AN205390

F²MC - 8FX Family, MB95200 Series, How to make on-Board Debug

Debugging on the target board is a very important step in project design. This document will describe how to debug on a target board. The debugging interface described in this document can be used both as the in-circuit debugging tool and the in-circuit programming tool.

Contents

1 Introduction....1
2 Application Environment....2
2.1 Debug Tool....2
2.2 SOFTUNE....2
3 MCU Products....3
4 Hardware Design....4
4.1 Single Flash MCU Debugging Circuit....4
4.2 Dual Flash MCU Debugging Circuit....5
5 Debugging Operation....7
5.1 Preparation before using F²MC-8L/8FX SOFTUNE....7
5.2 Debugging with F²MC-8L/8FX SOFTUNE....8
6 Additional Information....12
Document History....13

1 Introduction

Debugging on the target board is a very important step in project design. This document will describe how to debug on a target board.

The debugging interface described in this document can be used both as the in-circuit debugging tool and the in-circuit programming tool.

2 Application Environment

This chapter introduces application environment for MB95200H/210H SOP20 PGM adaptor.

2.1 Debug Tool

The debugging tool is BGMA (BGM Adaptor) and its model No. is MB2146-08-E, as shown in the picture below. It is available in the MB95200 MCU Starter Kit (PN: MB2146-410A-01-E).

Figure 1. BGM Adaptor

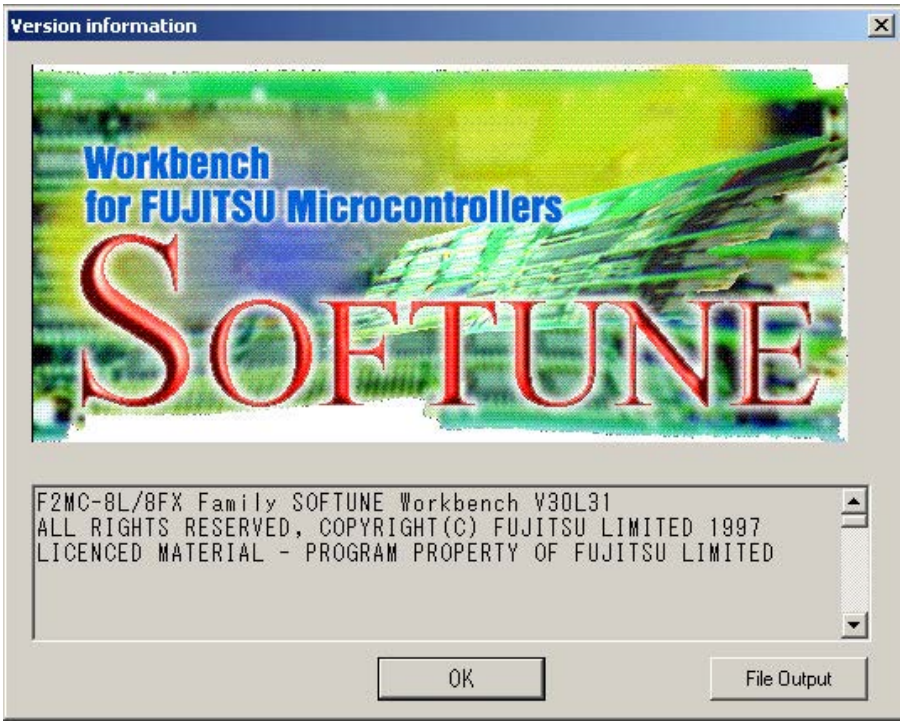


2.2 SOFTUNE

SOFTUNE is used as software development environment for programming and debugging.

The version currently being used is F^{2}MC-8L/8FX SOFTUNE Workbench V30L31, as shown below in the picture. It is available in the MB95200 MCU Starter Kit (PN: MB2146-410-01-E)

Figure 2. SOFTUNE Version



3 MCU Products

This chapter introduces different models of MCU products.

As the in-circuit debugging interface in the single flash MCU is different from that in the dual flash MCU, MCUs are classified into two categories in the list below.

Table 1. MCU Products

Series	Flash type	Chip list	Series	Flash type	Chip list
MB95F200H	Single Flash	MB95F202K	MB95F260H	Dual Flash	MB95F262K
		MB95F202H			MB95F262H
		MB95F203K			MB95F263K
		MB95F203H			MB95F263H
		MB95F204K			MB95F264K
		MB95F204H			MB95F264H
MB95F210H	Single Flash	MB95F212K	MB95F270H	Dual Flash	MB95F272K
		MB95F212H			MB95F272H
		MB95F213K			MB95F273K
		MB95F213H			MB95F273H
		MB95F214K			MB95F274K
		MB95F214H			MB95F274H
MB95F220H	Single Flash	MB95F222K	MB95F280H	Dual Flash	MB95F282K
		MB95F222H			MB95F282H
		MB95F223K			MB95F283K
		MB95F223H			MB95F283H
					MB95F284K
					MB95F284H
			MB95F330H	Dual Flash	MB95F332K
					MB95F332H
					MB95F333K
					MB95F333H
					MB95F334K
					MB95F334H

4 Hardware Design

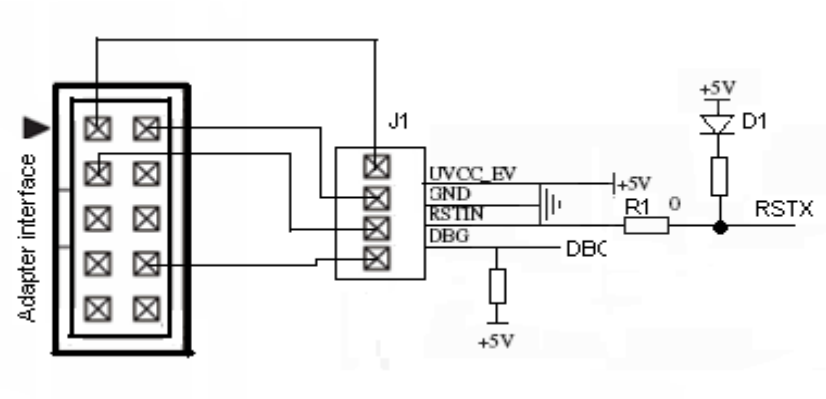
This chapter introduces the circuit of debugging interface.

As the MB95200 series has two types of flash, i.e. single flash and dual flash, there are two types of debugging circuits corresponding to them. This chapter will introduce the two programming circuits. Make sure to choose the correct circuit in application.

4.1 Single Flash MCU Debugging Circuit

Figure 3 shows the circuit diagram of the in-system debugging interface. To design the in-system debugging interface, four pins, UVCC_EV, GND, RSTIN and DBG, are needed. (Add pin RST_OUT if necessary).

Figure 3. Basic Connector Circuit for Single Flash MCU



Component Recommendation:

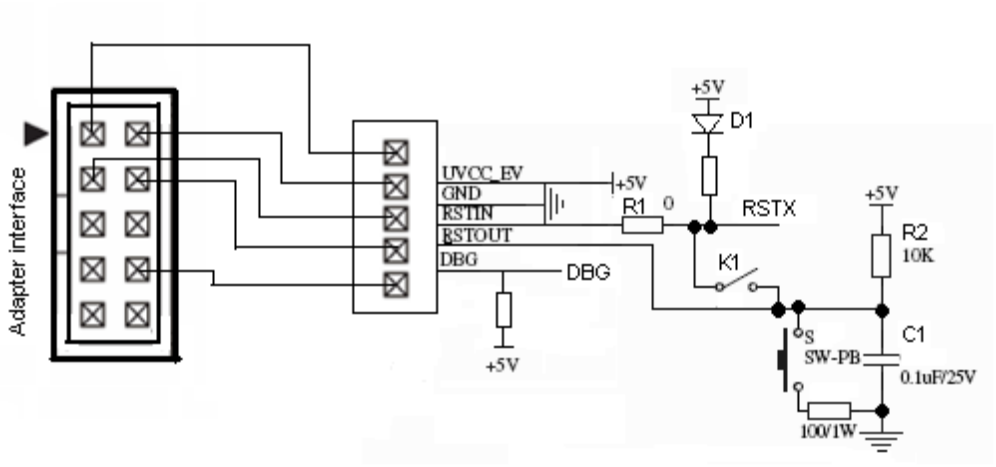
D1: VF < 0.3V when IF = 1mA. E.g. LL103A, 1SS294

Below is a list of functions of the interface connector.

1. J1 is a 10-pin connector, used as the circuit interface.
2. The write voltage (VCC = 4.5V to 5.5V) is supplied from the user system. The UVCC_EV and DBG pins control the PGM mode entry.
3. Pin 2 of the connector connects to the GND.
4. The debugger provides 10 V directly to the RSTX pin during flash erase/write operation. If it is pulled high in the user system, please consider to add a low-drop diode for separate H voltage.
5. The DBG pin provides 1-line UART communication with the debugger. Serial write mode can be set by providing special timing of DBG and VCC pins.

If the MCU target needs a reset circuit, please design the reset circuit according to the diagram below.

Figure 4. Debugging Connector and Reset Circuit for Single Flash MCU

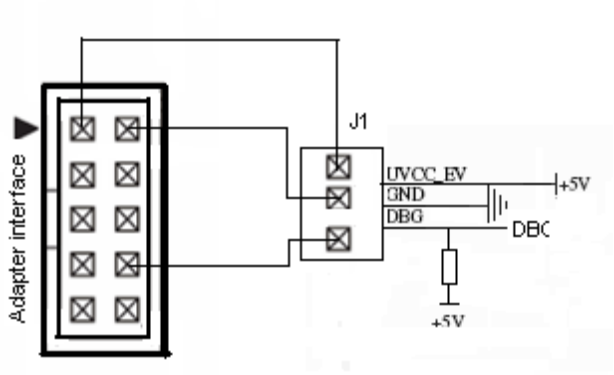


Pin 4 of the connector is used to connect to the reset circuit. Please design a switch K1 between the reset circuit and the reset pin (RSTX). Make sure to cut off K1 to stop the high voltage flowing through the reset circuit during the debugging operation. Turn on K1 to supply the reset circuit to the target MCU when the debugging operation is over.

4.2 Dual Flash MCU Debugging Circuit

Figure 5 shows the circuit diagram of the in-system debugging interface. To design the in-system debugging interface, three pins, UVCC_EV, GND and DBG, are needed. (Add pin RST_OUT if necessary.)

Figure 5. Dual Flash MCU Debugging Circuit



Component Recommendation:

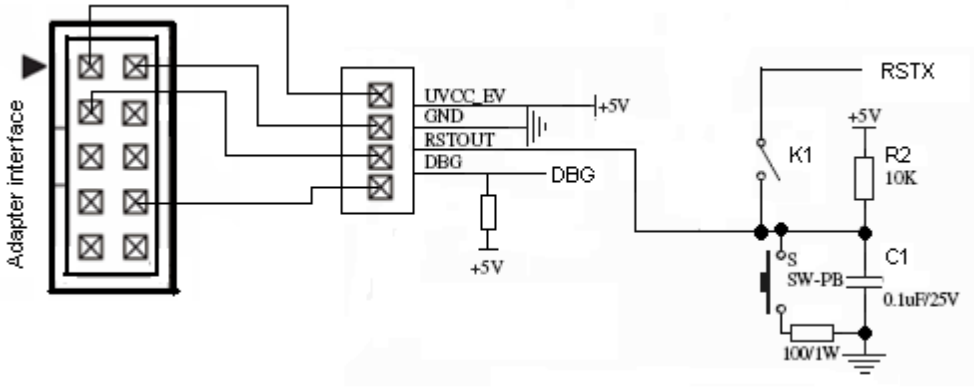
D1: VF < 0.3V when IF = 1mA. E.g. LL103A, 1SS294

Below is a list of functions of the interface connector.

1. J1 is a 10-pin connector, used as the interface of this circuit.
2. The write voltage (VCC = 4.5V to 5.5V) is supplied from the user system. The UVCC_EV and DBG pins control the PGM mode entry.
3. Pin 2 of the connector connects to the GND.
4. The DBG pin provides 1-line UART communication with the debugger. Serial write mode can be set by providing special timing of DBG and VCC pins.

If the MCU target need a reset circuit, and please design a reset circuit as follow circuit.

Figure 6. In-system Debugger Interface



Pin 4 of the connector is used to connect to the reset circuit. Design a switch K1 between the reset circuit and the reset pin (RSTX). When debugging operation is over, K1 can be switched on to supply the reset circuit to the target MCU.

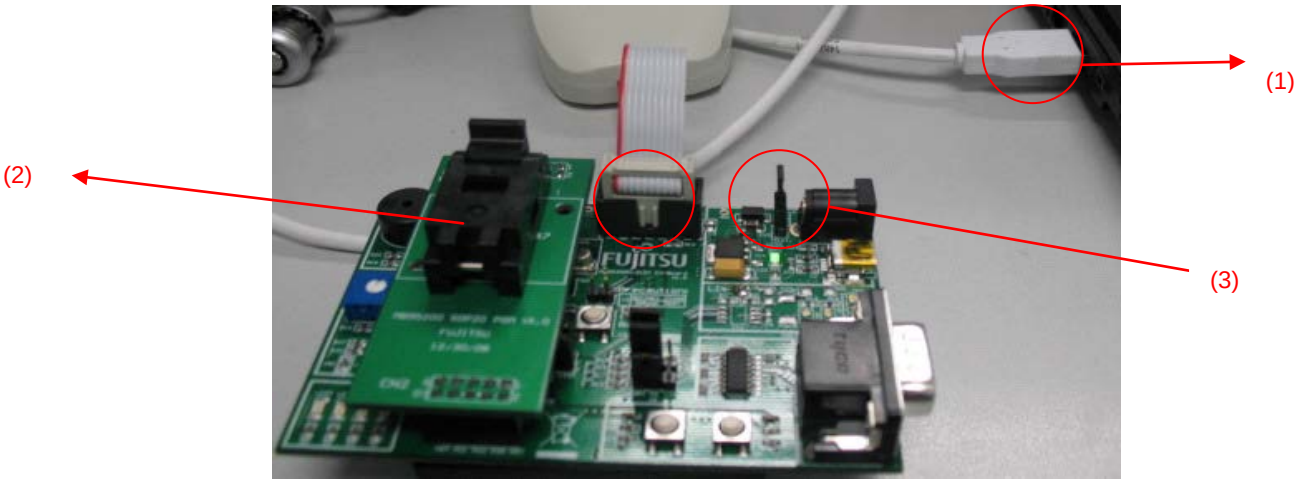
5 Debugging Operation

This chapter introduces debugging operation by F²MC-8L/8FX SOFTUNE.

5.1 Preparation before using F²MC-8L/8FX SOFTUNE

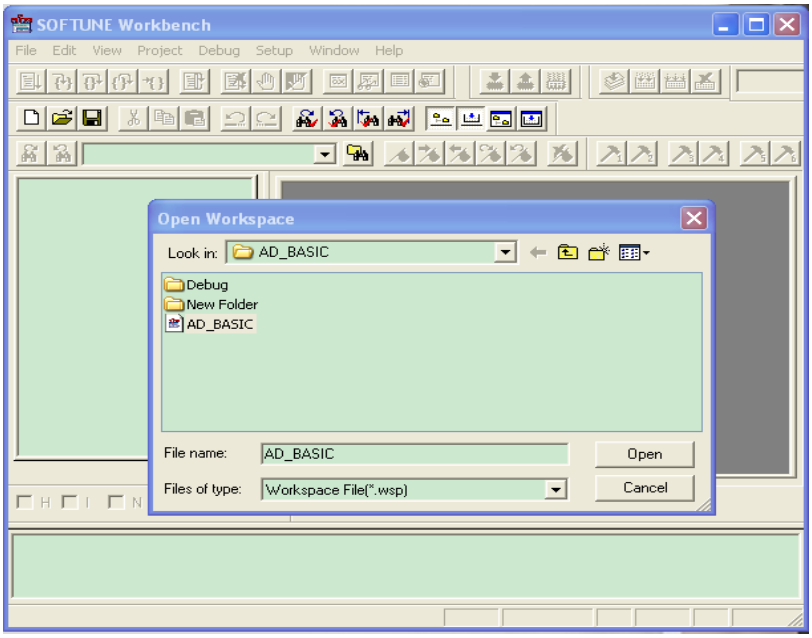
1. Connect the BGMA to a PC.
2. Connect the EV-board to the BGMA (Note that user can only power on the MCU after connecting the target board to the BGM adaptor).
3. Power on the EV-board.

Figure 7. Step 1-3



4. Open a project (E.g. IO_LED) using SOFTUNE.

Figure 8. Open the Demo Project

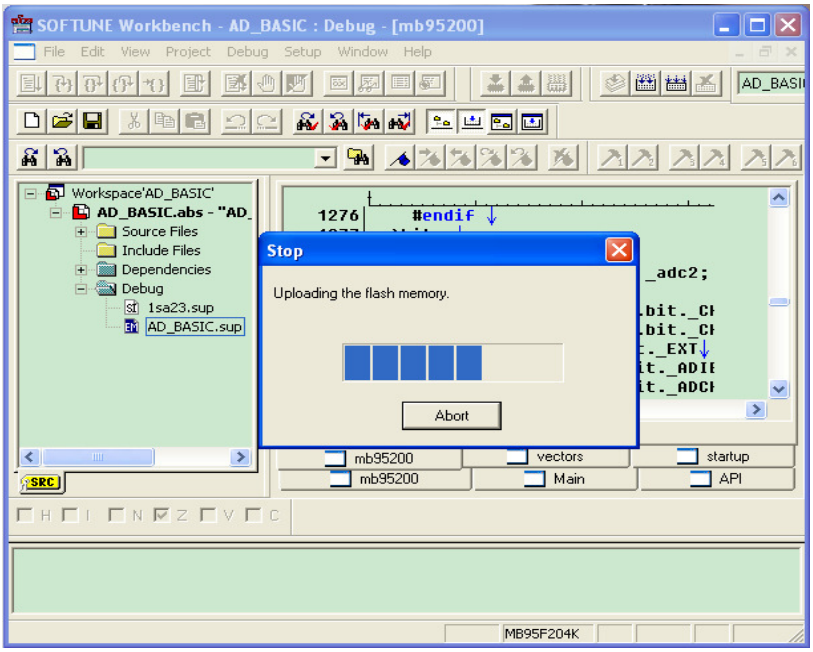


5.2 Debugging with F²MC-8L/8FX SOFTUNE

After the 3-step preparation introduced in Chapter 5.1, user can start debugging as shown below.

1. Start debugging. (by clicking **Start Debug** in **Debug**)

Figure 9. Start Debugging



2. Start Debugging (Code Update)

   Select **Debug** -> **Run** to start debugging or use the shortcut key as indicated in Figure 11.

Figure 10. Run Operation

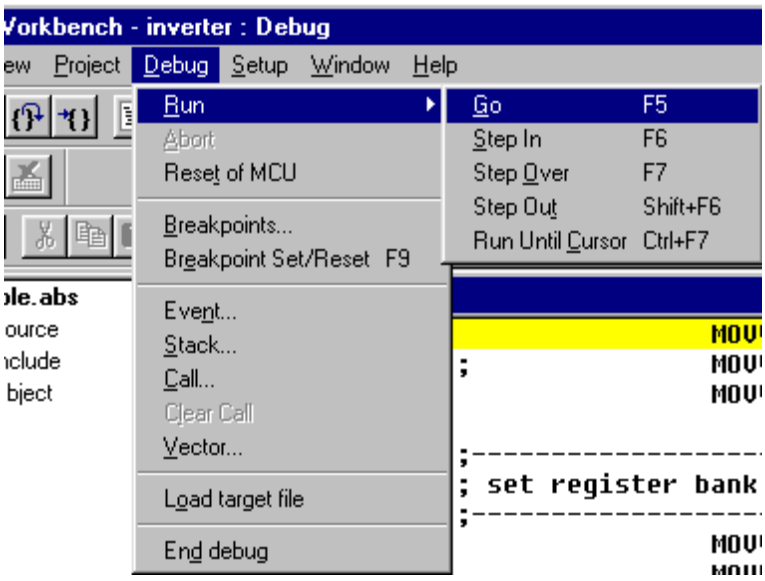
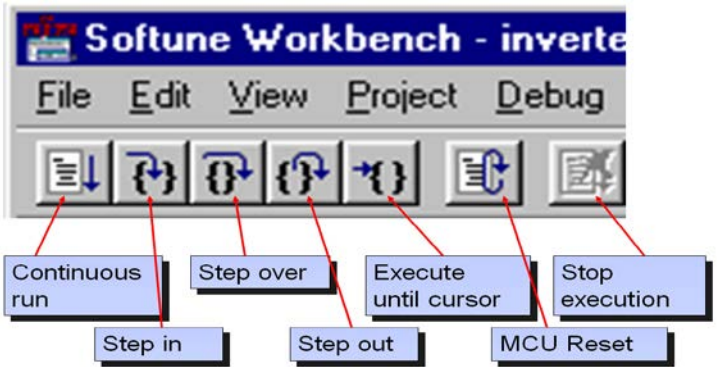


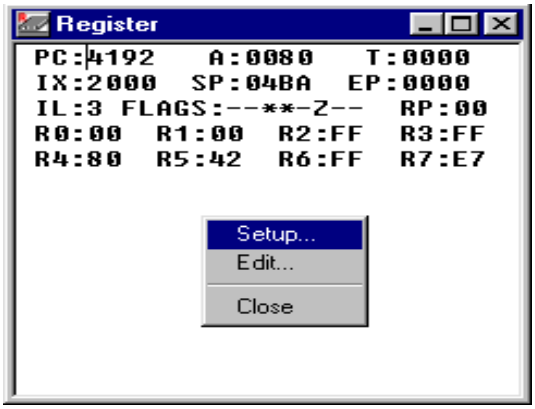
Figure 11. Shortcut Keys



3. Register Operation

Select **View** -> **Register** to open the **Register** window. Then right click, select **Setup** from shortcut menu to add/delete registers in the register window. Select **Edit** -> **Register** from the pull-down list, input the value in **Register value**, and Click **OK** to store the new value to the register.

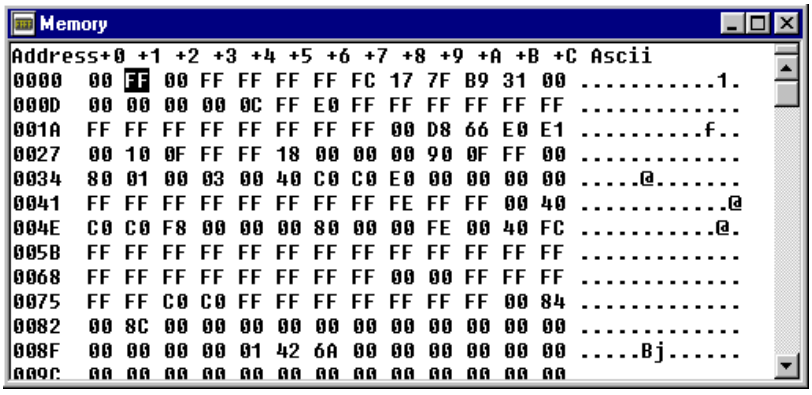
Figure 12. Register Operation



4. Memory Operation

Then select View -> Memory to display the Memory window, Address content can be modified by highlight the address and then type the new value.

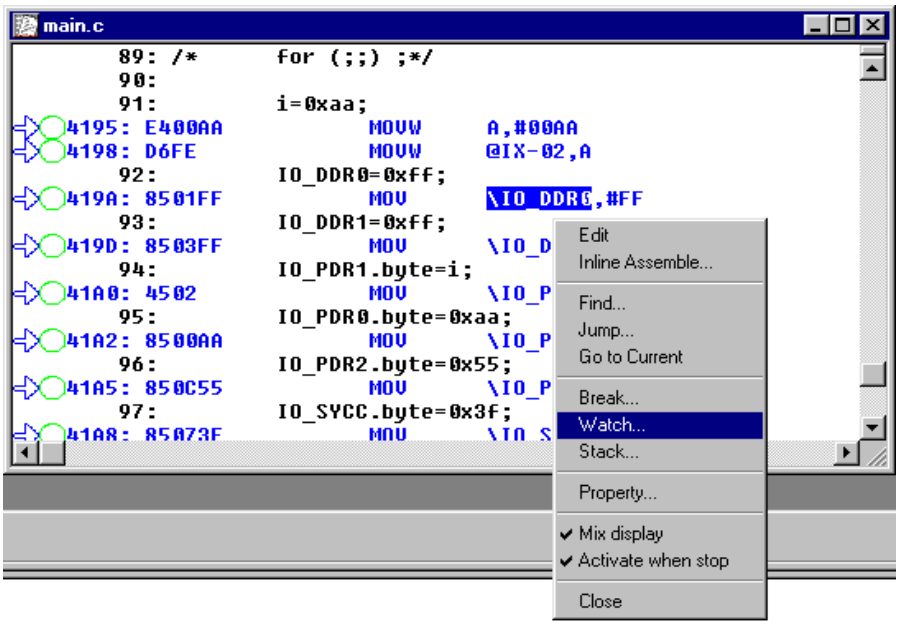
Figure 13. Memory Operation



5. Watch Operation

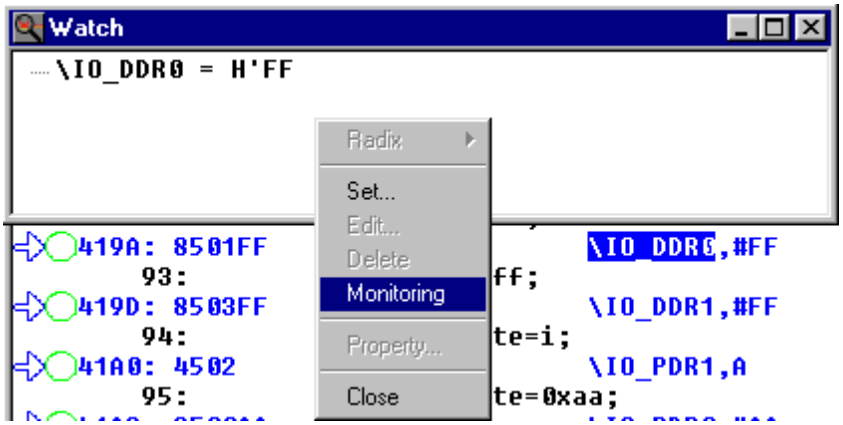
Highlight the symbol to be watched. Right click and select **Watch** in the shortcut menu.

Figure 14. Select the Watch Window



Put the mouse pointer inside the window. Right click and select **Monitoring** from shortcut menu to monitor symbols continuously.

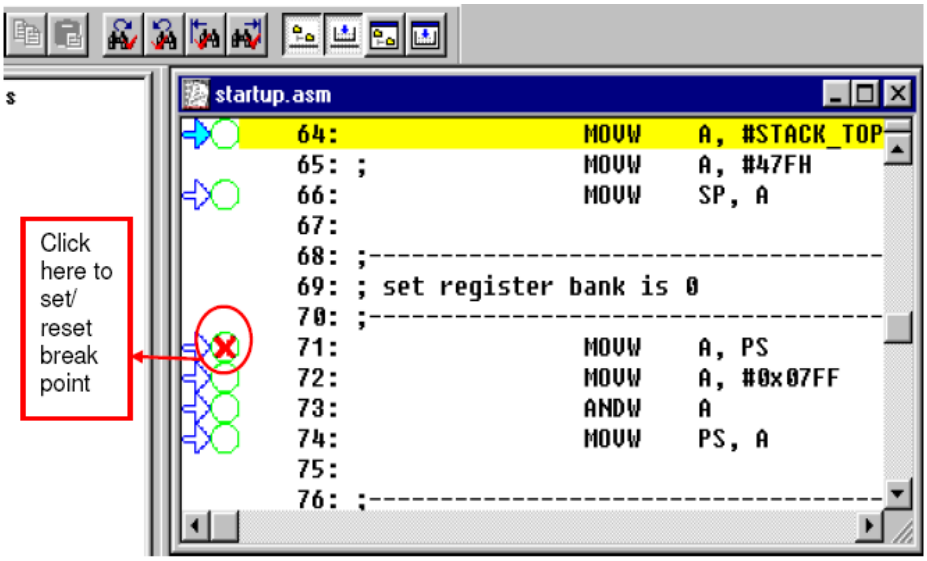
Figure 15. Continuously Monitoring



6. Breakpoints Setting

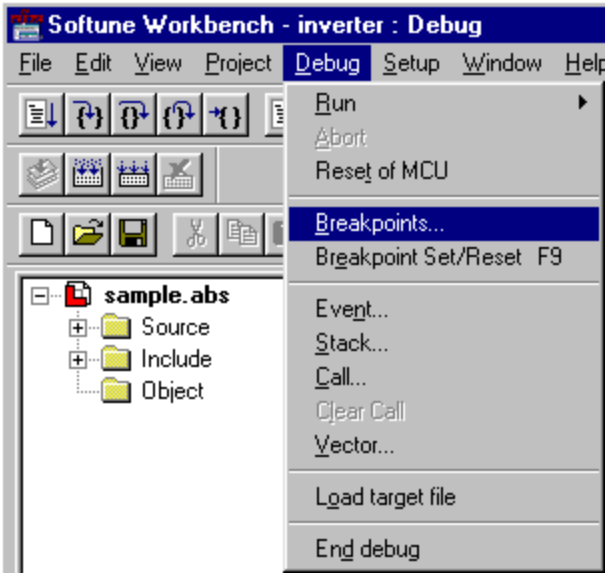
Set breakpoints by clicking the green circle in the code, as shown in the figure below.

Figure 16. Breakpoints Setting Method 1



Another way to set breakpoints is specifying the program address or data address by selecting Debug -> Breakpoints.

Figure 17. Breakpoints Setting Method 2



6 Additional Information

For more information on Cypress Microcontrollers Products, please visit the following websites:

http://www.cypress.com/cypress-microcontrollers

http://www.cypress.com/cypress-mcu-product-softwareexamples

Document History

Document Title: AN205390 – F²MC - 8FX Family, MB95200 Series, How to make on-Board Debug

Document Number: 002-05390

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	Benjamin. Yang	02/09/2009	V1.0, First Draft
			02/13/2009	Modify
*A	5267684	HUAL	05/11/2016	Migrated Spansion Application Note “MCU-AN-500018-E-11” 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.