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.

AN205257

FR, MB91460, Emulation Ram Extension Board Usage

There are several MCU of MB91460 series coming along with external bus interface. The intention of the EMA-MB91V460A-200/-210 board is to emulate external memory (Flash or RAM) until the final target hardware including the external memory is available.

Contents

1 Introduction....1
2 Integration in Softune Workbench....1
2.1 Integration steps....1
2.2 Hardware setup....2
2.3 Softune Workbench Project Preparation....3
3 Appendix....6
3.1 eme_cs.prc file....6
3.2 Related Documentation....10
Document History....11

1 Introduction

There are several MCU of MB91460 series coming along with external bus interface.

The intention of the EMA-MB91V460A-200/-210 board is to emulate external memory (Flash or RAM) until the final target hardware including the external memory is available.

This Application Note describes the usage of the EMA-MB91V460A-200/-210 board with the Softune Workbench IDE.

2 Integration in Softune Workbench

This chapter explains the necessary steps to activate the RAM extension board in Softune Workbench projects.

2.1 Integration steps

1. Setup EMA-xxx boards according to the Users Manual

Note:

Ensure that S301-2 is set to ‘OFF’ on EMA-MB91V460A-00x board.

2.1.1 Using procedure file

1. Copy the eme_csx.prc file into the ‘prc’ project folder. The file can be downloaded from the website
2. Setup used CS for final external memory, e.g. in start91460.asm
3. Start debug session and activate external memory emulation in ‘Setup" - "Debug environment’"
4. Set breakpoint in main() and execute until main()
5. Execute eme_csx.prc via command window to activate RAM extension board memory
6. Debug the application

2.1.2 Insert Memory settings into source Code

1. Setup used CS for emulation RAM, e.g. in start91460.asm
2. Debug the application

2.2 Hardware setup

To setup the EMA-MB91V460A-200/-210 together with the EMA-MB91V460A-00x or EMA-MB91FV460B-00x refer to the User's Manual of these boards.

Ensure that the correct memory size and access width is selected at the EMA-MB91V460A-200/-210 board.

Note:

Ensure that S301-2 is set to ‘OFF’ on EMA-MB91V460A-00x board.

2.3 Softune Workbench Project Preparation

2.3.1 Procedure file: eme_csx.prc

Copy the procedure file eme_csx.prc into the 'PRC' folder of your Softune Workbench workspace / project. This file is required in the debug session.

As the finally used memory might have different access times, the setup of the CS for the EMA-MB91V460A-200/-210 memory is done via the procedure file.

After the bus interface is initialised in the application (e.g. after execution of the start91460.asm) the procedure file needs to be executed.

2.3.2 Bus Interface settings in start91460.asm

The memory on the EMA-MB91V460A-200/-210 board shall replace the final memory. Using the procedure file, the settings for the CS can be set according to the final memory.

Using FME initialisation file, start9140.asm, the external bus interface is setup.

In case the settings for the final memory are unclear following settings need to be done in start91460.asm in order of correct prc file execution:

Chapter 4.8 External Bus Interface in start91460.asm:

- Enable bus interface
  - #set EXTBUS ON ; <<< Ext. Bus on/off
- Select CS usage
  - #set CS*n* ON ; <<< select CS (ON/OFF)
- Enable CS
  - #set ENACSX B'xxxxxxxx ; <<< set CS, ENACSX
- Select memory address area of CS
  - #set AREASEL*n* 0x xxxx ; <<< set start add. for CS*n*, ASR*n*

Wait states and CS configuration can already be configured for the final memory. For the EMA-MB91V460A-200/-210 memory these registers are configured within the prc file.

Ensure that all required Port Function Registers (PFRx) are set to external bus interface function.

The EMA-MB01V460A-200/-210 requires following signals:

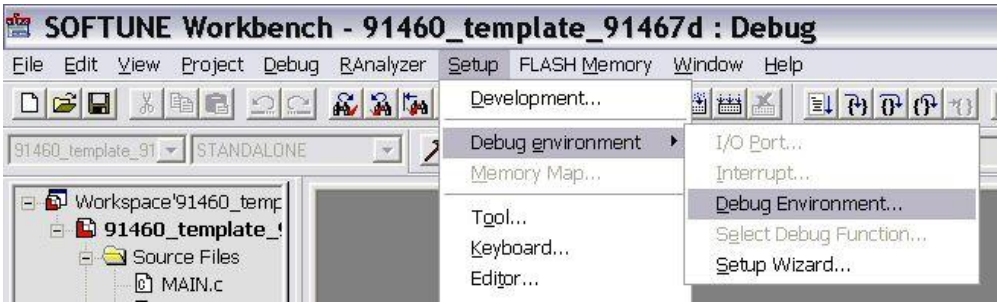
- RDX
- WEX
- D31 – Dx (depending on selected bus width)
- A0 – Ax (depending on selected address area)

In start91460.asm file these registers are set in chapter 4.8.10, 4.8.11 and 4.8.12

2.3.3 Setup Emulation RAM in Softune Workbench Debug session

Start the debug session via ‘Debug / Start Debug’. When debug session is active, select ‘Setup / Debug environment / Debug Environment’ and select tab ‘External memory emulation’.

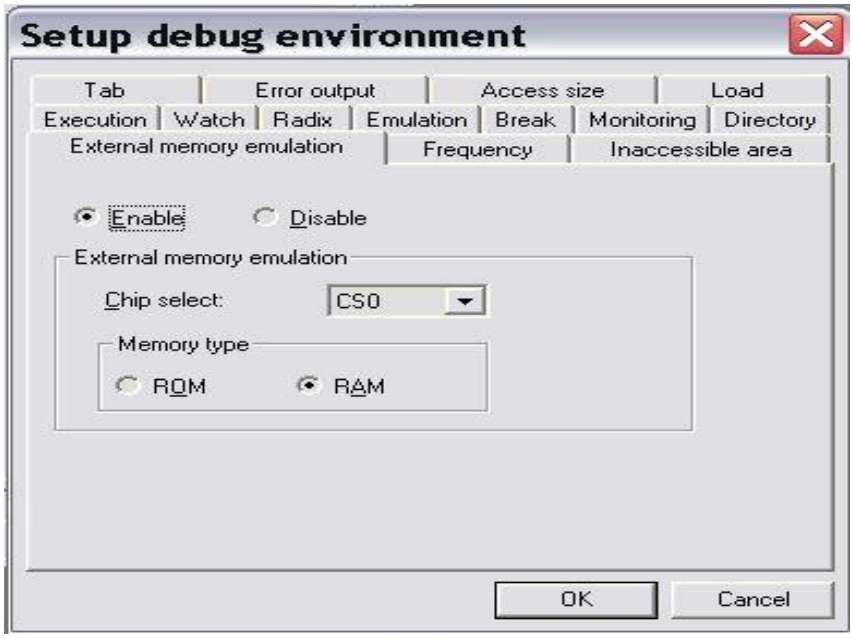
Figure 1. Setup external Ram Emulation



Enable the function using the check box ‘Enable’, select the used CS (Chip select) and memory emulation type (ROM or RAM).

Click on the ‘OK‘ button to apply the changes.

Figure 2: External Memory Emulation window



Leave the debug session. These changes will be stored in the debug files.

Start debug session again.

Set a breakpoint after initialisation of external bus interface. (E.g. using FME start91460.asm file, set breakpoint after start91460.asm execution)

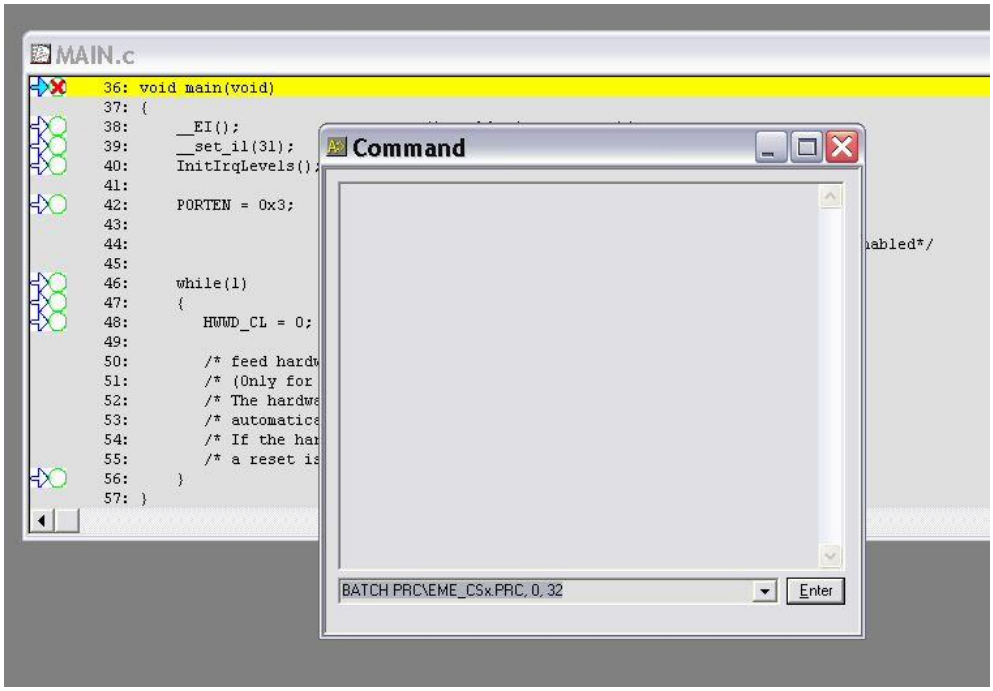
Open the Command Window and start the procedure file using following command sequence:

BATCH PRC\EME_CSx.PRC, <ChipSelect>, <Access Width>

<ChipSelect: 0, 1, 2, 3, 4, 5, 6> = Chip Select to be emulated

<Access Width: 8, 16, 32> = External bus data width (8-/16-/32-bit)

Figure 3 . Command Window usage



The bus interface is now configured to access the memory on the RAM extension board.

3 Appendix

3.1 eme_cs.prc file

```
# eme_cs.prc
# ===========
#
# Procedurefile for the FR-Emulator with EMA-MB91V460A-200.
# The procedure checks reads the configuration Register for
# external memory emulation
# set also the external memory emulation in the menue:
# setup - debug environment - debug environment external memory emulation

# To use this PRC file please follow the instructions below!
# 1. Run through startup procedure (ext Bus initialization in startup.asm)
and break @ main
# 2. Open a command window
# 3. Type following command: BATCH PRC\EME_CSx.PRC, <ChipSelect>, <Access
Width>
#     <ChipSelect : 0, 1, 2, 3, 4, 5, 6>     = Cips Selected to be emulated
#     <Access Width : 8, 16, 32> = External Bus Data Width (32/16/8 Bit)
#-------------------------------------------------------------------------
# variables
set variable CS = %P0
set variable AW = %P1
# --- hardware register
set variable PFR10 = 0xD8A
# address for CS:  (ASR1, ACR1, ACR1H, ACR1H)
set variable ASR  = %EVAL(0x640 + 4*%CS)
set variable ACR  = %EVAL(0x642 + 4*%CS)
set variable AWR  = %EVAL(0x660 + 2*%CS)
set variable ACRH = %EVAL(0x642 + 4*%CS)
set variable ACRL = %EVAL(0x642 + 4*%CS +1)
set variable CSE  = 0x680
# ------------------------------------------------------
set logging/UNEXPANSION CSconfig.log
            # enable logging
```

```
printf "-------------------------------------------------------\n"
printf "Register overview for CS%X\n", %CS
printf "and Access Size %X Bit\n\n", %AW
printf "Original Register Settings:\n"
printf "PFR10 : %X = %1X \n", %PFR10, %B(%PFR10)
printf "ASR%X  : %X = %1X \n", %CS, %ASR, %H(%ASR)
printf "ACR%X  : %X = %1X \n", %CS, %ACR, %H(%ACR)
printf "AWR%X  : %X = %1X \n", %CS, %AWR, %H(%AWR)

#32Bit access
IF %AW == 32
       set memory /HALFWORD (0x642 + 4*%CS) = 0x7822
ENDIF
#16Bit access
IF %AW == 16
       set memory /HALFWORD (0x642 + 4*%CS) = 0x7422
ENDIF
#8Bit access
IF %AW == 8
       set memory /HALFWORD (0x642 + 4*%CS) = 0x7022
ENDIF
set memory /HALFWORD (0x660 + 2*%CS) = 0x4378
printf "Emulation Register Settings:\n"
printf "ASR%X  : %X = %1X \n", %CS, %ASR, %H(%ASR)
printf "ACR%X  : %X = %1X \n", %CS, %ACR, %H(%ACR)
printf "AWR%X  : %X = %1X \n", %CS, %AWR, %H(%AWR)
printf "\n"
show memory /bit PFR10:5
IF %Bit(%PFR10:5) == 0
       printf "port 10.5 is used as IO port\n"
ELSE
       printf "port 10.5 is used as MCLKI\n"
ENDIF
IF %CSE == 0
       printf "no chip select available!\n"
ENDIF
IF %Bit(%CSE:0) == 1
       printf "chip select 0 activated!\n"
ENDIF
IF %Bit(%CSE:1) == 1
       printf "chip select 1 activated!\n"
ENDIF
IF %Bit(%CSE:2) == 1
       printf "chip select 2 activated!\n"
ENDIF
IF %Bit(%CSE:3) == 1
       printf "chip select 3 activated!\n"
ENDIF
IF %Bit(%CSE:4) == 1
       printf "chip select 4 activated!\n"
ENDIF
IF %Bit(%CSE:5) == 1
       printf "chip select 5 activated!\n"
ENDIF
IF %Bit(%CSE:6) == 1
       printf "chip select 6 activated!\n"
ENDIF
IF %Bit(%CSE:7) == 1
       printf "chip select 7 activated!\n"
ENDIF
```

```
printf "\n"
printf "CS%X - area select : 0x%04X0000 \n", %CS, %H(%ASR)
printf "\n"
# check ASZ -------------------------------------------------
set variable counter = (%B(%ACRH) / 0x10)
set variable size = 40
# printf "ACR%XH  : %X = %02X \n", %CS, %ACRH, %B(%ACRH)
# printf "counter : %X\n", %counter
WHILE counter != 0
       set variable size = size * 2
       SET VARIABLE counter = %EVAL(%counter-1)
ENDW

printf "CS%X - ASZ  : %01X   -->  ", %CS, (%B(%ACRH) / 0x10)
IF (%B(%ACRH) / 0x10) <4
       printf "memory size : %d KB\n",size
ELSE
       printf "memory size : %d MB\n",size/400
ENDIF
# check DBW -------------------------------------------------
printf "CS%X - DBW  : %01X   --> ", %CS, ((%B(%ACRH) / 0x4) & 0x3)
IF ((%B(%ACRH) / 0x4) & 0x3) == 0
       printf " 8 bit \n"
ELSEIF ((%B(%ACRH) / 0x4) & 0x3) == 1
       printf " 16 bit \n"
ELSEIF ((%B(%ACRH) / 0x4) & 0x3) == 2
       printf " 32 bit \n"
ELSE
       printf " reserved \n"
ENDIF

# check BST -------------------------------------------------
printf "CS%X - BST  : %01X   -->  ", %CS, ((%B(%ACRH) / 0x1) & 0x3)
IF ((%B(%ACRH) / 0x1) & 0x3) == 0
       printf "1 (single access) \n"
ELSEIF ((%B(%ACRH) / 0x1) & 0x3) == 1
       printf "2 bursts (address boundary: 1 bit) \n"
ELSEIF ((%B(%ACRH) / 0x1) & 0x3) == 2
       printf "4 bursts (address boundary: 2 bit) \n"
ELSE
       printf "8 bursts (address boundary: 3 bit) \n"
ENDIF


# check SREN ------------------------------------------------
printf "CS%X - SREN : %1X   --> ", %CS, %BIT(%ACRL:7)
IF %BIT(%ACRL:7)    ==0
       print " disable sharing by BRQ/BGRNTX\n"
ELSE
       print " enable sharing by BRQ/BGRNTX\n"
ENDIF


# check PFEN ------------------------------------------------
printf "CS%X - PFEN : %1X   --> ", %CS, %BIT(%ACRL:6)
IF %BIT(%ACRL:6)    ==0
       print " disable prefetch\n"
ELSE
       print " enable prefetch\n"
ENDIF
```

```
# check WREN ---------------------------------------------------
printf "CS%X - WREN : %1X   --> ", %CS, %BIT(%ACRL:5)
IF %BIT(%ACRL:5)    ==0
       print " disable write\n"
ELSE
       print " enable write\n"
ENDIF


# check LEND ---------------------------------------------------
printf "CS%X - LEND : %1X   --> ", %CS, %BIT(%ACRL:4)
IF %BIT(%ACRL:4)    ==0
       print " big endian byte order\n"
ELSE
       print " little endian byte order\n"
ENDIF


# check TYPE ---------------------------------------------------
printf "CS%X - TYPE : %1X   -->  ", %CS, (%B(%ACRL) & 0x0F)

IF (%B(%ACRL) & 0x0C) == 0
       printf "normal access\n"
ELSEIF (%B(%ACRL) & 0x0C) == 1
       printf "address data multiplex access\n"
ELSEIF (%B(%ACRL) & 0x09) == 0
       printf "disable WAIT insertion by the RDY\n"
ELSEIF (%B(%ACRL) & 0x09) == 1
       printf "enable WAIT insertion by the RDY\n"
ELSEIF (%B(%ACRL) & 0x0A) == 2
       printf "use the WEnX pin as the write strobe\n"
ELSEIF (%B(%ACRL) & 0x0F) == 8
       printf "Memory Type A: SDRAM/FCRAM\n"
ELSEIF (%B(%ACRL) & 0x0F) == 8
       printf "Memory Type B: FCRAM\n"
ELSE
       printf "wrong setting\n"
ENDIF



cancel logging
```

3.2 Related Documentation

3.2.1 Appnotes:

- AN205222 - FR Family MB2198-01 Emulator System Getting Started Guide
- AN204828 - F2MC-16FX Family, Emulating and Debugging with Softune and MB2198-01
- AN205146 - FR Family, MB91460 Emulation System

3.2.2 Documentation:

- MB91460A series Hardware Manual
- EMA-MB91V460A-00x User Guide
- EMA-MB91FV460B-00x User Guide
- EMA-MB91V460A-200 (Emulation RAM board) User Guide
- EMA-MB91V460A-210 (Emulation RAM board) User Guide

Document History

Document Title: AN205257 - FR, MB91460, Emulation RAM Extension Board Usage

Document Number: 002-05257

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	NOFL	12/05/2007	1.0, MSt Initial version
			06/02/2010	1.1, CEy Updated Application Note so that is also applicable for new Emulation RAM extension board EMA-MB91V460A-210
*A	5083714	NOFL	01/13/2016	Converted Spansion Application Note “MCU-AN-300048-E-V11” to Cypress format
*B	5870411	AESATMP9	09/01/2017	Updated logo and copyright.
*C	6058914	NOFL	02/05/2018	Removed references of obsolete specs across the document. Updated hyperlinks across the document. Updated to new template. Completing Sunset Review.

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 MCU

Cypress Developer Community

Community | 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, 2007-2018. 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.