AN205445

F²MC-8FX Family MB95200 Series 8-Bit Microcontroller ADC API

Associated Part Family: MB95200 Series

This document introduces API for A/D conversion.

1 Introduction

This document introduces API for A/D conversion.

We should set up three functions to control A/D conversion: initialize A/D conversion, set the state of A/D conversion and read the A/D conversion result.

2 A/D Conversion Library Function List

This section introduces all functions in A/D conversion library.

Table 1 lists the Clock library functions.

Table 1. A/D Conversion Functions

Function name	Description
void AD_Init(uchar Trigger,uchar Precision, uchar System_Clock,uchar Samp_Time, Channel)	Initialize ADC includes A/D Trigger mode, Channel, system clock, sample time, Precision
void AD_State(uchar State)	Setting A/D start or stop
uchar AD_read(void)	Read the value of the ADC

3 A/D Conversion Function Detail

This section introduces the detail of ADC function.

3.1 AD_Init Function

Table 2 describes AD_Init function.

Table 2. AD_Init Function

Function name	AD_Init
Function prototype	void AD_Init(uchar Trigger,Precision,uchar System_ Clock,uchar Samp_Time,uchar Channel)
Behavior description	Initialize the AD conversion
Input parameter1	Trigger, Select the AD Trigger mode
Input parameter2	Precision, Setting the conversion precision
Input parameter3	System_Clock, Select the system clock
Input parameter4	Samp_Time, Select the AD sample time
Input parameter5	Channel, Select the Analog input channel
Return value	None
Example	Set AD as 8-bit precision, Trigger by AD start bit, 2MCLK, samp_time 16×CLK, AN00 as input: AD_Init(Trigger_AD,Precision_8,MCLK_8,CKIN_16,AN05);

This function is to initialize the A/D conversion. The user could use this parameter to choose A/D sampling channel, select the system clock, sample time, precision, trigger type, and continues mode.

Table 3 describes the Trigger parameter values.

Table 3. Trigger Definition

Trigger	Description
Trigger_AD	Trigger by A/D bit
Trigger_TO00	Trigger by TO00

Table 4 describes the Precision parameter values.

Table 4. Precision Definition

Precision	Description
Precision_8	8 bit
Precision_10	10 bit

Table 5 describes the System_Clock parameter values.

Table 5. System_Clock Definition

System_Clock	Description
MCLK_1	1×MCLK
MCLK_2	2×MCLK
MCLK_4	4 ×MCLK
MCLK_8	8×MCLK

Table 6 describes the Samp_Time parameter values.

Table 6. Samp_Time Definition

Samp_Time	Description
CKIN_4	Clk×4
CKIN_7	Clk×7
CKIN_10	Clk×10
CKIN_16	Clk×16

Table 7 describes the Channel parameter values.

Table 7. Channel Definition

Channel	Description
AN00	AN00 pin
AN01	AN01 pin
AN02	AN02 pin
AN03	AN03 pin
AN04	AN04 pin
AN05	AN05 pin

3.2 AD_State Function

Table 8 describes AD_State function.

Table 8. AD_State Function

Function prototype	void AD_State (uchar State)
Behavior description	select the A/D conversion start or not
Input parameter	State, setting the A/D conversion state
Return value	None
Example	Start A/D conversion AD_State (AD_Start);

Table 9 describes the State parameter values.

Table 9. State Definition

State	Description
AD_Stop	A/D stop
AD_Start	A/D start

3.3 AD_Read Function

Table 10 describes AD_Read function.

Table 10. AD_Read Function

Function name	AD_Read
Function prototype	uchar AD_read (void)
Behavior description	Read the A/D conversion value from ADDH, ADDL
Input parameter	None
Return value	A/D conversion value from ADDH, ADDL
Example	Read A/D result: [variable]=AD_Read();//[variable]save AD result

In this routine, we should set up three functions to control A/D conversion. Select A/D start or not, initialize the A/D (include the channel, system clock, sample time, trigger mode, precision, and continuous), and read the value from the register (ADDH.ADDL).

4 Usage Demo

This section describes something we must pay attention to when we use.

4.1 How to Judge Whether A/D conversion is Finished

If A/D conversion has finished, there is two ways to judge, one is to check the A/D conversion bit ADC1 bit2 (ADMV) if 1 the conversion is not finished, and the other one is to check the interrupt flag bit ADC1 bit3 (ADI) if 1 the conversion finished. The two methods all finished automatic in AD_Read () module.

5 Additional Information

For more information about how to use MB95200H EV-board, BGM Adaptor and SOFTUNE, please refer to SKT MB2146-410-01-E User Manual, or visit websites:

http://www.cypress.com/documentation/software-and-drivers/f2mc-8fx-mb95200h210h-series-starter-kit-mb2146-410a-01-e-setup

Document History

Document Title: AN205445 – F²MC-8FX Family MB95200 Series 8-Bit Microcontroller ADC API

Document Number: 002-05445

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	HUAL	03/04/2009	Initial release
*A	5279101	HUAL	05/20/2016	Migrated Spansion Application Note MCU-AN- 500025-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

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, 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.