY_MinificationEnable         FALSE  /* Compression of baseline value */
#define TSCKEY_InterpolationEnable        FALSE  /* Interpolation of slide pos*/
#define TSCKEY_KeyNum                     4      /* Total GPIO TSCKey number */
#define TSCKEY_SlideKeyNum                0      /* TSCKey GPIO slide key number */
#define TSCKEY_IIRShiftConstL1            1      /* IIR filter Shift Level 1 */
#define TSCKEY_IIRShiftConstL2            2      /* IIR filter Shift Level 2 */
#define TSCKEY_IIRShiftConstL3            3      /* IIR filter Shift Level 3 */
#defineTSCKEY_IIRShiftConstL4             5      /* IIR filter Shift Level 4 */
/* IIR filter Approximating Const Level 1 */
#define TSCKEY_IIRNumConstL1              2<<TSCKEY_IIRShiftConstL1
/* IIR filter Approximating Const Level 2 */
#define TSCKEY_IIRNumConstL2              2<<TSCKEY_IIRShiftConstL2
/* IIR filter Approximating Const Level 3 */
#define TSCKEY_IIRNumConstL3              2<<TSCKEY_IIRShiftConstL3
/* IIR filter Approximating Const Level 4 */
#define TSCKEY_IIRNumConstL4              2<<TSCKEY_IIRShiftConstL4
#define TSCKEY_OffsetL0                   40     /* Offset Const Level 0 */
#define TSCKEY_OffsetL1                   100    /* Offset Const Level 0 */
#define TSCKEY_OffsetL2                   170    /* Offset Const Level 0 */
#define TSCKEY_OffsetL3                   200    /* Offset Const Level 0 */
#define TSCKEY_OffsetL4                   300    /* Offset Const Level 0 */
#define TSCKEY_SubAvgShiftConst           1      /* Average shift const */
#define TSCKEY_NoiseOffsetConst           100    /* Offset const of noise */
#define TSCKEY_DeltaBucketThrConst        50     /* Delta value threshold const */
#define TSCKEY_PercentSumConst            180    /* Percent sum const */
#define TSCKEY_MinificationConst          3      /* Baseline compression const */
#define TSCKEY_BalnUpdateConst            1      /* Baseline update counter const */
#define TSCKEY_AmpShiftConst              0      /* Amplify shiftconst */
```

Used to configure the sample number, threshold, and timer initial value.

OBSOLETE

3.2.3 Structure Type Define

Each TSC Key has 4 attributes: original value, filtered value, noise threshold and difference percent value.

```
typedefstruct
{
    unsignedintOriginValue;              /* Contains the original value */
    unsignedintValue;                    /* Contains the filtered value */
    unsignedintNoiseThreshold;           /* Contains the noise threshold */
    unsignedintPercent;                  /* Contains the value percent */
} Key_Const;
```

User can define new key directly using the following structure.

```
typedefstruct
{
       unsignedintValue;    /* Contains the current baseline value */
       signedintDeltaBucket;      /* Contains the key threshold level */
} Basline_Const;
```

OBSOLETE

3.3 Application Interface

All the functions supplied by the TSC_GPIO.lib will be introduced below, include the function prototype, input parameter(s), return value(s), and the function description.

3.3.1 TSCKey_Init

Prototype	void TSCKey_Init(unsigned char Key_Num)
Parameter:	unsigned char Key_Num Indicate the number of touch keys need initialization
Return	void
Description	Initialize LCD module 1. Disables analog function of TSC pin. 2. Initialize counter 3. Initialize pin status to discharge TSC pad 4. Get baseline and noise threshold 5. Initialize used flag
Remark	N/A

3.3.2 TSCKey_SampleUnit

Prototype	void TSCKey_SampleUnit(unsigned char TSCKEY_SampleNum)
Parameter:	unsigned char TSCKEY_SampleNum Indicate sample number
Return	void
Description	Bottom layer sample unit of TSC.
Remark	N/A

3.3.3 TSCKey_Filter

Prototype	void TSCKey_Filter(unsigned char Key_Num, unsigned char Order_Shift)
Parameter:	unsigned char Key_Num Indicate the number of touch keys need filter unsigned char Order_Shift Indicate the shift number of order for filter
Return	void
Description	Filter function of TSC sample value
Remark	N/A

3.3.4 TSCKey_GetValue

Prototype	void TSCKey_GetValue(unsigned char Key_Num)
Parameter:	unsigned char Key_Num Indicate the number of touch keys need filter
Return	Key word of individual keys in lower byte; Slide position of slider in upper byte
Description	Judgment touch status and coding key value
Remark	N/A

OBSOLETE

3.4 Cypress TSC Performance

Table 3: Performance

Resource	Amount	Description
ROM	1272 Byte	ROM space: If just need support 1 sensor, ROM of lib can be reduced to: 300 Byte If just need support 2 sensors, ROM of lib can be reduced to: 960 Byte If just need support 3 sensors, ROM of lib can be reduced to: 1120 Byte If need support 4 sensors, ROM of Lib is:1272 Byte
RAM	36 Byte	RAM space: If just need support 1 sensor, RAM of lib can be reduced to: 11 Byte If just need support 2 sensors, RAM of lib can be reduced to: 26 Byte If just need support 3 sensors, RAM of lib can be reduced to: 31 Byte If need support 4 sensors, RAM of Lib is:36 Byte
IO Port	1	P0
IO Pin	4	P04,P05,P06,P07
Timer	2	Time base Timer
Serial Port	1	Serial Port(P04,P05),only use in check threshold mode
Machine Clock	>=8M	
Scan 1 Key	2 ms	

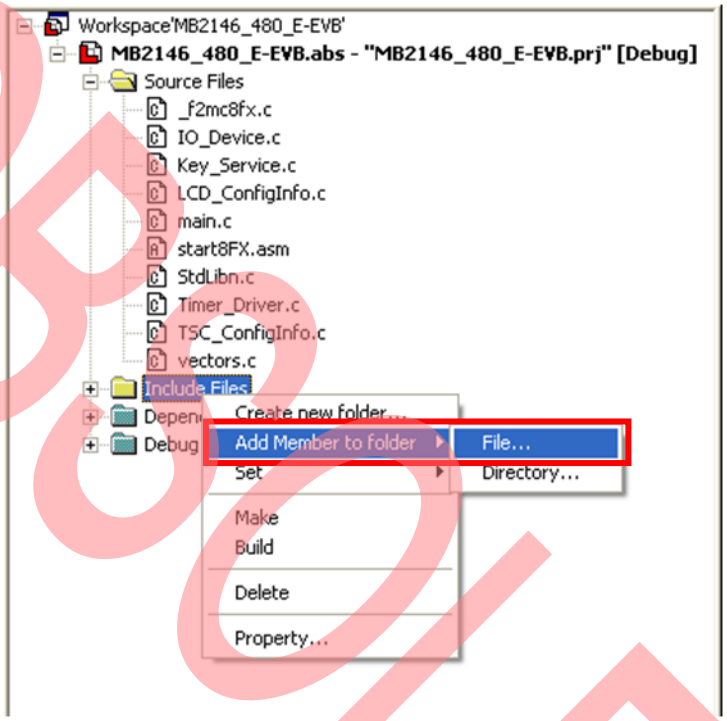
OBSOLETE

3.5 How to Add Cypress TSC_GPIO.lib

3.5.1 Add Cypress TSC_GPIO.lib to User's Project

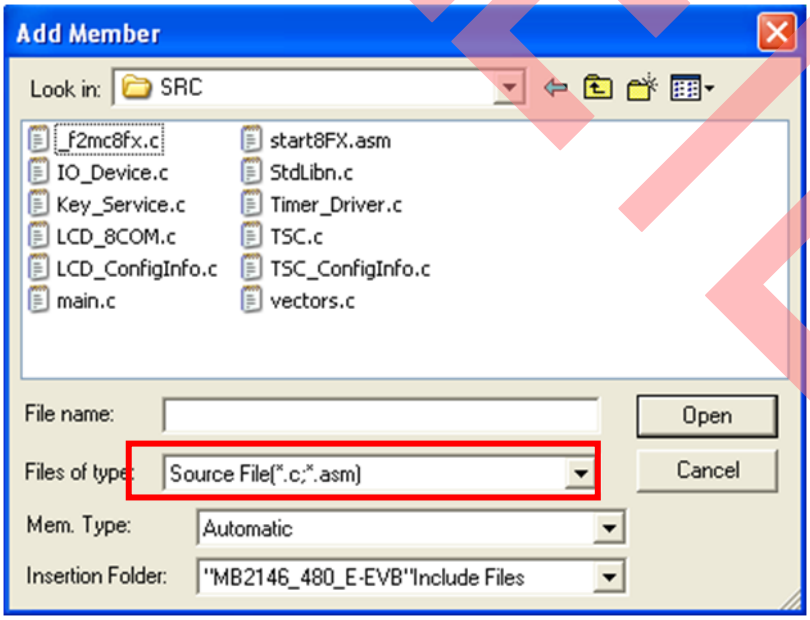
1. In Softune, Right click on folder *Include Files*→ select *Add member to folder* from the menu →select *File*.

Figure 5. Add member to folder



2. Because the default option of file type filters is *.c and * .asm, you can't found *TSC.lib* in dialog box of *Add Member*.

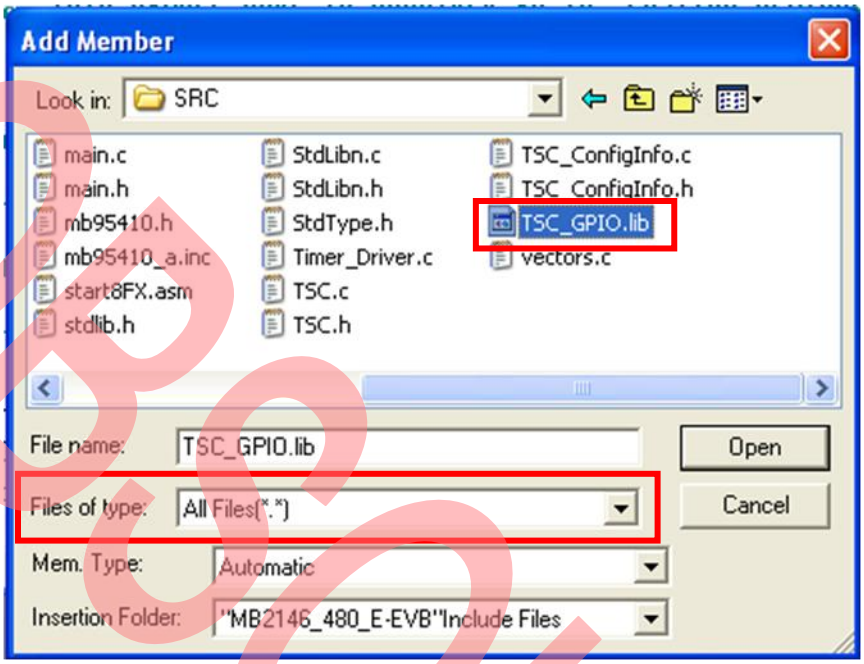
Figure 6. Popup Add Member dialog box



OBSOLETE

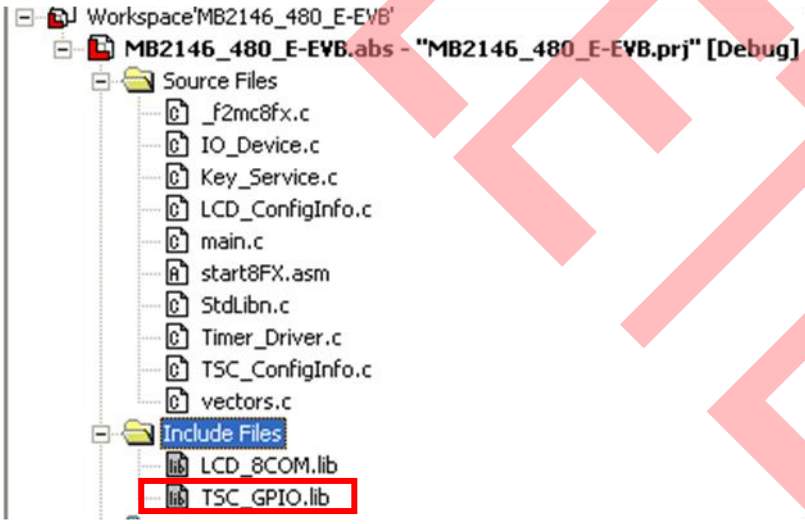
3. In *Add Member* dialog box, select ‘ALL Files’ from ‘Files of Type’, and then you will find the TSC.lib

Figure 7. Found the lib file



4. Double click TSC.lib, and then you can see it has been added in the folder *Include Files*

Figure 8. Add TSC.lib

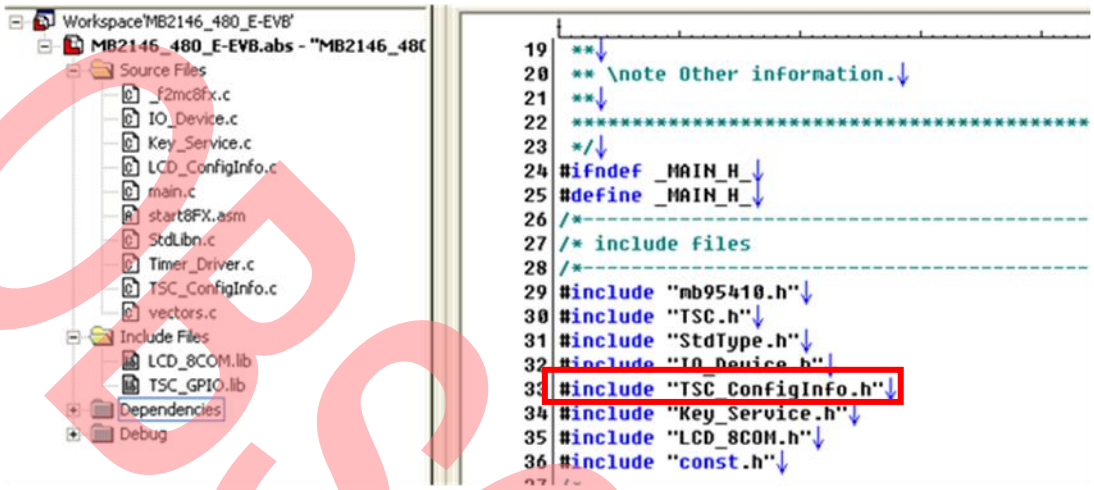


OBSOLETE

3.5.2 Include Header File

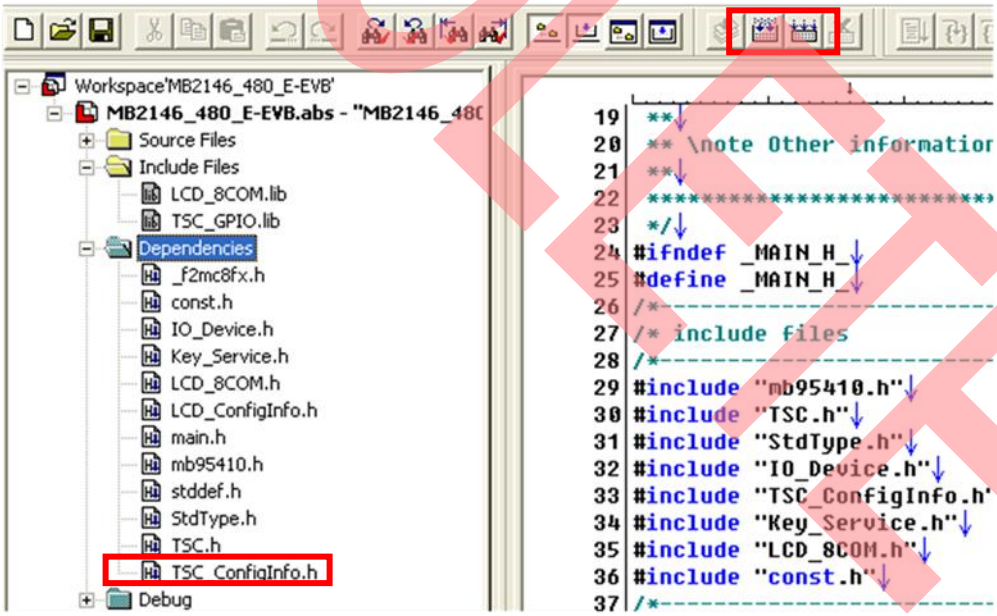
1. Add "#include "TSC_ConfigInfo.h "" in header file, such as in "main.h ".

Figure 9. Add include statement in C file



2. Compile the whole project, "TSC_ConfigInfo.h" will link TSC_GPIO.lib to c file, so that user program can use API functions in TSC_GPIO.lib.

Figure 10. Include header file successfully

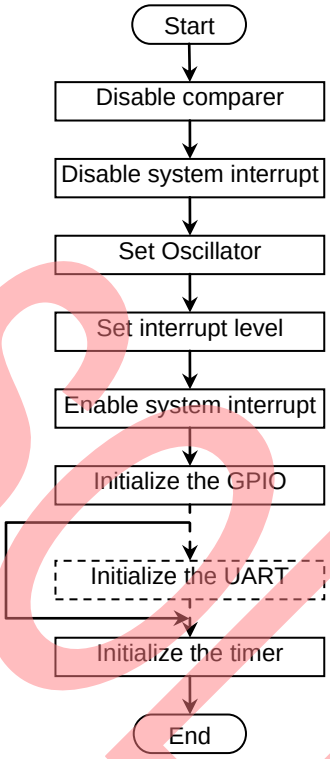


OBSOLETE

3.6 How to use Cypress TSC_GPIO.lib

After complete above works, we can start use the TSC library now. The operation categorized by if connect to GUI. Both of two operations need initialize TSC, initialize Timer, initialize GPIO and initialize interrupt. If connect to GUI, the UART module should also be initialized.

Figure 11. Initialize Flow chart



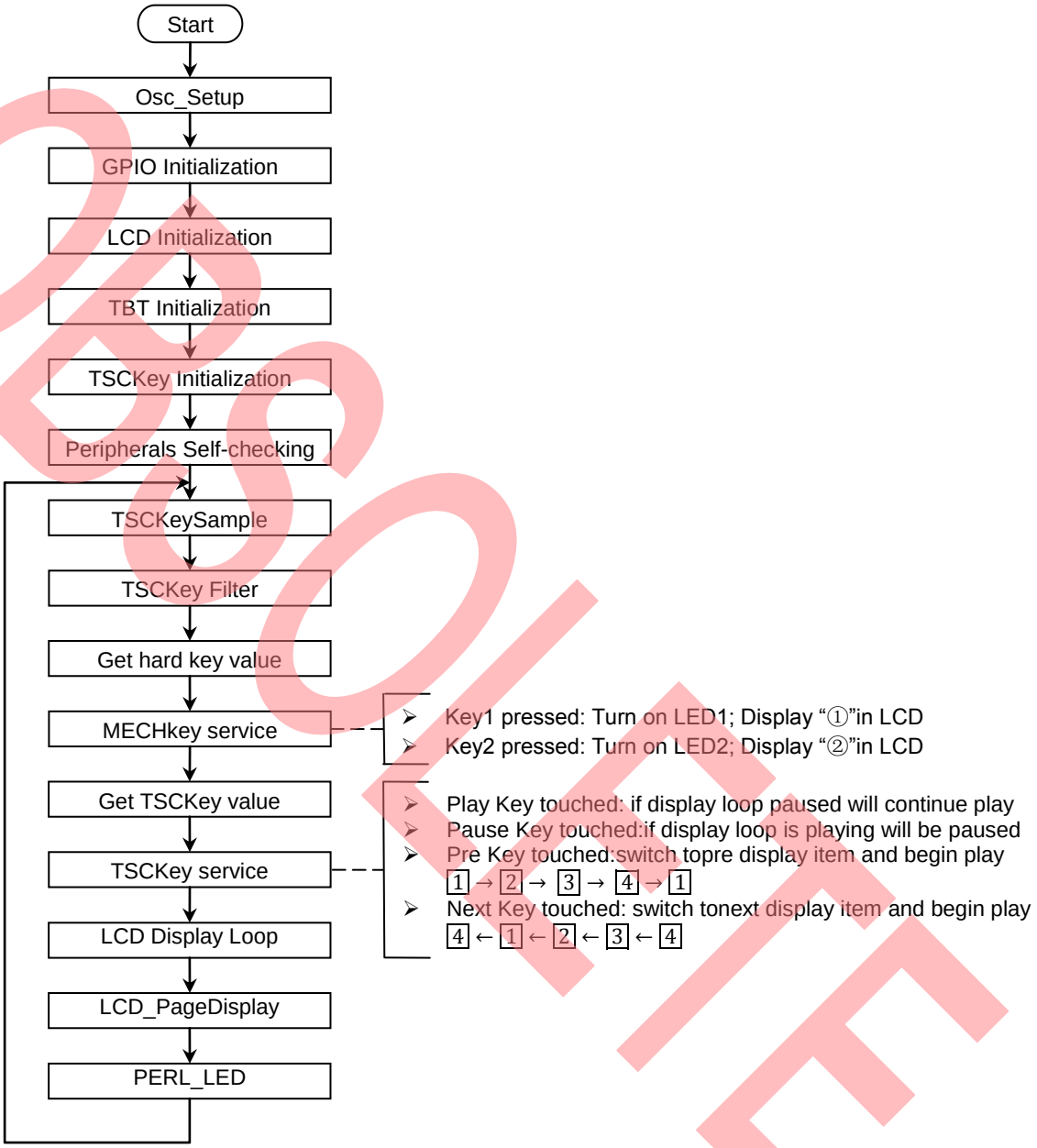
After all used modules have been initialized, program goes into a endless loop, to do below step circularly:

1. Wait the time interval interrupt is generate, clear the interrupt flag
2. Check the electrodes and save the counter value
3. Judgement if the new value bigger or lower than the threshold, if meet the condition, generate the key word
4. According to the key word jump to corresponding operation (such as drive LED)
5. Go to another new loop

If connect GUI, in above process, there also have the UART interrupt, MCU need send out counter value or get new threshold from GUI. The direct of transmission is determined by the user-defined protocol.

OBSOLETE

Figure 12. Main loop Flowchart



OBSOLETE

4 LIB Usage Notice

This chapter introduces LIB usage notice.

- Machine clock

The machine clock should be set to 8M or above. If the machine clock is less than 8M, the sensor response is slow.

- Interrupt

This solution needs to use Time base Timer interrupt to check sensor, and the interval time is 512us. It is unnecessary to use UART if the threshold is not gotten or tested with TSC GUI, and the interrupt setting in “vectors.c” as following:

```
#include "mb95410.h"
/*****************************************************************************
InitIrqLevels()
   This function  pre-sets all interrupt control registers. It can be used
to set all interrupt priorities in static applications. If this file
contains assignments to dedicated resources, verify  that the
appropriate controller is used.
*****************************************************************************/
voidInitIrqLevels(void)
{
       #ifdefenableUART
       ILR2 = 0xFE;          //  IRQ8:  LIN-UART (transmission)
       #else
       ILR2 = 0xFF;          //  IRQ8:  LIN-UART (transmission)
       #endif
       ILR4 = 0x3F;          //  IRQ19: Timebase timer
}
/*****************************************************************************
Prototypes
Add your own prototypes here. Each vector definition needs is prototype.
Either do it here or include a header file containing them.
*****************************************************************************/
__interrupt voidDefaultIRQHandler (void);
#ifdefenableUART
__interrupt void UART_ISR (void);
#endif
__interrupt void TBT_ISR (void);
```

OBSOLETE

```
/*****************************************************************************
   Vector definition

Use following statements to define vectors.
   All resource related vectors are predefined.
   Remaining software interrupts can be added hereas well.
*****************************************************************************/
#ifdefenableUART
#pragmaintvect UART_T                   8   //  IRQ8:  LIN-UART (transmission)
#else
#pragmaintvectDefaultIRQHandler         8   //  IRQ8:  LIN-UART (transmission)
#endif
#pragmaintvect TBT_ISR                  19  //  IRQ19: Timebase timer
```

OBSOLETE

5 Additional Information

For more Information on Cypress Semiconductor products, visit the following websites:

English version address:

www.cypress.com/cypress-microcontrollers

Chinese version address:

www.cypress.com/cypress-microcontrollers-cn

5.1 Sample Code

5.1.1 Main Function

Name: Main Function

Function: Initialize and configure.

main.c

```
/*!
 ******************************************************************************
 **
 ** \file main.c
 **
 ** $Id: main.c V2.0.0 2012.2.20 14:50 PM Lee.Song $
 **
 ** \brief .
 **
 ** Add here more detailed description if needed ...
 **

 **
*/
/*---------------------------------------------------------------------------*/
/* include files                                                             */
/*---------------------------------------------------------------------------*/
#include "main.h"
/*---------------------------------------------------------------------------*/
/* constants and macros                                                      */
/*---------------------------------------------------------------------------*/
volatile unsigned int LCD_DispTaskTime = 0;
RUN_FLAG run_flag;
/*---------------------------------------------------------------------------*/
/* local functions                                                           */
/*---------------------------------------------------------------------------*/
void Osc_Setup(void)
{
        #if EXClock_Used
                SYSC = 0xBF;
                SYCC = 0xF0;
                WATR = 0xF3;
                SYCC2 = 0xF4;
        #else
                SYCC = 0xF0;
                SYCC2 = 0xE5;
                CRTH_CRSEL0 = 1;
                CRTH_CRSEL1 = 0;
        #endif
        while(STBC_MRDY == 0);
}
/*---------------------------------------------------------------------------*/
/* local functions                                                           */
/*---------------------------------------------------------------------------*/
void RESET_WATCHDOG(void)
{
        WDTC = 0x35;
}
void main(void)
{
        unsigned char LoopIndex;
        CMR0_VCID = 1;    /* Set for use GPIO Disable CMP input */
        __DI();
        Osc_Setup();              /* Setup MCU main oscillator/FLL */
        InitIrqLevels(); /* Initialise Interrupt level and IRQ vector table */
        __EI();          /* Enable system interrupt now */
```

OBSOLETE

```
        /* Module initialization */
        GPIO_Init();
        LCD_Init();              /* LCD module initialization */
        TBT_Init();              /* Time Base Timer module initialization */
        TSCKey_Init(TSCKEY_KeyNum);       /* TSC Key module initialization */
        /* Load initial value */
        LCD_DispTaskTime = Now_Time;      /* Initial LCD display task time */
        run_flag.TSC_Play = TRUE;         /* Initial flag as TRUE */
        run_flag.LCD_PlayFirstLoop = TRUE;/* Initial first play flag as TRUE */
        run_flag.RunStatusLEDFilp = TRUE; /* Initial LED flip flag as TRUE */
        /* LCD display for check */
        LCD_AllDisplay(LCD_BUFF_LEN);/* Test the LCD hardware */
        /* LED display for check */
        LED1_On;
        LED2_On;
        /* Hold on display period of time for check */
        while((Now_Time - LCD_DispTaskTime) < LCD_DispTimeInterval);
        LCD_DispTaskTime = Now_Time;
        /* LCD shut down */
        LCD_PageDisplay();
        /* LED shut down */
        LED1_Off;
        LED2_Off;
        /* Beep ringing twice */
        PERL_Beep(2,300);
        /* Load loop display value to temp buffer */
        for(LoopIndex=0;LoopIndex<8;LoopIndex++)
        {
                LCD_DispBuffTemp[LoopIndex] = LCD_LoopDispBuff[LoopIndex];
        }
        /* Endless loop */
        while(1)
        {
                /* TSC key sample unit */
                TSCKey_SampleUnit(TSCKEY_SampleNumConst);
                /* TSC key fiter function */
                TSCKey_Filter(TSCKEY_KeyNum,TSCKEY_IIRShiftConstL4);
                /* Machanical key fiter function */
                MECHKey_Filter();
                /* Machanical key service */
                MECHKey_Service();
                /* TSC key service */
                TSCKey_Service();
                /* LCD display loop judgement */
                if(Now_Time - LCD_DispTaskTime >= LCD_DispTimeInterval)
                {
                        LCD_DispTaskTime = Now_Time;
                        if(run_flag.TSC_Play == TRUE)
                        {LCD_StepElectrovalency(((LCD_LoopIndex/LOGO_BUFF_LEN)
                                                        + 1),1);
                                LCD_DispLoop(LCD_DispBuffTemp);
                        }
                        /* System run status indication LED */
                        if(run_flag.RunStatusLEDFilp == TRUE)
                        {
                                run_flag.RunStatusLEDFilp = FALSE;
                                PERL_LED(1,LED_On);
                                PERL_LED(2,LED_Off);
                        }
                        else
                        {
                                run_flag.RunStatusLEDFilp = TRUE;
                                PERL_LED(1,LED_Off);
                                PERL_LED(2,LED_On);
                        }
                }
                /* LCD display */
                LCD_PageDisplay();        /* Display various data page on LCD screen */
        }
}
```

OBSOLETE

6 Document History

Document Title: AN205228 - F²MC-8FX Family, MB95410H/470H series TSC_GPIO.LIB

Document Number: 002-05228

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	-	02/23/2012	Initial release
*A	5293296	AMYC	10/27/2016	There is no link for TSC_GPIO library, So this AN is for obsolete.

OBSOLETE

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Lighting & Power Control	cypress.com/powerpsoc
Memory	cypress.com/memory
PSoC	cypress.com/psoc
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless/RF	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community

Forums | Projects | Videos | Blogs | Training| Components

Technical Support

cypress.com/support

PSoC is a registered trademark and PSoC Creator is a trademark of Cypress Semiconductor Corporation. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

© Cypress Semiconductor Corporation, 2012-2016. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.

OBSOLETE