The following document contains information on Cypress products. The document has the series name, product name, and ordering part numbering with the prefix “MB”. However, Cypress will offer these products to new and existing customers with the series name, product name, and ordering part number with the prefix “CY”.

How to Check the Ordering Part Number

1. Go to www.cypress.com/pcn.
2. Enter the keyword (for example, ordering part number) in the **SEARCH PCNS** field and click **Apply**.
3. Click the corresponding title from the search results.
4. Download the Affected Parts List file, which has details of all changes

For More Information

Please contact your local sales office for additional information about Cypress products and solutions.

About Cypress

Cypress is the leader in advanced embedded system solutions for the world's most innovative automotive, industrial, smart home appliances, consumer electronics and medical products. Cypress' microcontrollers, analog ICs, wireless and USB-based connectivity solutions and reliable, high-performance memories help engineers design differentiated products and get them to market first. Cypress is committed to providing customers with the best support and development resources on the planet enabling them to disrupt markets by creating new product categories in record time. To learn more, go to www.cypress.com.

AN205506

F²MC-8FX Family, MB95200 Series Keyboard development using external interrupt

This application note describes how to design a keyboard with the external interrupt and also illustrates the theories of external interrupt with example.

1 Introduction

There are three methods to design a keyboard with the MB95200 series MCU. (External interrupt/AD/matrix)

This document describes how to design a keyboard with the external interrupt.

This document also illustrates the theories of external interrupt with example.

2 Keyboard Development

This chapter introduces the Function of using external interrupt to realize the keyboard.

2.1 Function of External Interrupt Circuit

The external interrupt circuit is used to detect the edge of a signal inputted to an external interrupt pin and generate an interrupt request to the interrupt controller. As an interrupt wakes up the microcontroller from standby mode to return to its normal operation state, the operation mode can be changed by inputting a signal to the external interrupt pin.

2.2 Jittering Elimination

Jittering elimination is a problem in keyboard design. There are two methods to solve this problem. One is to add a capacitor in the hardware circuit and the other is to create a delay by using the 8/16-bit composite timer.

3 Hardware Design

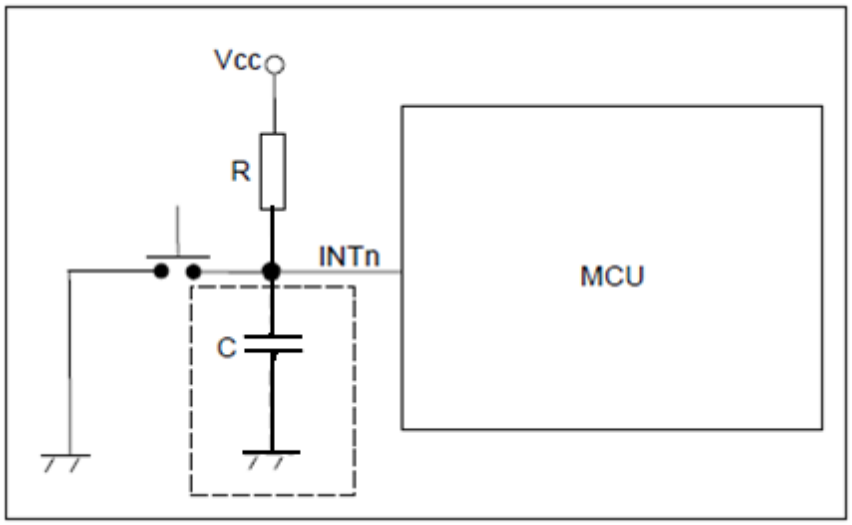
This chapter introduces the hardware circuit to design keyboard by using External Interrupt.

3.1 Hardware Modification

The hardware circuit for jittering elimination is different and needs modification.

To eliminate jittering by hardware, a capacitor is needed in the hardware circuit.

Figure 1. SOFTUNE Version



4 Resource Usage

This chapter introduces evaluation steps in normal operation mode.

For more on register setting, please refer to Chapter 15 of the MB95200 Series Hardware Manual.

4.1 External Interrupt Control Register (EICxy)

The external interrupt control register is used to control interrupt and select the edge polarity for the external interrupt input and control interrupts.

Table 1. External Interrupt Control Register

	bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Corresponding Ext. Int. No.
EIC10	EIR1	SL11	SL10	EIE1	EIR0	SL01	SL00	EIE0	INT03 INT02
EIC20	EIR1	SL11	SL10	EIE1	EIR0	SL01	SL00	EIE0	INT05 INT04
EIC30	EIR1	SL11	SL10	EIE1	EIR0	SL01	SL00	EIE0	INT07 INT06

4.2 Interrupt Level Setting Registers (ILR0 to ILR5)

Table 2. Configuration of External Interrupt Level Setting Registers

ILR0	L03	[1:0]	L02	[1:0]	L01	[1:0]	L00	[1:0]
ILR1	L07	[1:0]	L06	[1:0]	L05	[1:0]	L04	[1:0]
ILR2	L11	[1:0]	L10	[1:0]	L09	[1:0]	L08	[1:0]
ILR3	L15	[1:0]	L14	[1:0]	L13	[1:0]	L012	[1:0]
ILR4	L19	[1:0]	L18	[1:0]	L17	[1:0]	L016	[1:0]
ILR5	L23	[1:0]	L22	[1:0]	L21	[1:0]	L020	[1:0]

One of the interrupt level setting registers (ILR0) contains 4 pairs of bits assigned to interrupt requests from external interrupt.

Table 3. Interrupt Sources

Interrupt Request No.	Bit name in Interrupt Level Setting Register	Interrupt Source
IRQ0	L00[1:0]	External Interrupt ch.4
IRQ1	L01[1:0]	External Interrupt ch.5
IRQ2	L02[1:0]	External Interrupt ch.2/ ch.6
IRQ3	L03[1:0]	External Interrupt ch.3/ch.7

Each pair of bits (interrupt level setting bits as two-bit data) sets an interrupt level.

Table 4. Relationships between Interrupt Level Setting Bits and Interrupts Levels

LXX[1:0]	Interrupt Level	Priority
00	0	Highest
01	1	
10	2	
11	3	Lowest (No interrupts accepted)

XX: 00 to 03 Corresponding interrupt number

During execution of a main program, the interrupt level bits in the condition code register

(CCR: IL1, IL0) usually indicate “11B”.

5 Software Design

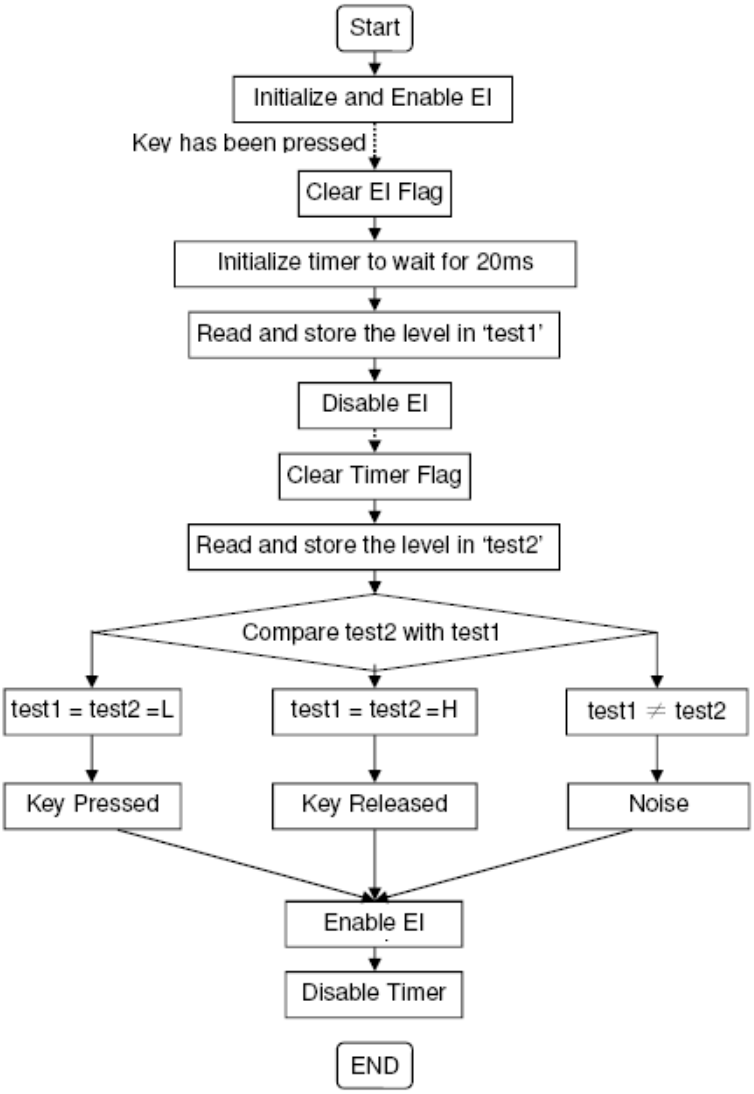
This chapter introduces the keyboard design by using external interrupt.

5.1 Software Design by Using External Interrupt

Before perform this function, first initialize the register of the external interrupt, set up interrupt levels, and then enable external interrupt. If a signal is detected, the system will activate the external interrupt function, waiting for about 20ms~100ms to see if the signal is noise.

The flow chart is as below:

Figure 2. Flow Chart of Keyboard Design by Using External Interrupt



6 Sample Code

This chapter gives an example, illustrating how to use the external interrupt to perform the keyboard function.

The following example shows how to use the external interrupt Channel 6 to perform a keyboard function. This sample code is based on MB2146-410-01. External interrupt channel 6 is used for to detect the key status. The default input level is high.

Program enters ISR while generating an Ext Int in Channel 6, clears flag, activates a timer to create a 20ms interval for the jittering elimination, then waits for the next external interrupt signal.

```
/* EI initialize*/
void InitExtInt06 (void)
{
       DDR0 &= 0xBF;        // P06 input
       EIC30 = 0x07;        // External Interrupt 6 double edge
                            // & Enable external interrupt output
}
/* Timer Interrupt routine*/
__interrupt void Timer_Interrupt (void)
{
       if (T00CR1_IE)
       {
       T00CR1_IE = 0;       // Clear timer interrupt request flag
       test02 = PDR0_P06;
       level_deal();
       EIC30_EIE0 = 1;      // Restart External Interrupt ch6
       T00CR1_OE = 0;       // Disable timer 0
       }
}
/* System clock select routine*/
void clock_Select(void) // Main clock initialize
{
       SYCC = 0x00;
       WATR = 0x00;
       STBC = 0x01;
       SYCC2= 0xF4;
       while(STBC_MRDY);
}
/* Key routine*/
void level_deal(void)
{
       if((test01==test02)&&(test02==0))              // Judge if key pressed
       {
       DDR0_P05 = 1;
       PDR0_P05 = 0;
       }
       else if((test01==test02)&&(test02==1))  // Judge if key released
       {
       DDR0_P05 = 1;
       PDR0_P05 = 1;
       }
}
```

```
/* EI interrupt routine*/
__interrupt void ExInt06 (void)
{
       if (EIC30_EIR0)
       {
       EIC30_EIR0 = 0;            // Clear interrupt request flag
                                  // interrupt server routine
       EIC30_EIE0 = 0;
       T01DR = 0x4E;              // set count value (16 bit, 20ms)
       T00DR = 0x14;
       TMCR0 = 0x10;              // 16-bit, no filtering
       T00CR0 = 0x80;             // Interval timer with single mode
       T00CR1 = 0xA0;             // Start timer
       test01 = PDR0_P06;         // Read EI level
       }
}
/* Main Loop */
void main(void)
{
       clock_Select();            // System Clock select routine
       InitExtInt06();            // External interrupt routine
       InitIrqLevels();
       __EI();
       while(1);
}
```

Please note that the corresponding interrupt vector and level shall be defined in the vector.c module of Cypress's standard template project.

```
...
void InitIrqLevels(void)
{
       ILR0 = 0xCF; //IRQ2: external interrupt ch6
       ILR1 = 0xF3; //IRQ5: 8/16-bit timer ch0 (lower)
       ...
}
__interrupt void ExInt06 (void);     // Prototype
__interrupt void Timer_Interrupt(void);
...
#pragma intvect ExInt06 2           // IRQ2: external interrupt ch6
#pragma intvect Timer_Interrupt 5   // IRQ5: 8/16-bit timer ch0 (lower)
```

Refer to Appendix Sample Code for project "EI_Basic".

7 Performance Evaluation

As stated before, there are two methods to eliminate jittering, hardware and software.

Encoding for the hardware method is very simple, only using the external interrupt, but the jittering elimination process is not good since the system may not be able to scan the edge.

By comparison, the software method is more useful.

8 Additional Information

For more Information on Cypress MB95200 products, visit the following website:

http://www.cypress.com/8fx-mb95200

9 Document History

Document Title: AN205506 - F²MC-8FX Family, MB95200 Series Keyboard development using external interrupt

Document Number: 002-05506

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	HUAL	02/16/2009	Initial release
*A	5264388	HUAL	05/10/2016	Migrated Spansion Application Note MCU-AN-500036-E-10 to Cypress format

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

cypress.com/psoc

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

Cypress Developer Community

Community | Forums |Blogs | Video |Training

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, 2009-2016. 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.