AN205571

F²MC-8FX Family, MB95310/370 Series I2C Hardware Design API

This Application note introduces API for I2C hardware design. This is two wires synchronization protocol in device communication, which has bi-directional wires SDA and SCL. In following chapters we will describe the I2C hardware design and protocol.

1 Introduction

This document introduces API for I2C hardware design.

I2C is two wires synchronization protocol in device communication, which has bi-directional wires SDA and SCL. In following chapters we will describe the I2C hardware design and protocol.

2 Background

This chapter introduces the background of I2C.

Compared with others serial extended interface protocol, the I2C is used widely. I2C bus has standard criterion and multitudinous periphery apparatus.

The I2C interface is a two-wire, bi-directional bus consisting of a serial data line (SDA) and a serial clock line (SCL). The devices connected to the bus via these two wires can exchange data, and each device can operate as a sender or receiver in accordance with their respective functions based on the unique address assigned to each device.

Following are some features of I2C:

- integrated Criterion
- independent Structure
- Easy to use

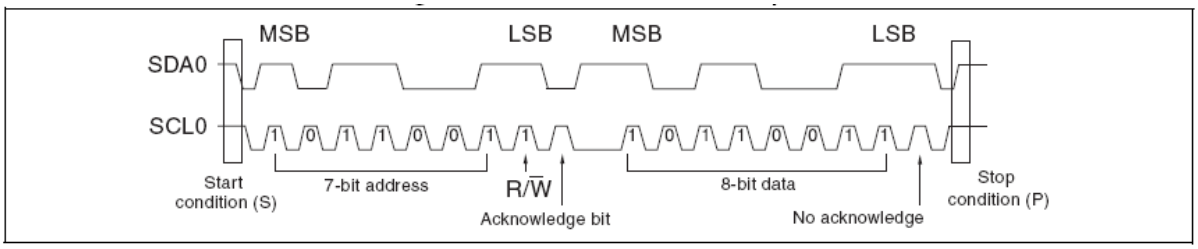
When I2C is used, typical use module of SCM is a small system. It can include I/O, keyboard, display module, clock, A/D, data storage and so on. For what the software become easy and content become more abundance.

3 Description of I2C Protocol

This chapter describes protocol of I2C.

In I2C bus, SCL generates clock and SDA reloads data out and in. Figure 1 simply describes I2C protocol.

Figure 1. I2C Protocol



The slave address is transmitted after a start condition (S) is generated.

Data is transmitted after the address. Data can be transmitted continuously to the same slave address.

The data is eight bits followed by an acknowledgment.

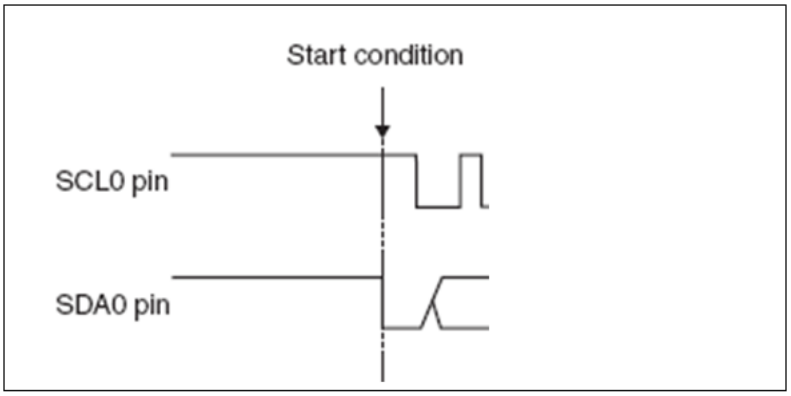
Data transfer is always ended in the master stop condition (P). However, the repeated start condition can be used to transmit the address which indicates a different slave without generating a stop condition.

3.1 Start Condition

When bus is free, change SDA from high to low while SCL is high will generate a start signal.

In MB95F310 writing “1” to register IBCR10_MSS and ICCR0_EN will generate a start signal. Detailed start status refer to Figure 2.

Figure 2. Status of Starting



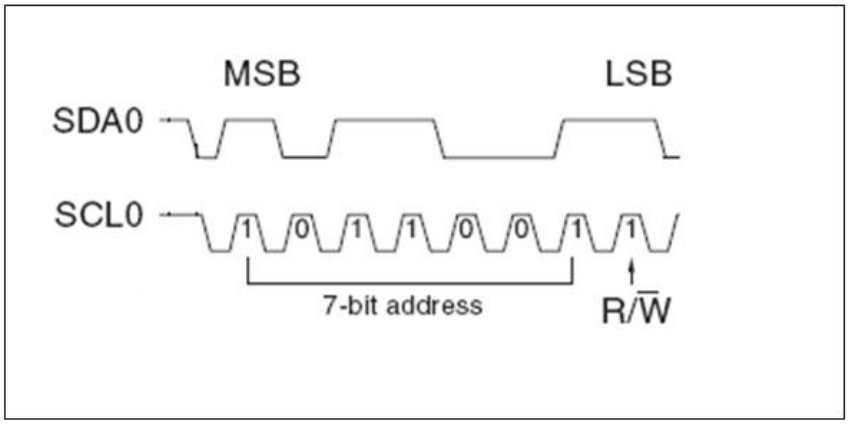
Start signal can generate again when writing “1” to MCU register IBCR10_SCC.

3.2 Address

This address is seven bits followed by the data direction bit (R/W) in the eighth bit position. About the direction, “0” is writing and “1” is reading

In MB95F310 MCU master mode slave address is written to register IDDR0, when start signal generated data of IDDR0 send out to slave device.

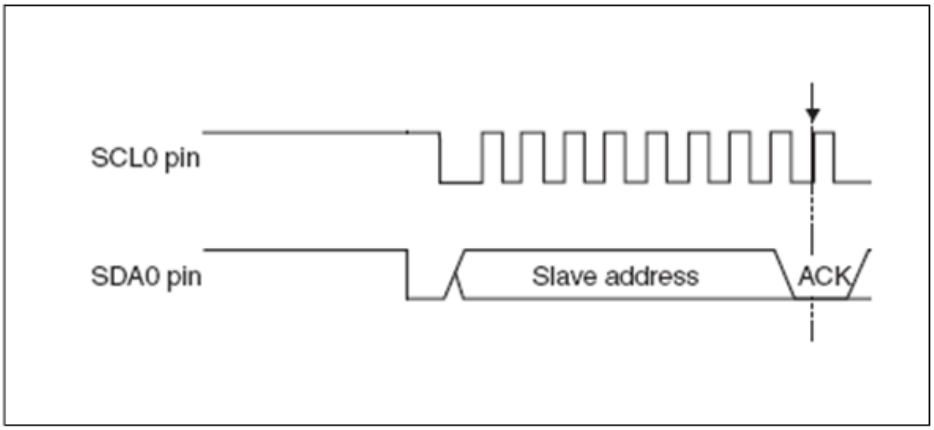
Figure 3. Status of Address



3.3 Acknowledge

Acknowledge is generated when slave device received address or data. Acknowledge is low level when SCL is in the ninth pulse.

Figure 4. Status of Acknowledge



If the SDA is high at the ninth pulse, which indicates that the slave device didn’t receive the address or data from master device.

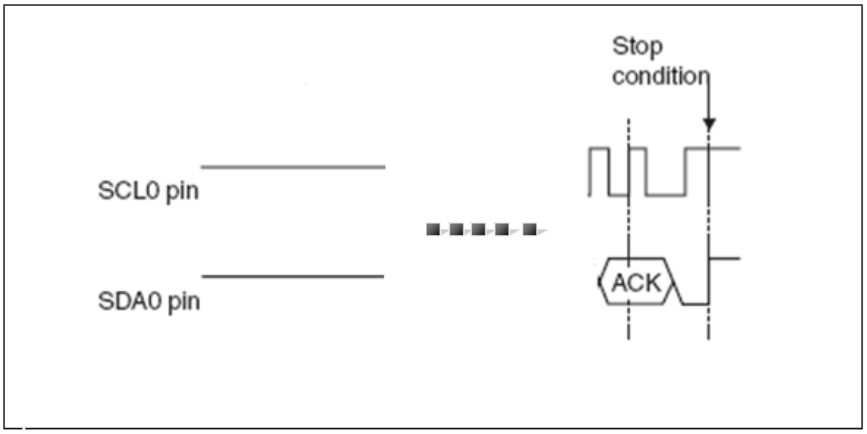
3.4 Data

Whatever writing to/reading from slave device the data is eight bits. Only when master device received the acknowledge data can transmitted out. In MB95F310 MCU the register IDDR0 is used to save the data.

3.5 Stop Condition

When SCL is high, changing SDA from low to high will generate a stop signal. In MB95F310 MCU writes “0” to register IBCR10_MSS will generate a stop signal.

Figure 5. Status of Stop



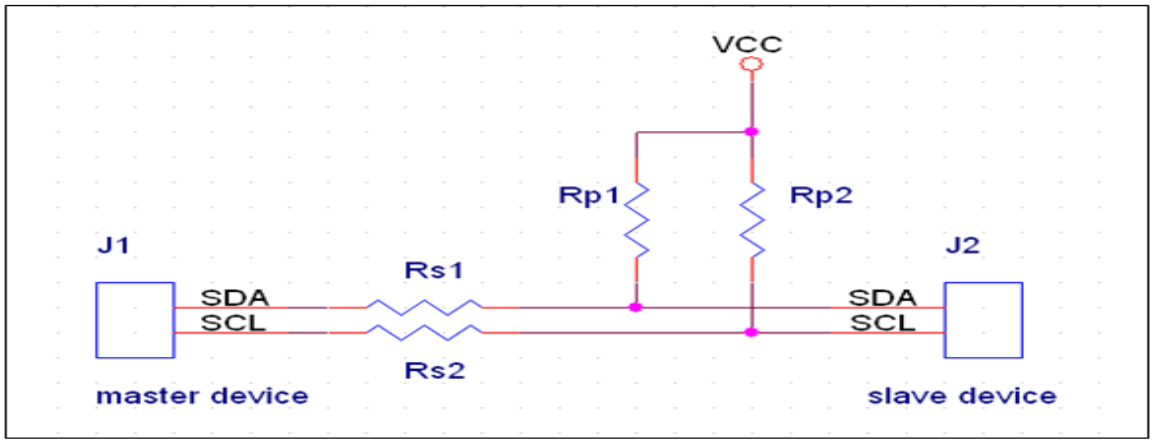
4 I2C Hardware Design

This chapter describes hardware design of I2C.

4.1 Hardware Design

Actually, I2C is two wires, which are SDA and SCL. Master SDA is connected with slave SDA, and master SCL is connected with slave SCL. Figure 6 describes the hardware connection in detailed.

Figure 6. I2C Hardware Design

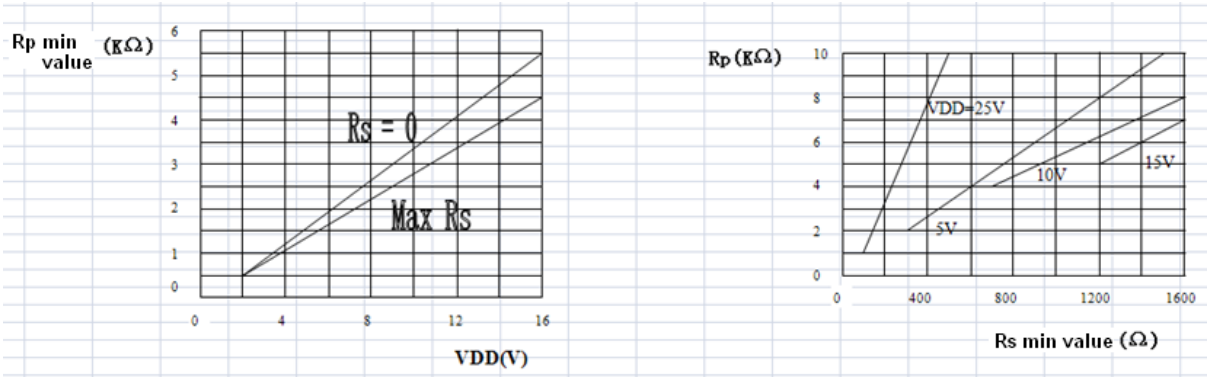


4.2 Pull Up Resister

Because the start signal is generated when SCL is high, so user must connect pull up resister to I2C.

The pull up resister is restricted by power voltage. That is because when voltage V_{OLMAX} is 0.4V, the Min current must be 3mA. Figure 7 describes the relationship between power voltage V_{DD} and pull up resister $R_{P,}$ pull up resister R_P and $R_{S.}$

Figure 7. Resister of Pull Up



4.3 I2C Electronic Character

In order to guarantee I2C transmitting from error, I2C timing is restricted strictly. Table 1 is I2C bus timing character.

Table 1. I2C Timing

Parameter	Standard Mode		High Speed Mode		Unit
	MIN	MAX	MIN	MAX	
SCL Frequency	0	100	0	400	kHz
Vacant time between first stop signal and second start signal	4.7	-	1.3	-	μs
Start holding time	4	-	0.6	-	μs
SCL cycle of low level	4.7	-	1.3	-	μs
SCL cycle of high level	4	-	0.6	-	μs
Building time of double start signal	4.7	-	0.6	-	μs
Data building time	250	-	100	-	ns
Rising time of SDA and SCL	-	1000	20	300	ns
Dropping time of SDA and SCL	-	300	20	300	ns
Building time of stop signal	4	-	0.6	-	μs
Reload Capacity of every transmitting	-	400	-	400	pF

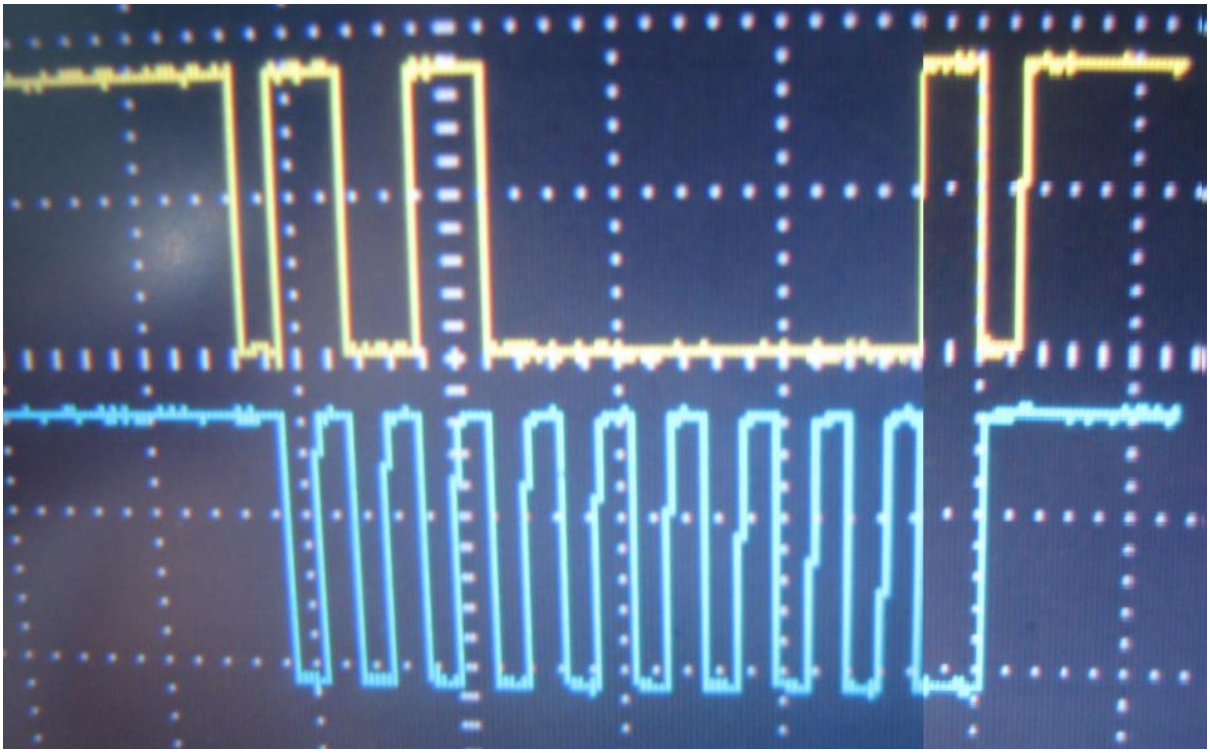
About the speed of I2C data transmission it is decided by SCL high level voltage and low level voltage. In actually the transmitting speed can be modified by adding delay for SCL.

5 Usage Demo

This chapter describes the I2C wave and I2C multi connection.

In I2C transmitting the signal is transferred synchronously. Figure 8 shows the status when bit transferring.

Figure 8. I2C Wave



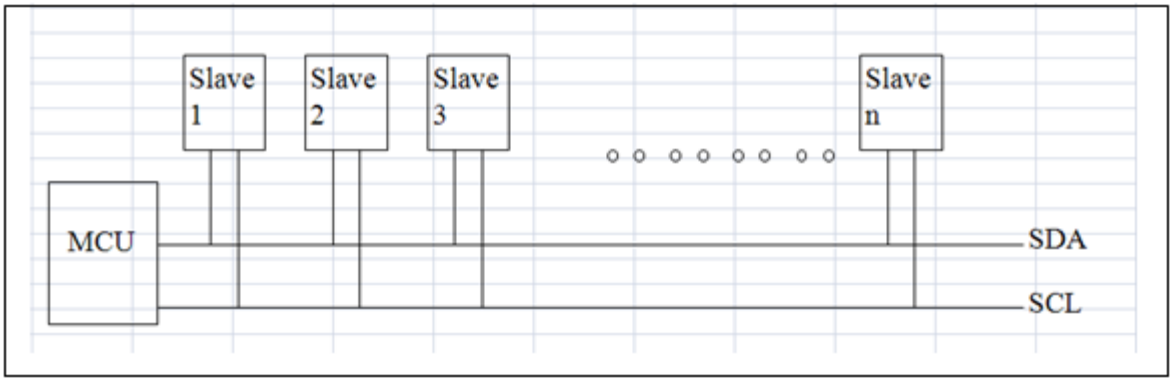
The yellow wave is SDA which transfers data. The blue wave is SCL which transfers clock.

This wave displays the start condition, data transfer condition, acknowledge generate condition, and stop condition.

One master I2C device can control several I2C devices. Every slave device has only address. When the slave device received the address which is same as itself, the slave device generates acknowledge to master, and then begin to receive data.

Figure 9 describes the situation when one master controls several slave device.

Figure 9. I2C Multi-connection



6 Additional Information

For more Information on MB95310/370 series I2C Hardware Design, visit the following website:

www.cypress.com/documentation/application-notes/mb95310370-i2c-hardware-design-based-mb95310370

7 Document History

Document Title: AN205571 - F²MC-8FX Family, MB95310/370 Series I2C Hardware Design API

Document Number: 002-05571

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	AMYC	11/10/2009	Initial release
*A	5292256	AMYC	10/14/2016	Migrated Spansion Application Note MCU-AN- 500050-E-10 to Cypress format
*B	5832465	AESATMP9	07/25/2017	Updated logo and copyright.

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
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

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

© Cypress Semiconductor Corporation, 2009-2017. 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, WICED, 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.