F²MC-16FX FAMILY
16-BIT MICROCONTROLLER
ALL SERIES

SETUP AND DEBUGGING WITH EUROSCOPE

APPLICATION NOTE
Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Issue</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007-05-15</td>
<td>V0.1; 1st version; MWi</td>
</tr>
<tr>
<td>2007-06-13</td>
<td>V0.2; Software and hardware considerations added; MWi</td>
</tr>
<tr>
<td>2007-06-21</td>
<td>V0.3; Corrections done; MWi</td>
</tr>
<tr>
<td>2007-06-21</td>
<td>V1.0; 1st release; MWi</td>
</tr>
<tr>
<td>2007-06-26</td>
<td>V1.1; Workaround for local view added; MWi</td>
</tr>
<tr>
<td>2007-07-02</td>
<td>V1.2; License dialog changed; MWi</td>
</tr>
<tr>
<td>2007-07-06</td>
<td>V1.3; Configuration notes added; MWi</td>
</tr>
<tr>
<td>2007-07-10</td>
<td>V1.4; Module view change added; MWi</td>
</tr>
<tr>
<td>2007-07-30</td>
<td>V1.5; Updated and USB support note added; MWi</td>
</tr>
<tr>
<td>2007-09-25</td>
<td>V1.6; Reviewed the document and updated with review findings; MPi</td>
</tr>
<tr>
<td>2007-10-04</td>
<td>V1.7; Updated figure 2.1; MPi</td>
</tr>
<tr>
<td>2007-10-25</td>
<td>V1.8; update features of EUROScope Lite; add supported USARTs; PHu</td>
</tr>
<tr>
<td>2007-11-09</td>
<td>V1.9; update features of EUROScope Lite; add notes on supported USARTs; Add note about data breakpoints for mirrored constants; simplify PLL factors table; clarify supported baud rates; PHu</td>
</tr>
<tr>
<td>2008-01-14</td>
<td>V2.0; changes due to EUROScope version 3.0 done; MWi</td>
</tr>
<tr>
<td>2008-01-30</td>
<td>V2.1; device restriction appendix updated, functionality explanations added; MWi</td>
</tr>
<tr>
<td>2008-02-04</td>
<td>V2.2; typos corrected; MWi</td>
</tr>
<tr>
<td>2008-03-10</td>
<td>V2.3; overview of differences Lite/Professional version added (Appx. B); MWi</td>
</tr>
<tr>
<td>2008-04-15</td>
<td>V2.4; Changes to EUROScope version 3.01 added; MWi</td>
</tr>
<tr>
<td>2008-12-09</td>
<td>V2.5; Notes on using EUROScope while Flash Programming added; MWi</td>
</tr>
<tr>
<td>2008-01-07</td>
<td>V2.6; Changes to EUROScope version 3.02 and 3.03 added; MWi</td>
</tr>
</tbody>
</table>

This document contains 66 pages.
Warranty and Disclaimer

To the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH restricts its warranties and its liability for all products delivered free of charge (e.g. software include or header files, application examples, target boards, evaluation boards, engineering samples of IC’s etc.), its performance and any consequential damages, on the use of the Product in accordance with (i) the terms of the License Agreement and the Sale and Purchase Agreement under which agreements the Product has been delivered, (ii) the technical descriptions and (iii) all accompanying written materials. In addition, to the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH disclaims all warranties and liabilities for the performance of the Product and any consequential damages in cases of unauthorized decompiling and/or reverse engineering and/or disassembling. Note all these products are intended and must only be used in an evaluation laboratory environment.

1. Fujitsu Microelectronics Europe GmbH warrants that the Product will perform substantially in accordance with the accompanying written materials for a period of 90 days form the date of receipt by the customer. Concerning the hardware components of the Product, Fujitsu Microelectronics Europe GmbH warrants that the Product will be free from defects in material and workmanship under use and service as specified in the accompanying written materials for a duration of 1 year from the date of receipt by the customer.

2. Should a Product turn out to be defect, Fujitsu Microelectronics Europe GmbH’s entire liability and the customer’s exclusive remedy shall be, at Fujitsu Microelectronics Europe GmbH’s sole discretion, either return of the purchase price and the license fee, or replacement of the Product or parts thereof, if the Product is returned to Fujitsu Microelectronics Europe GmbH in original packing and without further defects resulting from the customer’s use or the transport. However, this warranty is excluded if the defect has resulted from an accident not attributable to Fujitsu Microelectronics Europe GmbH, or abuse or misapplication attributable to the customer or any other third party not relating to Fujitsu Microelectronics Europe GmbH.

3. To the maximum extent permitted by applicable law Fujitsu Microelectronics Europe GmbH disclaims all other warranties, whether expressed or implied, in particular, but not limited to, warranties of merchantability and fitness for a particular purpose for which the Product is not designated.

4. To the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH’s and its suppliers’ liability is restricted to intention and gross negligence.

**NO LIABILITY FOR CONSEQUENTIAL DAMAGES**

To the maximum extent permitted by applicable law, in no event shall Fujitsu Microelectronics Europe GmbH and its suppliers be liable for any damages whatsoever (including but without limitation, consequential and/or indirect damages for personal injury, assets of substantial value, loss of profits, interruption of business operation, loss of information, or any other monetary or pecuniary loss) arising from the use of the Product.

Should one of the above stipulations be or become invalid and/or unenforceable, the remaining stipulations shall stay in full effect.
Contents

REVISION HISTORY .................................................................................................................. 2
WARRANTY AND DISCLAIMER ............................................................................................... 3
CONTENTS ............................................................................................................................... 4
0 INTRODUCTION .................................................................................................................. 7
1 LICENSE & INSTALLATION ................................................................................................. 8
  1.1 Obtain License for using EUROS .................................................................................. 8
  1.2 Installation Remark ..................................................................................................... 10
2 APPLICATION PREPARATION ......................................................................................... 11
  2.1 MCU Boot sequence ................................................................................................... 11
  2.2 Background Debug Configuration Block ...................................................................... 12
    2.2.1 BDM Activation Marker (BDMAM) ...................................................................... 12
    2.2.2 BDM Configuration Marker (BDMCM) ................................................................ 13
    2.2.3 BDM Baud Rate Divider Marker (BBRDM) ............................................................ 14
    2.2.4 BDM External Address Marker (BEAM) ............................................................... 15
    2.2.5 BDM Watchdog Pattern Marker (BWPM) ............................................................. 15
    2.2.6 BDM Default Breakpoint Marker (BDBM) ............................................................ 16
  2.3 Configuration Assembly Module .................................................................................. 17
  2.4 Configuration by EUROScope ..................................................................................... 18
  2.5 Application Software Considerations .......................................................................... 19
  2.6 Application Hardware Considerations .......................................................................... 20
3 STARTING A DEBUG SESSION .......................................................................................... 21
  3.1 Building and Flashing the User Application ............................................................... 21
    3.1.1 Flash Support by EUROScope ............................................................................. 21
  3.2 Starting EUROScope ................................................................................................... 22
    3.2.1 Loading Configuration .......................................................................................... 22
    3.2.2 Loading application Absolute Format Load Module (ABS File) .......................... 23
    3.2.3 File Path to Source Files ..................................................................................... 25
4 DEBUGGING – FIRST STEPS ............................................................................................. 27
  4.1 Example application ...................................................................................................... 27
  4.2 Initialize Target and run until Main ............................................................................ 27
  4.3 Start execution .............................................................................................................. 28
  4.4 Stop execution .............................................................................................................. 28

MCU-AN-300235-E-V26 - 4 - © Fujitsu Microelectronics Europe GmbH
Contents

5 BREAKPOINTS AND PROGRAM STEPPING ............................................................... 29
  5.1 General ................................................................................................................... 29
  5.2 Setting Breakpoints ................................................................................................. 29
    5.2.1 Setting in Code Windows .......................................................................... 29
    5.2.2 Setting by Menu Dialog ............................................................................. 30
    5.2.3 Setting Data Breakpoint ............................................................................ 31
    5.2.4 Setting by Click Button .............................................................................. 32
  5.3 Using Breakpoints ................................................................................................... 32
  5.4 Setting Range Breaks ............................................................................................. 32
    5.4.1 Data Range Break..................................................................................... 32
    5.4.2 Code Range Break.................................................................................... 33
  5.5 Number of available Breakpoints ............................................................................ 33
  5.6 Stepping through Application .................................................................................. 34
  5.7 Source Code Window Handling .............................................................................. 36

6 VARIABLE WATCHING AND MANIPULATING ............................................................. 39
  6.1 General ................................................................................................................... 39
  6.2 Watching and Manipulating Processor Status ........................................................ 39
  6.3 Watching and Manipulating CPU Registers............................................................ 39
  6.4 Watching and Manipulating C Variables ................................................................. 40
  6.5 Watching and Manipulating Memory....................................................................... 41
    6.5.1 Change Contents ...................................................................................... 41
    6.5.2 Fill Memory................................................................................................ 42
  6.6 Automatic Local Variable Watch View .................................................................... 42

7 FUNCTION BROWSER ................................................................................................... 43
  7.1 General ................................................................................................................... 43
  7.2 Opening Function Browser ...................................................................................... 43

8 SFR VIEW ........................................................................................................................ 44
  8.1 General ................................................................................................................... 44
  8.2 Opening SFR View.................................................................................................. 44
  8.3 Using SFR View ...................................................................................................... 44

9 CALLSTACK VIEW ......................................................................................................... 45
  9.1 General ................................................................................................................... 45
  9.2 Opening the Callstack View .................................................................................... 45
  9.3 Using Callstack View............................................................................................... 45

10COMMAND LINE INTERPRETER ................................................................................... 47
  10.1 General 47
10.2 Opening the Command Line Interpreter ................................................................. 47

11 TERMINAL WINDOW ...................................................................................................... 48
   11.1 General 48
   11.2 Opening Terminal Window ................................................................................ 48

12 FLASHING AN APPLICATION IN INTERNAL/EXTERNAL VECTOR MODE ................. 49
   12.1 Re-Flashing Application ..................................................................................... 49
   12.2 Downloading Application .................................................................................. 50
   12.3 Smart Download ............................................................................................... 51
   12.4 Breakpoints after Flashing ................................................................................ 51

13 FLASHING AN APPLICATION IN SERIAL COMMUNICATION MODE ......................... 52
   13.1 (Re-)Flashing Application ................................................................................ 52

14 ENDING A DEBUG SESSION ....................................................................................... 53
   14.1 Communication Closure .................................................................................... 53
   14.2 Save Configuration ........................................................................................... 53
   14.3 New Build during Debug Session ...................................................................... 53

15 NOTES ON USING EUROSCOPE ............................................................................... 54
   15.1 Reset Target ..................................................................................................... 54
   15.2 Serial Connection ............................................................................................ 54
   15.3 Flash Programming .......................................................................................... 54
      15.3.1 Source Code .......................................................................................... 55
      15.3.2 Assembly Code ..................................................................................... 56
   15.4 USB support (SK-16FX-100PMC Board) ............................................................. 57

16 APPENDIX A ............................................................................................................... 59
   16.1 Devices which do not support Background Debugging ...................................... 59
   16.2 Maximum PLL Factor Device List ....................................................................... 59
   16.3 Breakpoint at PLL Register (PLLCR) Access ..................................................... 59
   16.4 Supported USARTs ......................................................................................... 60

17 APPENDIX B ............................................................................................................... 62
   17.1 Lite/Professional Differences Table and Implementation since Version 3.0x ......... 62

18 RELATED APPLICATION NOTES AND CONTACT ...................................................... 63
   18.1 Manuals, Application Notes and Customer Information .................................... 63
   18.2 Contact to Fujitsu ............................................................................................ 63

LIST OF FIGURES .............................................................................................................. 64
LIST OF TABLES ................................................................................................................ 66
0 Introduction

This application note describes the necessary steps to setup a debugging environment using EUROScope and shows how to debug an application.
1 License & Installation

LICENCE REGULATION AND INSTALLATION REMARK

1.1 Obtain License for using EUROS

To use EUROScope and EUROScope Lite, a valid license is necessary. When starting EUROScope just after installing, you will get the following notification window:

![EUROS License Key](image)

Please note down your host ID. This host ID is necessary for registration and checked together with the license by EUROScope at every start.

If you click at one of the "obtain licensing key" buttons your standard browser will open and you will be directed to a registration website.

At the Fujitsu Softune software registration web page (accessible via button No 2), there is also a field for obtaining EUROScope-Lite license. Please find a screen shot at the next page. It is mandatory that all fields with an *) are filled out. Otherwise, this will create counter questions from Fujitsu’s side and delay and hinder the license procedure.

The license key will be automatically sent to you by e-mail, which then has to be copied to your EUROScope installation directory. Note that the received key might be stored in a spam folder by your email host or by your client software.

The registration website for EUROScope-Lite can also be found at:


Note that special characters (like “ä”, “ö”, “ü”, “ß”) have to be avoided in the fields of the registration form. Otherwise the generated license key may not work.
SOFTWARE REGISTRATION EUROPE (EMEA)

This form should be used to register your Software Development Environment. Prior to the installation, please check the Fujitsu Disclaimer and Privacy. This registration form is valid for European area only. If your location is outside Europe (EMEA), please contact your local sales office.

"Required field"

Address
- Company *
- Title *
- First name *
- Street *
- ZIP/Postal code *
- City *
- Country *
- Local Fujitsu or
  Installation Sales
  Office *
- Phone *
- Email *
- Application Area *
- Present Description

Newsletter
- Newsletter will periodically be sent to the E-Mail address mentioned above.

Software Workbench
- 8Bit (FM1-8)
- 16Bit (FM1-16)
- 32Bit (PR)
- Number of installations
  on all PC's
- PC Operating System

EUROScape lite
- 16FX
- Host D *

Information will be sent to Fujitsu FME partner EUROs Embedded Systems GmbH.

Terms of Use *
- Please check this box to fully agree with the Disclaimer and Privacy.

Clear Form | Submit

Figure 1-2: EUROS Software Registration
**Note:** When changing your host ID (e.g. because of new hardware), you have to obtain a new license key.

### 1.2 Installation Remark

During the installation process you are asked for the target architecture. Please only check Fujitsu F16LX/F16FX:
2 Application Preparation

HOW TO MAKE AN APPLICATION DEBUG-ABLE FOR BACKGROUND DEBUGGING

2.1 MCU Boot sequence

The 16FX MCUs have several operation modes (those decide the course of action after reset). In this application note the so-called “Background Debugging Mode” (BDM) is described using internal vector mode (single-chip) mode. The following flow chart shows the boot sequence. The mode-of-interest is shown in the red-dotted-line box.

![Flow chart showing the MCU boot sequence.]

**Important Note:** Since the communication from target to host PC (EUROScope) needs a USART channel, this USART channel must not used in the user application. Be also careful while setting the Port Input Enable Register (PIER) and Data Direction Register (DDR) which includes the SIN pin setting of this USART. It must stay enabled for debugging. Please also see chapter 2.5.
2.2 Background Debug Configuration Block

To make an application debug-able a special configuration has to be done. This Background Debugging Configuration Block (BDCB) is located in the address space from 0xDF0040 to 0xDF007E.

In the next step, this BDCB is explained in detail. Later it is shown, how this configuration can be appended to an existing application or done by EUROScope itself.

2.2.1 BDM Activation Marker (BDMAM)

The background debugging is enabled, if the long-word contents at addresses 0xDF0040 (0xDF0040-0xDF0043) equals to 0x292D3A7B (“Magic Word”).

<table>
<thead>
<tr>
<th>Address</th>
<th>Content</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xDF0040</td>
<td>0x7B</td>
<td>Enable BDM (if ROM/Flash security is disabled) – any other content disables BDM.</td>
</tr>
<tr>
<td>0xDF0041</td>
<td>0x3A</td>
<td></td>
</tr>
<tr>
<td>0xDF0042</td>
<td>0x2D</td>
<td></td>
</tr>
<tr>
<td>0xDF0043</td>
<td>0x29</td>
<td></td>
</tr>
</tbody>
</table>

Table 2-1: BDMAM

Setting Flash Security Byte (MSFB/SFSB) and BDM Enable Marker together is not recommended. The following table shows the recommended setting for BDMAM with reference to MSFB/SFSB:

<table>
<thead>
<tr>
<th>BDMAM</th>
<th>MSFB</th>
<th>SFSB</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;&gt;0x292D3A7B</td>
<td>&lt;&gt;0x99</td>
<td>&lt;&gt;0x99</td>
<td>BDM disabled, Security disabled</td>
</tr>
<tr>
<td>=0x292D3A7B</td>
<td>&lt;&gt;0x99</td>
<td>&lt;&gt;0x99</td>
<td>BDM enabled, Security disabled</td>
</tr>
<tr>
<td>&lt;&gt;0x292D3A7B</td>
<td>&lt;&gt;0x99</td>
<td>=0x99</td>
<td>BDM disabled, Security enabled</td>
</tr>
<tr>
<td>=0x292D3A7B</td>
<td>&lt;&gt;0x99</td>
<td>=0x99</td>
<td>BDM enabled, Security enabled, Not recommended</td>
</tr>
</tbody>
</table>

Table 2-2: Recommended setting for BDMAM
### 2.2.2 BDM Configuration Marker (BDMCM)

In the following configuration several conditions and UART settings can be done.

<table>
<thead>
<tr>
<th>Address</th>
<th>Bit No.</th>
<th>Name</th>
<th>Content</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 0 UCS</td>
<td>0, 0</td>
<td>USART A Channel Selection</td>
<td>0, 0</td>
<td>USART A</td>
</tr>
<tr>
<td></td>
<td>0, 1</td>
<td>USART B</td>
<td>0, 1</td>
<td>USART B</td>
</tr>
<tr>
<td></td>
<td>1, 0</td>
<td>USART C</td>
<td>1, 0</td>
<td>USART C</td>
</tr>
<tr>
<td></td>
<td>1, 1</td>
<td>USART D</td>
<td>1, 1</td>
<td>USART D</td>
</tr>
<tr>
<td>3, 2 MODE</td>
<td>0, 0</td>
<td>Communication Mode</td>
<td>0, 0</td>
<td>Asynchronous</td>
</tr>
<tr>
<td></td>
<td>0, 1</td>
<td></td>
<td>0, 1</td>
<td>Synchronous*</td>
</tr>
<tr>
<td></td>
<td>1, 0</td>
<td></td>
<td>1, 0</td>
<td>K-Line*</td>
</tr>
<tr>
<td></td>
<td>1, 1</td>
<td>Reserved</td>
<td>1, 1</td>
<td>Reserved</td>
</tr>
<tr>
<td>0xDF0044</td>
<td>4</td>
<td>STRT</td>
<td>0</td>
<td>Automatic Start</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Run User Program after Reset</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>EBPC</td>
<td>0</td>
<td>External breakpoint configuration*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Read Breakpoint setting from 0xDF0050</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>KRC</td>
<td>0</td>
<td>Keep RC-Clock*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Stick to RC-Clock if active on Break</td>
</tr>
<tr>
<td>0xDF0045</td>
<td>7</td>
<td>CAL</td>
<td>0</td>
<td>Calibrate USART baud rate</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>KBCD</td>
<td>0</td>
<td>Increase frequency up to Main Clock</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>EMON</td>
<td>0</td>
<td>Use Boot-ROM internal BDM Monitor</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>..</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

* Not supported yet

| Table 2-3: BDMCM |
### 2.2.3 BDM Baud Rate Divider Marker (BBRDM)

<table>
<thead>
<tr>
<th>Address</th>
<th>Bit No.</th>
<th>Name</th>
<th>Content</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xDF0046</td>
<td>-</td>
<td>BBRDML</td>
<td>X</td>
<td>USART Baud Rate Divider lower Byte</td>
</tr>
<tr>
<td>0xDF0047</td>
<td>-</td>
<td>BBRDMH</td>
<td>X</td>
<td>USART Baud Rate Divider higher Byte</td>
</tr>
</tbody>
</table>

Table 2-4: BBRDM

The USART Baud Rate Divider should be bigger or equal to 20 and smaller than $2^{16}$. Please be aware that the chosen USART channel must not used in the user application. The divider $div$ should be calculated by:

\[
div = \left\lfloor \frac{f_{\text{crystal}} \cdot \text{MaxPLLFactor} + f_{\text{baud}}}{f_{\text{baud}}} \right\rfloor
\]

- $f_{\text{crystal}}$: maximum possible PLL factor (see Appendix A (16))
- $f_{\text{baud}}$: used external crystal frequency
- $f_{\text{baud}}$: USART baud rate
- $\left\lfloor \cdot \right\rfloor$: floor function

The following part of start.asm derives divider $div$ and makes sure it is stored at the correct location (0xDF0046-0xDF0047) in RCB once the hex image is generated.

```assembly
; THIS SAMPLE CODE IS PROVIDED AS IS AND IS SUBJECT TO ALTERATIONS. FUIJITSU;
; MICROELECTRONICS ACCEPTS NO RESPONSIBILITY OR LIABILITY FOR ANY ERRORS OR;
; ELIGIBILITY FOR ANY PURPOSES.
; (c) Fujitsu Microelectronics Europe GmbH;
;---------------------------------------------------------------------------
; 6.18 Debug address specification
;---------------------------------------------------------------------------
; BDM configuration section should always be defined for later
; configuration by e.g. debugger tool or (special) programmer tool.

.SECTION BDM_CONFIG, CONST, LOCATE=H'DF0040
#if BACKGROUND_DEBUGGING == ON
.DATA.L 0x292D3A7B
.ORG H'DF0044
.DATA.W BDM_CONFIGURATION
.ORG H'DF0046
#if (SERIES == MB96340 && DEVICE < 7)
.DATA.W (D'16 * CRYSTAL + BDM_BAUDRATE) / BDM_BAUDRATE
#else
.DATA.W (D'32 * CRYSTAL + BDM_BAUDRATE) / BDM_BAUDRATE
#endif ; (SERIES == MB96340 && if DEVICE < 7)
```

The divider value $div$ along with PLL clock Multiplier (PLLCR:PMS) and Peripheral Clock 1 Divider (CKFPCR:PC1D) are used by the Boot ROM to calculate the final value the goes into BGRn register in order to get the desired baud rate.

Note that the chosen baud rate has to be fit to the USART settings. Also consider the given values of EUROScope (Preferences → Configure Target Connection : Baudrate) and capability of the USART baud rate generator together with the peripheral clock settings.
The MCU will adapt to this baud rate automatically as long as the Peripheral Clock 1 (CLKP1) is set to RC Clock (2 MHz setting), Main Clock, or PLL Clock.

### 2.2.4 BDM External Address Marker (BEAM)

This 24 Bit marker (from address $0xDF0048$ to $0xDF0050$) defines an address outside the ROM Configuration Block. Depending on the BDM Configuration Marker, the address has two meanings:

<table>
<thead>
<tr>
<th>EMON</th>
<th>EBPC</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Address marker is not used</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Address marker points to an address for default breakpoint configuration of Patch Function.</td>
</tr>
<tr>
<td>1</td>
<td>X</td>
<td>Address marker points to an address for alternative monitor program.</td>
</tr>
</tbody>
</table>

Table 2-5: BEAM

### 2.2.5 BDM Watchdog Pattern Marker (BWPM)

This 8 Bit marker (at address $0xDF004B$) defines reset pattern for the internal watchdog in Foreground Debug Mode. If the marker value equal to $0x00$ then only the pattern $0x00$ is written to clear the watchdog. For any value other than $0x00$ the marker value and its inverse are written pair-wise.
### 2.2.6 BDM Default Breakpoint Marker (BDBM)

If STRT bit of the BDM Configuration Marker (BDMCM) is set and EPBC bit of BDMCM is cleared, the following settings are copied to the appropriate registers of Memory Patch Function unit after Reset and before entering debugging mode. If PFCSn’ = 0x0000 the power-on breakpoint functionality is disabled.

<table>
<thead>
<tr>
<th>Address</th>
<th>Name</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xDF0050</td>
<td>PFCS0’</td>
<td>Initialization value for Patch Function Control/Status Registers (16 bit)</td>
</tr>
<tr>
<td>0xDF0052</td>
<td>PFCS1’</td>
<td></td>
</tr>
<tr>
<td>0xDF0054</td>
<td>PFCS2’</td>
<td></td>
</tr>
<tr>
<td>0xDF0056</td>
<td>PFCS3’</td>
<td></td>
</tr>
<tr>
<td>0xDF0058</td>
<td>PFA0’</td>
<td></td>
</tr>
<tr>
<td>0xDF005B</td>
<td>PFA1’</td>
<td></td>
</tr>
<tr>
<td>0xDF005E</td>
<td>PFA2’</td>
<td></td>
</tr>
<tr>
<td>0xDF0061</td>
<td>PFA3’</td>
<td></td>
</tr>
<tr>
<td>0xDF0064</td>
<td>PFA4’</td>
<td></td>
</tr>
<tr>
<td>0xDF0067</td>
<td>PFA5’</td>
<td></td>
</tr>
<tr>
<td>0xDF006A</td>
<td>PFA6’</td>
<td></td>
</tr>
<tr>
<td>0xDF006D</td>
<td>PFA7’</td>
<td></td>
</tr>
<tr>
<td>0xDF0070</td>
<td>PFD0’</td>
<td></td>
</tr>
<tr>
<td>0xDF0072</td>
<td>PFD1’</td>
<td></td>
</tr>
<tr>
<td>0xDF0074</td>
<td>PFD2’</td>
<td></td>
</tr>
<tr>
<td>0xDF0076</td>
<td>PFD3’</td>
<td></td>
</tr>
<tr>
<td>0xDF0078</td>
<td>PFD4’</td>
<td></td>
</tr>
<tr>
<td>0xDF007A</td>
<td>PFD5’</td>
<td></td>
</tr>
<tr>
<td>0xDF007C</td>
<td>PFD6’</td>
<td></td>
</tr>
<tr>
<td>0xDF007E</td>
<td>PFD7’</td>
<td></td>
</tr>
</tbody>
</table>

Table 2-6: BDBM
2.3 Configuration Assembly Module

The following assembly module shows an example how to configure the BDM using `start.asm`, if you are using our standard templates.

```asm
; THIS SAMPLE CODE IS PROVIDED AS IS AND IS SUBJECT TO ALTERATIONS. FUJITSU
; MICROMELECTRONICS ACCEPTS NO RESPONSIBILITY OR LIABILITY FOR ANY ERRORS OR
; ELIGIBILITY FOR ANY PURPOSES.
; (C) Fujitsu Microelectronics Europe GmbH

;---------------------------------------------------------------------------
;====================================================================
; 4.18  Enable Background Debugging Mode
;====================================================================

#set      BACKGROUND_DEBUGGING  ON  ; <<< enable Background Debugging
; mode

#set      BDM_CONFIGURATION  B'0000000000010000  ; <<< set BDM configuration
; ||||||||++--- BdmUART
; ||||||||      (0: A, 1: B, 2: C, 3: D)
; ||||||++----- BdmComMode
; |||      (0: Async., 01: Sync.,
; ||| 10: BdmKLine, 11: res.)
; ||+------- BdmAutoStart
; |+-------- BdmExtBreakpointCfg
; |+--------- BdmKeepRClock
; +------------ BdmCaliRClock
; +------------- BdmUserKernel

#set      BDM_BAUDRATE    115200   ; <<< set Baudrate in Bits/s for BDM

#set      BDM_EXT_CONFIG  0xFFFFFF ; <<< set external Config/Kernel

#set      BDM_WD_PATTERN  0x00     ; <<< set watchdog pattern

#set      BDM_PFCS0       0x0000   ; <<< set default breakpoint
#set      BDM_PFCS1       0x0000   ; configurations
#set      BDM_PFCS2       0x0000
#set      BDM_PFCS3       0x0000

#set      BDM_PFA0        0xFFFFFF ; <<< set address
#set      BDM_PFA1        0xFFFFFF ; configurations
#set      BDM_PFA2        0xFFFFFF
#set      BDM_PFA3        0xFFFFFF
#set      BDM_PFA4        0xFFFFFF
#set      BDM_PFA5        0xFFFFFF
#set      BDM_PFA6        0xFFFFFF
#set      BDM_PFA7        0xFFFFFF

#set      BDM_PFD0        0xFFFF   ; <<< set patch data
#set      BDM_PFD1        0xFFFF   ; configurations
#set      BDM_PFD2        0xFFFF
#set      BDM_PFD3        0xFFFF
#set      BDM_PFD4        0xFFFF
#set      BDM_PFD5        0xFFFF
#set      BDM_PFD6        0xFFFF
#set      BDM_PFD7        0xFFFF
```

© Fujitsu Microelectronics Europe GmbH - 17 - MCU-AN-300235-E-V26
2.4 Configuration by EUROScope
EUROScope V3.00 offers Flash programming and BDM configuration. The configuration occurs, if the user clicks on the Flash menu button.

The BDM configuration corresponds to the settings described at 2.2 et seqq.
Please note, that also the unlock data for the Main and Satellite Flash memory can be set here.

**Note:** If the *BDM Activation marker* is unchecked the original settings of the MHX file are used.
2.5 Application Software Considerations

Because the communication from target to host PC (EUROScope) needs a USART channel, this USART channel must not used in the user application. Be also careful while setting the Port Input Enable Register (PIER) and Data Direction Register (DDR) which includes the SIN pin setting of this USART. It must stay enabled for debugging. If synchronous communication mode (not available yet) is used, the same appears to the corresponding SCK pin input enable and direction.

If port input enable registers are set to a default value at the beginning of an application, it is recommended to use a USART port input enable and direction mask.

Example:

```c
/* THIS SAMPLE CODE IS PROVIDED AS IS AND IS SUBJECT TO ALTERATIONS. FUJITSU */
/* MICROELECTRONICS ACCEPTS NO RESPONSIBILITY OR LIABILITY FOR ANY ERRORS OR */
/* ELIGIBILITY FOR ANY PURPOSES. */
/* (C) Fujitsu Microelectronics Europe GmbH */
/*----------------------------------------------------------------------------*/

// Example for MB96F34x USRAT0 for asynchronous Background Debugging
// USART-SIN0 : Port P08_2

#define PORT08_DD 0x0F // Example: Port 08_0 to 08_3 to output
// direction in application:
//
// D08_7 6 5 4 3 2 1 0
// 0 0 0 0 1 1 1 1

#define PORT08_IE 0xF0 // Example: Port 08_4 to 08_7 input is
// enabled in application:
//
// IE08_7 6 5 4 3 2 1 0
// 1 1 1 1 0 0 0 0

#define USART0_SIN_DD_MASK 0xFB // And-Mask P08_2 for Input Direction (=0)
#define USART0_SIN_IE_MASK 0x04 // Or-Mask P08_2 for Input Enable (=1)

.. .

DDR08  = PORT08_DD & USART0_SIN_DD_MASK;  // results in 0x0B:
//
// D08_7 6 5 4 3 2 1 0
// 0 0 0 0 1 1 1 1

PIER08 = PORT08_IE | USART0_SIN_IE_MASK;  // results in 0xF4:
//
// IE08_7 6 5 4 3 2 1 0
// 1 1 1 1 0 1 0 0

.. .
```
2.6 Application Hardware Considerations

The built-in firmware uses a USART channel for communication to the host PC. Therefore the corresponding SIN pin of the used USART is set to input. It is strongly recommended to connect this pin to VCC via a pull-up resistor as shown in the below figure to avoid floating pin state, when USART is not connected to the host PC.

Not using the pull-up may disturb the firmware and the application (if configured to auto-run) may also not start or may stop during run time.

Connecting the SIN pin to system ground (pull-down) is not recommended, because the USART may interpret this as a start bit condition even if no falling edge occurred.

**Note:** If synchronous communication mode (not supported yet) is used, the same is true and needs to be followed for the corresponding SCK pin.
3 Starting a Debug Session

HOW TO START A DEBUG SESSION

3.1 Building and Flashing the User Application

To start debugging the user has to build his application in order to generate a Loadmodule (ABS file) and a Motorola-Hex File (MHX file). Afterwards the application has to be flashed to the MCU by using the Fujitsu 16FX Flash Programmer (Please see help menu for detailed usage) and the generated MHX file.

![Flash Programmer](image)

Figure 3-1: Flash Programmer

After successful Flash programming close the Flash Programmer software to free the used COM port. Then the MCU has to be set to internal/external vector mode (single-chip/normal run) mode:

<table>
<thead>
<tr>
<th>MD0</th>
<th>MD1</th>
<th>MD2</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>H</td>
<td>L</td>
</tr>
</tbody>
</table>

Table 3-1: Mode Switch Setting

3.1.1 Flash Support by EUROScope

- For Flashing in internal/external vector mode (supported since version 3.00) with EUROScope please refer to chapter 12.
- For Flashing in serial communication mode (supported since version 3.01) with EUROScope please refer to chapter 13.
3.2 Starting EUROScope

After start of EUROScope it will appear like in the following picture.

![EUROScope](image)

The individual windows will be explained later. Note that the workspace view may differ, if the default configuration is loaded.

3.2.1 Loading Configuration

When EUROScope is installed for the first time it uses a default configuration (default.cfg), which is contained in the installation path. If the user creates his own configuration file, it will be used by the current EUROScope version and possible new installations. New installations would overwrite the default configuration file (default.cfg) provide the file path of the new and old installation is same.

A configuration can be loaded by Preferences → Load Configuration … .

![Load Configuration](image)
3.2.2 Loading application Absolute Format Load Module (ABS File)

Now use menu **File → Open Application** and browse to the directory of the abs file of your application.

![EUROScope – Loading ABS File](image)

A progress bar will appear as follows and afterwards the application is loaded to EUROScope.

![EUROScope – Progress Bar](image)

**Note:**

Once the configuration is saved, the ABS file will be loaded automatically at start of EUROScope and after re-flashing the application.

Then use menu **Preferences → Select Target Connection** and choose "Fujitsu 16FXBootROM (RS232)".

![EUROScope – Select Target Connection](image)

After this use menu **Preferences → Configure Target Connection**.
Then select the COM port for communication and the baud rate value same as that set in BBRDM of the configuration block (by BDM_BAUDRATE). If you are using a USB-RS232 converter, the emulated COM port can be seen in the device manager of the system settings, when connected (please refer to chapter 0 for more information on this).

Check "Stop at serial interrupt" and choose "Asynchronous communication" and "Int./Ext. vector mode". Finally check "Use fast download (unbuffered)".

After this configuration reset MCU and choose menu Communication → Open. If the communication to the Boot ROM of the MCU is successfully established almost all of the EUROScope windows are filled with contents.

Please note that opening the communication stops the running application.

**Note:** On some PC systems the automatic local variable view shows no contents. In this case please reload abs file.

Please also see chapter 15.4 (USB support) for troubleshooting.
3.2.3 File Path to Source Files
If the abs file is not located in the same folder as that of source code modules, it would be necessary to configure the source code path in EUROScope. This path(s) can be set as relative to the abs file location or an absolute path.

This can be done via the menu Preferences → Source Paths…

If Fujitsu template or example project is used, the source module path can be set by: ..\Src\n
![Figure 3-9: EUROScope – Source Path](image)

If the sources are still not displayed, please reload the abs file while being connected to target.
The following screenshot shows an example debugging session workspace:

- Assembly Window
- Source Code Window
- Terminal Window*
- Processor Register Window
- Memory Window
- Call Stack Window*
- Objects Window*
- Trace, Event, Coverage Window (not supported yet)*
- Breakpoint Window
- Variable Watch Window

Figure 3-10: EUROScope – Debugging Session

Note that all windows can be either floating at the workspace or be docked to others.

*) Not supported in EUROScope Lite version and not displayed by default configuration
## 4 Debugging – First steps

### HOW TO USE EUROSCOPE DEBUGGER

#### 4.1 Example application

For explanation of the debugging features of EUROScope the following "application" is used:

```c
/* THIS SAMPLE CODE IS PROVIDED AS IS AND IS SUBJECT TO ALTERATIONS. FUJITSU */
/* MICROELECTRONICS ACCEPTS NO RESPONSIBILITY OR LIABILITY FOR ANY ERRORS OR */
/* ELIGIBILITY FOR ANY PURPOSES. */
/* */
/* (C) Fujitsu Microelectronics Europe GmbH */
/*---------------------------------------------------------------------------*/
#include "mb96346rw.h"
volatile unsigned long cnt;

void wait (unsigned long a)
{
    for (cnt = 0; cnt < a; cnt++); // wait
}

void main(void)
{
    unsigned long local_cnt;
    PDR00 = 0x00; // switch off all LEDs (on Port00)
    DDR00 = 0xFF; // set parallel port direction register: output
    local_cnt = 80000;
    while(1)
    {
        wait(local_cnt);
        PDR00++; // counter...
    }
}
```

The setup for background debugging mode is described in chapter 2.

#### 4.2 Initialize Target and run until Main

To initialize the target system and stop at the top of the main function, click on \[\text{Initialize Target}\] or press “Shift-F2”. The stopped execution is indicated then with a yellow arrow, which stands for the actual program counter place in the source code (and assembly) window.

```c
void main(void)
{
    unsigned long local_cnt;
    PDR00 = 0x00; // switch off all LEDs (on Port00)
    DDR00 = 0xFF; // set parallel port direction register : output
}
```

**Figure 4-1: EUROScope – Initialize Target**

This BDM initialization replaces the Reset of the MCU.
4.3 Start execution

To start the application click on [Start] or press “F5”. If LEDs are connected on Port00 they will be flashing.

4.4 Stop execution

To stop or halt a running application click on [Stop] or press “F6”. The actual program counter is shown with a yellow arrow in the source code and assembly window.
5 Breakpoints and Program Stepping

HOW TO SET BREAKPOINTS AND HOW TO USE SINGLE STEPS

5.1 General

The number of hardware breakpoints depends on the used MCU. Mostly 4 or 8 breakpoints are available. Software breakpoints are unlimited, but can only set in the RAM area.

In the source code view not all lines are able to get a breakpoint. A breakable line is indicated by a small dot right from the line number.

![Small dots for breakpoints](image1)

Please note that there are grouped lines. They are connected with a line between the small dots. These lines can only hold a breakpoint together.

![Grouped lines](image2)

5.2 Setting Breakpoints

5.2.1 Setting in Code Windows

Breakpoints can be directly set in the assembly and source code window by double-clicking in the line column at the left window side. A filled red circle indicates an active break point.

![Setting Breakpoint](image3)

To deactivate a breakpoint, click into the red circle. The circle then gets white filled indicating that the breakpoint is deactivated.

![Deactivating Breakpoint](image4)

To delete a breakpoint double-click on it.
The following snapshot shows the same breakpoint in the assembly view.

![EUROScope – Breakpoint Assembly View]

**5.2.2 Setting by Menu Dialog**

Breakpoints also can be set in the breakpoint window by right-clicking and choosing *New*… .

![EUROScope – Setting Breakpoint by Menu Dialog]
In the address line an address or an application symbol can be set by the pull-down menu as shown below.

![EUROScope – Setting Breakpoint by Menu Dialog II](image)

Please only check “Execute” and “Force usage of hardware breakpoint”.

### 5.2.3 Setting Data Breakpoint

Another method of setting a (data) breakpoint is to use a symbol (variable or special function register). In this case please make sure, that the symbol begins with an underline and Read and Write break condition is checked.

![EUROScope – Setting Data Breakpoint by Menu Dialog](image)

Note that the program would be halted some instructions after the instruction which does actual data access due to the CPU instruction queue.
Note:
Be careful when using "Address in HEX". The following items will fail (until EUROScope version 3.0.3 inclusive):
   - Range breaks are interpreted as single break (first operand is used)
   - Variable names are interpreted as address, when beginning with A-F regardless of the rest of the variable name

5.2.3.1 Data Breakpoints at ROM Constants
In the small or medium memory model (_near) ROM constants cannot be set for a break point automatically when ROM-mirror function is used. The symbol reference is still located in 0xFF bank. When breaking on a constant read access, please change address in the break configuration from 0xFFxxxx to 0x00xxxx.

5.2.3.2 Data Breakpoints at local Variables
It is not recommendable to set a data breakpoint to a local variable, if they are not storage-class specified by static. Because of the unknown lifetime of local variables and the fix address of a data breakpoint, undefined behavior of EUROScope may result (sudden break of no visible cause).

5.2.4 Setting by Click Button
The last method to set/delete a breakpoint is to mark a line with the cursor and press the breakpoint button.

Please note that a breakpoint can only be set in the source view at lines, which have a dot (.) at the right side of the source line number (break-able code lines).

5.3 Using Breakpoints
Assume a breakpoint is set at the place in the example above. After starting the application with “F5” the execution will stop at the breakpoint shortly. A reached breakpoint is indicated with the yellow arrow in front of the filled red breakpoint circle.

5.4 Setting Range Breaks
Please note that range break is available since EUROScope version 3.01.

5.4.1 Data Range Break
To set a data range break, the user has to know, in which memory area observed variables are located. Assume that the user wants to have a break condition between 0x4250 and 0x4260. This address range can then be entered in the breakpoint dialog by using a “-” (minus) between start and end address:
Please note that when a read or write access to an address within the range is performed by the application the instruction where the application stops is several instructions behind the access. This is a hardware restriction of data range break.

Please note that data range break is always aligned to 16bit boundary (even address) to preserve the number of breakpoints. Allowing full range alignment the number of internal breakpoint registers would consume up to 3.

5.4.2 Code Range Break

To set a code range break the syntax is the same as described above. Please check Execute and uncheck Read and Write in the breakpoint dialog if necessary.

Important note:

The start and end address must be aligned to an instruction. Otherwise the code range hardware hit is not performed properly and undefined behavior of EUROScope may result.

Also note that once the code range break was hit, the according range break is disabled (indicated by white circle in breakpoint window). Otherwise single stepping within the range break is not possible.

5.5 Number of available Breakpoints

The number of available breakpoints can be seen in Preferences → Configure Target Connection:
Please note, that single stepping consumes at least one break point, so that the individual settable user breakpoints never reach the maximum number of available breakpoints. Single stepping is always active when application is halted.

### 5.6 Stepping through Application

Now we want to step an instruction further. Therefore the menu button \( \text{F11} \) or “F11” can be used.

```
22 | local_cnt = 00000;
24 |     while(1)
25 |     { wait(local_cnt);
26 |         PDIRC4++;  // counter...
29 |     }
30 |
```

What has happened here? The application seems to have stepped somehow out of the `while`-loop.

Please note that in the source code window some code lines are grouped. In this case line 23, 25 and 27 are grouped. With this grouping the lines are treated as one single “breakable” line.

Having a look in the assembly window would make the picture clearer.
The compiler optimization did not choose a real local stack variable but loads the argument for `wait` immediately before calling. By marking a line in the assembly window, it is possible to step by single assembly instruction.

The next single step stops in the `wait` function.

```c
void wait (unsigned long a)
{
    for(cnt = 0; cnt < a; cnt++); // Wait
}
```

Please also have a look at the assembly window. A green triangle indicates the return address from the subroutine `wait`.

If we now use step-in again, we would get trapped, because the `wait` function code is executed 80000 times. Therefore the step-out or “Run until current function is left” (“Shift-F11”) can be used. In this case the sub function is executed completely and the next stop is out of this function.

Note that this does not work for ISR until EUROScope version 3.0.3 inclusive.
If sub function is not required to be called in single steps, the step-over functionality can be used \( F10 \) ("F10") instead.

The “Run until cursor” functionality \( F4 \) ("F4") executes the application until the cursor position is reached. This temporary break point is indicated with a white-filled red circle.

```
local_cnt = 0000;
while(1) {
  wait(local_cnt);
  DDR00++;  // counter...
}
```

Figure 5-17: EUROScope – Run until

**Important Note:**

In EUROScope versions < 3.03 it may happen in some cases, that CPU registers are modified during single stepping. Please do not use these versions.

### 5.7 Source Code Window Handling

It is possible to have more than one source code window since EUROScope version 3.00. To open a new module move the mouse to the tab area and choose New Source window from the menu list.

Figure 5-18: EUROScope – New Source Window Dialog

Afterwards a new tab and a new (empty) source code window will occur. Then move the mouse into the window area and choose show module by right clicking and browse to the module you want to be displayed.
16FX- Setup and Debugging with EUROScope
Breakpoints and Program Stepping

Then select to the desired source module file:

Please also refer section 15.1 for the breakpoint behavior after reset.

A quick way to show a new source window (since EUROScope version 3.00) is to use the menu View → New Module View.
Note:
It is recommended to check *Follow PC* at the left most source window. In this case the contents will change to an actual stopped module, if this module is not opened in other tabs.

![EUROScope - Follow PC](image)

*Figure 5-22: EUROScope – Follow PC*
6 Variable Watching and Manipulating

HOW TO MANIPULATE CPU REGISTERS, VARIABLES AND MEMORY

6.1 General

With EUROScope all windows with changeable contents are refreshed when stopping the application by breakpoint or stop button. All changed values prior to the last state are displayed in red text, so that it is easy for the user to track changes.

6.2 Watching and Manipulating Processor Status

The CPU flags can be found at the bottom of the register view window. A flag can be changed by double-clicking to it.

![Figure 6-1: EUROScope – Processor Status](image)

In this example screen shot the carry flag was changed from 0 to 1.

The flags are:

<table>
<thead>
<tr>
<th>Abbr.</th>
<th>Flag name</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Interrupt enable flag (1 = enable)</td>
</tr>
<tr>
<td>S</td>
<td>Stack flag (0 = User stack; 1 = System stack)</td>
</tr>
<tr>
<td>T</td>
<td>Sticky bit flag (1 = shift right instruction executed)</td>
</tr>
<tr>
<td>N</td>
<td>Negative flag (MSB = 1 in last operation)</td>
</tr>
<tr>
<td>Z</td>
<td>Zero flag (Last operation resulted in &quot;0&quot;)</td>
</tr>
<tr>
<td>V</td>
<td>Overflow flag (Overflow at last operation)</td>
</tr>
<tr>
<td>C</td>
<td>Carry flag (Last operation caused carry)</td>
</tr>
</tbody>
</table>

![Table 6-1: Processor Status](image)

6.3 Watching and Manipulating CPU Registers

The CPU registers can be observed and changed in the register view window. To change a register, double-click to it. A pop-up window will be opened. Here the new contents can be set. The following screen shot shows a change in RW0:

![Figure 6-2: EUROScope – CPU Registers](image)
The values are always hexadecimal. Please be careful when changing registers, so that the application state at a break condition is not destroyed.

The registers are:

<table>
<thead>
<tr>
<th>Abbr.</th>
<th>Register name</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Accumulator</td>
</tr>
<tr>
<td>AL</td>
<td>Accumulator lower word</td>
</tr>
<tr>
<td>AH</td>
<td>Accumulator upper word</td>
</tr>
<tr>
<td>USP</td>
<td>User Stack Pointer</td>
</tr>
<tr>
<td>SSP</td>
<td>System Stack Pointer</td>
</tr>
<tr>
<td>ILM</td>
<td>Interrupt Level Mask (belongs to PS)</td>
</tr>
<tr>
<td>RP</td>
<td>Register Bank Pointer (belongs to PS)</td>
</tr>
<tr>
<td>CCR</td>
<td>Condition Code Register (belongs to PS)</td>
</tr>
<tr>
<td>DTB</td>
<td>Data Bank Register</td>
</tr>
<tr>
<td>ADB</td>
<td>Additional Data Bank Register</td>
</tr>
<tr>
<td>DPR</td>
<td>Direct Page Register</td>
</tr>
<tr>
<td>PC</td>
<td>Program Counter</td>
</tr>
<tr>
<td>R0-R7</td>
<td>General Purpose Registers</td>
</tr>
<tr>
<td>RW0-RW7</td>
<td></td>
</tr>
<tr>
<td>RL0-RL3</td>
<td></td>
</tr>
</tbody>
</table>

Table 6-2: CPU Registers

6.4 Watching and Manipulating C Variables

Global variables are automatically added to the variable window (at EUROScope professional version). Local variables and function arguments appear within their dedicated sub functions. The variable contents are updated after every execution stop of the application.

In the EUROScope Lite version the number of global variables in the “Watch” tab is limited to 8. Global variables are not displayed in the “Global” tab, but all local variables are displayed in “Local” tab. In the EUROScope professional version these limitations do not exist.

The following table gives an overview of the variable features of the EUROScope versions:

<table>
<thead>
<tr>
<th>Variable Tab</th>
<th>EUROScope Lite</th>
<th>EUROScope Professional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local</td>
<td>all</td>
<td>all</td>
</tr>
<tr>
<td>Global</td>
<td>n. a.</td>
<td>all</td>
</tr>
<tr>
<td>Watch</td>
<td>all Locals</td>
<td>all Locals</td>
</tr>
<tr>
<td></td>
<td>8 Globals</td>
<td>all Globals</td>
</tr>
</tbody>
</table>

Table 6-3: EUROScope – C Variables

Additionally variables can be added individually to the “Watch” tab view.

Changing a variable content can be done by clicking to the shown value. The value can then be entered.

The variable value view can be set to decimal or hexadecimal. This can be chosen by right-clicking and checking or un-checking “Show ints as decimal” option.
6.5 Watching and Manipulating Memory

6.5.1 Change Contents

Memory contents can be observed in the memory view window. The contents are updated after every execution stop of the application.

Assume the value for the global variable \( \text{cnt} \) should be changed. In the global variable tab (or watch tab) view we can see that \( \text{cnt} \) is located at address \( \text{0x4240} \). To see this address in the memory window right-click into it and choose “Show Address ...” In the pop-up window the address \( \text{0x4240} \) can be entered then.

Now the memory window shows the address contents beginning at \( \text{0x4240} \). Just click at the content of this address and enter e.g. \( \text{0x1234} \) in the pop-up “Edit memory” window.

Have a look at the variable window. The content of \( \text{cnt} \) did also change.

The content view of the memory window can be chosen 8-bit, 16-bit or 32-bit wide.
6.5.2 Fill Memory

Right-clicking into the memory window shows the Fill menu. Choosing this a memory area can be filled with a given Byte.

6.6 Automatic Local Variable Watch View

On some PC systems (EUROScope Version < 3.01) the automatic local variable watch shows no contents. In this case reload abs file after open communication.
7 Function Browser

HOW TO USE THE FUNCTION BROWSER VIEW

7.1 General
Note that this view is only available in the EUROScope professional version.

7.2 Opening Function Browser
To open the function browser view choose menu View → New Function Browser…. Now a new dock-able window occurs with contains three tabs: Labels, Functions, and Modules. In these views a click to a function results in an update in the source and assembly view to the clicked function.

The labels view contains all labels and function names. The function view shows all c module functions and the modules view contains all functions and labels with respect to the modules itself.
8 SFR View

HOW TO USE THE SPECIAL FUNCTION REGISTER VIEW

8.1 General
Note that this view is only available in the EUROScope professional version.

8.2 Opening SFR View
The data for the SFR view has to be loaded before usage. Therefore choose menu View → New SFR… . If this window is chosen first, it will be empty. To load the SFR file for Fujitsu 16FX series, right click and choose Load SFR file…. Then browse to the location of the SFR directory (normally the SFR directory of the EUROScope installation path) and choose the file 96340 Eva2.sfr. After this the SFR view contains the top level register groups.

<table>
<thead>
<tr>
<th>Register</th>
<th>Address</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clocks</td>
<td>0x401</td>
<td></td>
<td>Clock Control Registers</td>
</tr>
<tr>
<td>RCB</td>
<td>0x070000</td>
<td></td>
<td>ROM Configuration Block</td>
</tr>
<tr>
<td>PATCH</td>
<td>0x5810</td>
<td></td>
<td>Memory Patch Function</td>
</tr>
<tr>
<td>I/O Ports</td>
<td>0x0</td>
<td></td>
<td>I/O Ports</td>
</tr>
<tr>
<td>ADC</td>
<td>0x18</td>
<td></td>
<td>Analog to Digital Converter</td>
</tr>
<tr>
<td>I/O Timer</td>
<td>0x20</td>
<td></td>
<td>I/O Timer</td>
</tr>
<tr>
<td>I/O Timer</td>
<td>0x24</td>
<td></td>
<td>I/O Timer</td>
</tr>
<tr>
<td>I/O Timer</td>
<td>0x500</td>
<td></td>
<td>I/O Timer</td>
</tr>
<tr>
<td>I/O Timer</td>
<td>0x584</td>
<td></td>
<td>I/O Timer</td>
</tr>
<tr>
<td>External Int.</td>
<td>0x58</td>
<td></td>
<td>External Interrupts</td>
</tr>
<tr>
<td>RLT</td>
<td>0x60</td>
<td></td>
<td>Reload Timer</td>
</tr>
<tr>
<td>PPG</td>
<td>0x74</td>
<td></td>
<td>Programmable Pulse Generator</td>
</tr>
<tr>
<td>I2Cx</td>
<td>0x4AC</td>
<td></td>
<td>Inter Integrated Circuit Bus</td>
</tr>
<tr>
<td>LIN USART</td>
<td>0x5CB</td>
<td></td>
<td>LIN USART</td>
</tr>
<tr>
<td>DMA</td>
<td>0x100</td>
<td></td>
<td>Direct Memory Access</td>
</tr>
<tr>
<td>GPH_FAM</td>
<td>0x180</td>
<td></td>
<td>CPU · General Purpose registers (RAM access)</td>
</tr>
<tr>
<td>Interrupts</td>
<td>0x3A0</td>
<td></td>
<td>Interrupts</td>
</tr>
<tr>
<td>ROM Mirror</td>
<td>0x3AE</td>
<td></td>
<td>ROM Mirroring Module</td>
</tr>
<tr>
<td>Memory Fas...</td>
<td>0x3AF</td>
<td></td>
<td>Memory Patch Function</td>
</tr>
</tbody>
</table>

Figure 8-1: EUROScope – SFR View

8.3 Using SFR View
To change SFR contents “browse” to the desired register and set a new value by double-clicking to the current value entry. The next screen shot shows for example setting a new value to Port00:

<table>
<thead>
<tr>
<th>Register</th>
<th>Address</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PATCH</td>
<td>0x380</td>
<td></td>
<td>Memory Patch Function</td>
</tr>
<tr>
<td>I/O Ports</td>
<td>0x0</td>
<td></td>
<td>I/O Ports</td>
</tr>
<tr>
<td>Port 00</td>
<td>0x0</td>
<td></td>
<td>I/O Port 00</td>
</tr>
<tr>
<td>DFR</td>
<td>0x0</td>
<td>0x0</td>
<td>Port Data Register</td>
</tr>
<tr>
<td>DDR</td>
<td>0x400</td>
<td>0x0F</td>
<td>Port Data Directions Register</td>
</tr>
<tr>
<td>PIER</td>
<td>0x454</td>
<td>0x0D</td>
<td>Port Input Enable Register</td>
</tr>
<tr>
<td>FILLR</td>
<td>0x58</td>
<td>0x0D</td>
<td>Port Input Level Register</td>
</tr>
</tbody>
</table>

Figure 8-2: EUROScope – Using SFR View

After change, the new value is displayed in red text.

Please note, that the SFR view only shows the read-out values of a register – not obligatory the written-to value. Example: Baud rate registers of USARTs.
9 Callstack View

HOW TO READ THE CALLSTACK VIEW

9.1 General
Note that this functionality is only available in the EUROScope professional version.

9.2 Opening the Callstack View
Choose View → New Callstack… to open the Callstack window.

9.3 Using Callstack View
Assume a breakpoint is set at the function wait and the application is started. After reaching the breakpoint the following Callstack view will occur:

![Callstack View](image1)

Figure 9-1: EUROScope – Callstack View

The yellow arrow indicates the actual program counter location – here the function `wait()`. After the entry it is shown how many bytes the program counter is away from the top of the function. Here it is 2 bytes. Then the module name and the absolute address follow. The next line shows the caller function `_main` and the location in bytes from where `wait` was called. The last entry shows the last label of the STARTUP module code (`end`).

If the line of the `_main` function is double-clicked, the following view will occur:

![Callstack View II](image2)

Figure 9-2: EUROScope – Callstack View II
Then the source window will display the return location from `wait`, indicated by a green triangle. Please remember that in the source code window the triangle is displayed at the top of grouped instruction lines.
10 Command Line Interpreter

HOW TO USE THE COMMAND LINE INTERPRETER

10.1 General
Note that this functionality is only available in the EUROScope professional version.

10.2 Opening the Command Line Interpreter
Choose View → New CommandLine….

The command help shows an overview of all usable commands. Please refer to EUROScope manual for detailed command explanation.
11 Terminal Window

HOW TO USE THE TERMINAL WINDOW

11.1 General
Note that this functionality is only available in the EUROScope professional version.

11.2 Opening Terminal Window
Choose the menu View → New Terminal to open the terminal window. It can be docked to the workspace or remain flowing.

By clicking with the right mouse button and choosing Configure, the terminal window can be configured for COM port number, baudrate, etc.
12 Flashing an Application in Internal/External Vector Mode

HOW TO REFRESH AN APPLICATION IN INTERNAL/EXTERNAL VECTOR MODE

12.1 Re-Flashing Application

Once the application is flashed in the target and the communication is established, EUROScope (since version 3.00) is able to re-flash the application by itself. The mode pin state is the same as for debugging, so it can stay at internal vector/normal run mode:

<table>
<thead>
<tr>
<th>MD0</th>
<th>MD1</th>
<th>MD2</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>H</td>
<td>L</td>
</tr>
</tbody>
</table>

Table 12-1: Mode Switch Setting

Please also be sure that the communication is still in internal/external vector mode:

Therefore the Flash dialog should be opened by clicking on the Flash button:

The Unlock/BDM dialog described in 2.4 will open (if Do not show again was not checked before). Please give EUROScope then some few seconds for downloading the Flash kernel into the MCU RAM.

Afterwards the flash dialog will occur as follows:
Please make sure that all sectors box is checked. Then the MCU is erased by clicking on Erase button. It will take then some seconds for the chip erase. A growing and disappearing progress bar indicates, that chip erasing is ongoing.

Afterwards the application MHX file can be chosen and programmed to the MCU by clicking on Program button.

At the end, please click on Done button.

A pop-up window will remind you, that you also have to force a hardware reset to your target system.

The ABS file of your application is automatically loaded afterwards.

12.2 Downloading Application

Alternatively Download to Target or Open Application and download to Target can be chosen.
The first should be chosen, if the debugging settings should not be changed, but the application is rebuilt.

The seconds method should be chosen, if a completely new application should be debugged.

12.3 Smart Download

This function (for professional version) is not implemented yet.

Please note that a **hardware reset** has also to be forced when using *Smart Download*. The mechanism is the same as described above.

12.4 Breakpoints after Flashing

Please note, that after the Flashing process all former set breakpoints are removed. This is for safety reasons, if the addresses of instructions move with the updated project and point to addresses, which are not aligned to instructions anymore.
13 Flashing an Application in Serial Communication Mode

HOW TO REFLASH AN APPLICATION IN SERIAL COMMUNICATION MODE

13.1 (Re-)Flashing Application

Since EUROScope version 3.01 it supports also Flash programming in serial communication mode (commonly known as Flash programming mode). The mode pin settings are the same like for the Fujitsu Flash Programmer:

<table>
<thead>
<tr>
<th>MD0</th>
<th>MD1</th>
<th>MD2</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>H</td>
<td>L</td>
</tr>
</tbody>
</table>

Table 13-1: Mode Switch Setting

To establish the serial communication please switch mode pins described in the table above and reset MCU externally.

Then choose menu from EUROScope menu bar: Preferences → Configure Target Connection. Afterwards select Serial communication mode in Communication options.

![EUROScope Connection Configuration Dialog](image)

Figure 13-1: EUROScope – Connection Configuration Dialog

Click OK. Afterwards the Flash programming is the same described in 12.1.

Important Note:

Do not forget to set communication mode back to internal/external vector mode after Flashing, to set back the mode pins, and to reset MCU before starting a new debug session.
14 Ending a Debug Session

WHAT HAS TO BE REGARDED

14.1 Communication Closure

If the MCU is in run mode the communication can be closed with the MCU remain running. The communication can be re-established. Please note that this (re-establishing the communication) will force an immediate stop of the application (Same behavior if automatic start is configured).

14.2 Save Configuration

Before ending EUROSope the current configuration should be saved with the menu Preferences → Save Configuration [as]. The saved cfg-file then contains all settings of the workspace. If this is not done, all settings and watch variable collection will be lost in the next debug session. The last configuration is loaded at debug session start. If a new project was debugged a new configuration file should be saved.

14.3 New Build during Debug Session

If the application is built again during a debug session, EUROSope does not need to be closed.

EUROSope version < 3.00: All that is needed is close the communication, re-flash the application with the Fujitsu 16FX Flash Programmer and reload the abs file in EUROSope, reset target and open communication again.

EUROSope version ≥ 3.00: All that is needed is to re-flash the application with EUROSope and reset the target. The abs file is reloaded automatically.
15 Notes on using EUROScope

NOTES ON USING EUROSCOPE

15.1 Reset Target

Please note that the button “Reset Target” only performs the software reset within the MCU. The hardware reset functionality is not implemented yet.

The behavior related to existent breakpoints, after pressing the “Reset Target” button is dependent on the Automatic Start bit (BDMCM:STRT) configuration.

If the automatic start is enabled (bit BDMCM:STRT = 1) then after “Reset Target” button is pressed all the breakpoint gets cleared (i.e. Memory Patch Function Control/Status Registers – PFACSn gets initialized to 0x0000 because of the software reset) and then the application starts running. Once the user presses “Stop Running Application” button then the application stops executing and the breakpoints are restored.

If the automatic start is disabled (bit BDMCM:STRT = 0) then after “Reset Target” button is pressed all the breakpoint gets cleared (i.e. Memory Patch Function Control/Status Registers – PFACSn gets initialized to 0x0000 because of the software reset) and then the application halts at the address pointed by default reset vector (0xFFFFDC) and then the breakpoints are restored.

It should be noted that the above discussed behavior is valid for EUROScope versions 2.1.06 and higher.

15.2 Serial Connection

Sometimes it is necessary to disconnect the target temporary and reconnect again, if the debug protocol is out of synchronization.

15.3 Flash Programming

Flash programming with EUROScope is possible since version 3.00, but there is a restriction when writing the command to the Flash I/F. In this case single stepping or breaking in the command write instruction and the polling after it is not possible, because Flash read out is not possible. The Boot ROM reloads its configuration during internal break and thus will get wrong information about used USART and baud rate. There is no way to get EUROScope back to working except reset.

Please note, that these restrictions are not valid for Flash B or Data Flash. It is only relevant for Main Flash (A) where the debug configuration block is located.

The following code shows the area in a red box, where setting breakpoints and stepping through are not allowed.
15.3.1 Source Code

```c
unsigned char Main_Flash_sector_erase(volatile __far unsigned int *sec_adr)
{
    unsigned char flag = 0;
    unsigned char MFMCS_save;
    unsigned int MFMTMC_save;

    // preparations
    MFMCS_save = MFMCS;    // save Flash settings
    MFMTMC_save = MFMTMC;
    MFMCS_CRBE = 0;        // disable Code Read Buffer
    MFMCS_DRBE = 0;        // disable Data Read Buffer
    MFMTMC = 0x4B3D;       // slow down Flash access to 4 wait states
    MFMCS_WE = 1;          // set write enable flag

    // start with Flash sector erase sequence
    *seq_AAAA = 0x00AA;    // sends erase command to the pointed address
    *seq_5554 = 0x0055;
    *seq_AAAA = 0x0080;
    *seq_AAAA = 0x00AA;
    *seq_5554 = 0x0055;
    *sec_adr  = 0x0030;
    while ((*sec_adr & DQ3) != DQ3);  // sector erase timer ready?

    while(flag == 0)
    {
        if((*sec_adr & DQ7) == DQ7)     // data polling
        {
            flag = 0x10;                  // successful erased!
        }
        else
        {
            flag = 0x40;                // time out error!
        }
    }

    MFMCS_WE = 0;          // reset write enable flag
    MFMTMC = MFMTMC_save;    // restore Flash settings
    MFMCS = MFMCS_save;
    return(flag);
}
```

No breakpoints or single stepping allowed!
### 15.3.2 Assembly Code

```
0042FD: 0804          LINK  #04
0042FF: 4F03          PUSHW RW0,RW1
004301: 0D             MOVN A,#0
004302: 7B3FC         MOV @RW3-04,A
004305: 52F103        MOV A,#0F1
004306: 7283FD        MOV @RW3-03,A
00430B: 5AF203        MOVW A,#0F2
00430E: CBFE          MOVW @RW3-02,A
004310: 6C5CF103      CLR8 RW3:14
004314: 6C5DF103      CLR8 RW3:15
004318: 73DF2033D4B   MOVW 03F2,#4B3D
00431E: 4B5405DF00    MOVL A,#0DF0554
004323: 71A0          MOVL RL0,A
004325: 4255          MOV A,#55
004327: 6F3800        MOVW @RL0+00,A
00432E: 4BAA0ADF00    MOVL A,#00DF0AAA
004333: 71A0          MOVL RL0,A
004335: 42F1          MOV A,#F1
004337: 6F3800        MOVW @RL0+00,A
004343: 4BAAADF00     MOVL A,#0DF0AAA
00434A: 71A0          MOVL RL0,A
00434C: 42AA          MOV A,#AA
00434D: 6F3800        MOVW @RL0+00,A
004352: 4B5405DF00    MOVL A,#0DF0554
004357: 71A0          MOVL RL0,A
004359: 4255          MOV A,#55
00435B: 6F3800        MOVW @RL0+00,A
00435E: 719306        MOVL A,#9306
004361: 71A0          MOVL RL0,A
004363: 4290          MOV A,#90
004365: 6F3800        MOVW @RL0+00,A
004368: 719306        MOVL A,#9306
00436B: 71A0          MOVL RL0,A
00436D: 6F4800        MOVW A,#RL0+00
004370: D8             MOVN A,#8
004371: 2C             ANDW A
004372: D8             MOVN A,#8
004373: 2E             CMPW A
004374: F002          BZ  004378
004376: 60F0          BRA 004368
004378: 7293FC        MOV A,#9303-04
00437B: F13D          BNZ 0043BA
00437D: 719306        MOVL A,#9306
004380: 71A0          MOVL RL0,A
004382: 6F4800        MOVW A,#RL0+00
004385: 730000        MOVW A,#0080
004387: 3A0000        ANDW A,#0080
004388: 3B0000        MOVN A,#0080,004390
00438C: 7193FC10      MOV @RW3-04,#10
004390: 719306        MOVL A,#9306
004393: 71A0          MOVL RL0,A
004395: 6F4800        MOVW A,#RL0+00
004398: 3C2000        ANDW A,#0020
00439B: 3A00019       MOVN A,#0020,0043BB
00439F: 719306        MOVL A,#9306
0043A2: 71A0          MOVL RL0,A
0043A4: 6F4800        MOVW A,#RL0+00
0043A7: 3C3000        ANDW A,#0F80
0043AA: 3A00006       MOVN A,#0F80,0043B4
0043AE: 71D3FC20      MOV @RW3-04,#20
0043B2: 6004          BRA 004388
0043B4: 71D3FC40      MOV @RW3-04,#40
0043BE: 60BE          BRA 004378
0043BA: 6C5BF103      CLR8 RW3:15
0043BE: 5BF203        MOV A,#4F2,0A
0043C0: 7293FD        MOV A,#9303-03
0043C3: 7293FC        MOV A,#9303-04
0043C9: 5F03          POPW RW0,RW1
0043C8: 09             UNLINK
0043CF: 66             RETP
```

No breakpoints or single stepping allowed!
15.4 USB support (SK-16FX-100PMC Board)

Using SK-16FX-100PMC board it is possible to use USB interface (which acts as a virtual COM port) for the communication between the target i.e. SK-16FX-100PMC and the host i.e. EUROScope. This communication can be speed-up by decreasing the latency time of the USB driver. This can be done as follows:

Go to Windows Start Menu: Settings → Control Panel → System. Then use the Hardware tab and click on Device Manager.

![Device Manager](image)

Figure 15-1: USB Support – Device Manager

Then browse to Ports (COM&LPT) and search for FUJITSU Microcontroller Board (COM\n), where \n is the virtual COM port number, which the installer has enumerated. Double click to the entry and choose tab Port Settings and then Advanced.
To speed up the communication choose the value “1” at the BM Options → Latency Timer (msec). Click on OK on the both open dialog boxes.

Note the COM Port Number option can be used to change the COM port number, if EUROScope may have problems to configure the same (that means if the particular COM port number does not appear in the dialog box shown in Figure 3-7). So finally one needs to make sure that the COM port number is same at both these places.
16 Appendix A

DEVICE RESTRICTIONS AND FEATURES

16.1 Devices which do not support Background Debugging
The following devices do not support background debugging.

<table>
<thead>
<tr>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>MB96F348HxA</td>
</tr>
<tr>
<td>MB96F348TxA</td>
</tr>
</tbody>
</table>

Table 16-1: Devices with no BDM functionality

16.2 Maximum PLL Factor Device List
The following table shows several derivatives and their maximum PLL factors, which has to be considered for the baud rate calculation (also refer section 2.2.3). Please always check dedicated hardware manual and data sheet for maximum PLL factor especially if the used device is not listed here.

<table>
<thead>
<tr>
<th>Device</th>
<th>Maximum PLL Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>MB96F346AxA</td>
<td>16</td>
</tr>
<tr>
<td>MB96F346RxA</td>
<td>16</td>
</tr>
<tr>
<td>MB96F346YxA</td>
<td>16</td>
</tr>
<tr>
<td>MB96F347AxA</td>
<td>16</td>
</tr>
<tr>
<td>MB96F347RxA</td>
<td>16</td>
</tr>
<tr>
<td>MB96F347YxA</td>
<td>16</td>
</tr>
<tr>
<td>MB96F348AxA</td>
<td>16</td>
</tr>
<tr>
<td>MB96F348RxA</td>
<td>16</td>
</tr>
<tr>
<td>MB96F348YxA</td>
<td>16</td>
</tr>
<tr>
<td>All others</td>
<td>32</td>
</tr>
</tbody>
</table>

Table 16-2: Device Specific Maximum PLL Factor

16.3 Breakpoint at PLL Register (PLLCR) Access
The following table shows devices on which it is not possible to set a breakpoint to an instruction, which changes PLL settings. This includes single stepping. In this case a wrong debugging USART baud rate is calculated by the Boot-ROM.

Countermeasure: Set breakpoint behind PLLCR access instruction and use run.

<table>
<thead>
<tr>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>MB96F346RxA</td>
</tr>
<tr>
<td>MB96F346YxA</td>
</tr>
<tr>
<td>MB96F347RxA</td>
</tr>
<tr>
<td>MB96F347YxA</td>
</tr>
<tr>
<td>MB96F348RxA</td>
</tr>
<tr>
<td>MB96F348YxA</td>
</tr>
</tbody>
</table>

Table 16-3: Device Specific PLL Register Access Restriction
### 16.4 Supported USARTs

<table>
<thead>
<tr>
<th>Logical USART</th>
<th>Physical USART</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>7'</td>
</tr>
<tr>
<td>B</td>
<td>8'</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 16-4: MB96320 USART mapping

<table>
<thead>
<tr>
<th>Logical USART</th>
<th>Physical USART</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 16-5: MB96330 USART mapping

<table>
<thead>
<tr>
<th>Logical USART</th>
<th>Physical USART</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 16-6: MB96340 USART mapping

<table>
<thead>
<tr>
<th>Logical USART</th>
<th>Physical USART</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>7'</td>
</tr>
<tr>
<td>B</td>
<td>8'</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 16-7: MB96350 USART mapping

<table>
<thead>
<tr>
<th>Logical USART</th>
<th>Physical USART</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 16-8: MB96370 USART mapping

1. Not supported by MB96F326RSA, MB96F326RWA, MB96F326YSA, MB96F326YWA.
2. The following devices do not support EUROScope: MB96F348HSA, MB96F348HWA, MB96F348TSA, MB96F348TWA. See also 16.1.
3. Not supported by MB96F356RSA, MB96F356RWA, MB96F356YSA, MB96F356YWA.
<table>
<thead>
<tr>
<th>Logical USART</th>
<th>Physical USART</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Table 16-9: MB96380 USART mapping

<table>
<thead>
<tr>
<th>Logical USART</th>
<th>Physical USART</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Table 16-10: MB96390 USART mapping
### LITE/FULL VERSION DIFFERENCES OVERVIEW

#### 17.1 Lite/Professional Differences Table and Implementation since Version 3.0x

<table>
<thead>
<tr>
<th>Function</th>
<th>Comment</th>
<th>Package</th>
<th>Implementation (V3.00)</th>
<th>Implementation (V3.01)</th>
<th>Implementation (V3.02)</th>
<th>Implementation (V3.03)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verify Program Image</td>
<td></td>
<td>Prof.</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Reserve channels for application</td>
<td></td>
<td>Prof.</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Callstack</td>
<td></td>
<td>Prof.</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>EUROS objects</td>
<td>if EUROS installed</td>
<td>Prof.</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>EUROStrace</td>
<td>if EUROS installed</td>
<td>Prof.</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>SFR window</td>
<td></td>
<td>Prof.</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Variables (local) window</td>
<td></td>
<td>Lite</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Variables (local/glob*) window</td>
<td></td>
<td>Prof.</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Command Line interpreter</td>
<td></td>
<td>Prof.</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>ORTI objects</td>
<td>if OSEK installed</td>
<td>Prof.</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Terminal Window</td>
<td></td>
<td>Prof.</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Variable Monitoring</td>
<td></td>
<td>Prof./Lite</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Variable Tracking</td>
<td></td>
<td>Prof.</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>

* limited to 8 in Lite version.

Table 17-1: Lite/Profession version differences
18 Related Application Notes and Contact

OVERVIEW OF RELATED APPLICATION NOTES AND CONTACT TO FUJITSU

18.1 Manuals, Application Notes and Customer Information

- EUROScope Reference Manual
- Customer Information: CI-300102-E-V10-EUROScope_V3_0_restrictions
- Customer Information: CI-300103-E-V10-EUROScope_V3_0_1_restrictions
- Customer Information: CI-300108-E-V10-EUROScope_V3_0_2_restrictions
- Customer Information: CI-300109-E-V10-EUROScope_V3_0_3_restrictions

18.2 Contact to Fujitsu

Websites:
http://emea.fujitsu.com/microelectronics
http://mcu.emea.fujitsu.com/

Contact:
micro_info@fme.fujitsu.com
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1-1</td>
<td>EUROS License Key</td>
<td>8</td>
</tr>
<tr>
<td>Figure 1-2</td>
<td>EUROS Software Registration</td>
<td>9</td>
</tr>
<tr>
<td>Figure 2-1</td>
<td>Internal Vector mode operation</td>
<td>11</td>
</tr>
<tr>
<td>Figure 2-2</td>
<td>BDM Dialog</td>
<td>18</td>
</tr>
<tr>
<td>Figure 2-3</td>
<td>Hardware Considerations</td>
<td>20</td>
</tr>
<tr>
<td>Figure 3-1</td>
<td>Flash Programmer</td>
<td>21</td>
</tr>
<tr>
<td>Figure 3-2</td>
<td>EUROScope</td>
<td>22</td>
</tr>
<tr>
<td>Figure 3-3</td>
<td>EUROScope – Loading Configuration</td>
<td>22</td>
</tr>
<tr>
<td>Figure 3-4</td>
<td>EUROScope – Loading ABS File</td>
<td>23</td>
</tr>
<tr>
<td>Figure 3-5</td>
<td>EUROscope – Progress Bar</td>
<td>23</td>
</tr>
<tr>
<td>Figure 3-6</td>
<td>EUROscope – Select Target Connection</td>
<td>23</td>
</tr>
<tr>
<td>Figure 3-7</td>
<td>EUROscope – Configure Target Connection</td>
<td>24</td>
</tr>
<tr>
<td>Figure 3-8</td>
<td>Device Manager</td>
<td>24</td>
</tr>
<tr>
<td>Figure 3-9</td>
<td>EUROscope – Source Path</td>
<td>25</td>
</tr>
<tr>
<td>Figure 3-10</td>
<td>EUROscope – Debugging Session</td>
<td>26</td>
</tr>
<tr>
<td>Figure 4-1</td>
<td>EUROscope – Initialize Target</td>
<td>27</td>
</tr>
<tr>
<td>Figure 5-1</td>
<td>EUROscope – Breakpoints</td>
<td>29</td>
</tr>
<tr>
<td>Figure 5-2</td>
<td>EUROscope – Grouped Lines</td>
<td>29</td>
</tr>
<tr>
<td>Figure 5-3</td>
<td>EUROscope – Setting Breakpoint</td>
<td>29</td>
</tr>
<tr>
<td>Figure 5-4</td>
<td>EUROscope – Deactivating Breakpoint</td>
<td>29</td>
</tr>
<tr>
<td>Figure 5-5</td>
<td>EUROscope – Breakpoint Assembly View</td>
<td>30</td>
</tr>
<tr>
<td>Figure 5-6</td>
<td>EUROscope – Setting Breakpoint by Menu Dialog</td>
<td>30</td>
</tr>
<tr>
<td>Figure 5-7</td>
<td>EUROscope – Setting Breakpoint by Menu Dialog II</td>
<td>31</td>
</tr>
<tr>
<td>Figure 5-8</td>
<td>EUROscope – Setting Data Breakpoint by Menu Dialog</td>
<td>31</td>
</tr>
<tr>
<td>Figure 5-9</td>
<td>EUROscope – Using Breakpoints</td>
<td>32</td>
</tr>
<tr>
<td>Figure 5-10</td>
<td>EUROscope – Data Range Break Dialog</td>
<td>33</td>
</tr>
<tr>
<td>Figure 5-11</td>
<td>EUROscope – Code Range Break Dialog</td>
<td>33</td>
</tr>
<tr>
<td>Figure 5-12</td>
<td>EUROscope – Available HW Breakpoints</td>
<td>34</td>
</tr>
<tr>
<td>Figure 5-13</td>
<td>EUROscope – Stepping through Application</td>
<td>34</td>
</tr>
<tr>
<td>Figure 5-14</td>
<td>EUROscope – Assembly Window</td>
<td>35</td>
</tr>
<tr>
<td>Figure 5-15</td>
<td>EUROscope – Single Step</td>
<td>35</td>
</tr>
<tr>
<td>Figure 5-16</td>
<td>EUROscope – Single Step Assembly View</td>
<td>35</td>
</tr>
<tr>
<td>Figure 5-17</td>
<td>EUROscope – Run until</td>
<td>36</td>
</tr>
<tr>
<td>Figure 5-18</td>
<td>EUROSCope – New Source Window Dialog</td>
<td>36</td>
</tr>
<tr>
<td>Figure 5-19</td>
<td>EUROscope – Show Module I</td>
<td>37</td>
</tr>
<tr>
<td>Figure 5-20</td>
<td>EUROscope – Show Module II</td>
<td>37</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>5-21</td>
<td>EUROScope – New Module View</td>
<td>37</td>
</tr>
<tr>
<td>5-22</td>
<td>EUROScope – Follow PC</td>
<td>38</td>
</tr>
<tr>
<td>6-1</td>
<td>EUROScope – Processor Status</td>
<td>39</td>
</tr>
<tr>
<td>6-2</td>
<td>EUROScope – CPU Registers</td>
<td>39</td>
</tr>
<tr>
<td>6-3</td>
<td>EUROScope – Global Variables</td>
<td>40</td>
</tr>
<tr>
<td>6-4</td>
<td>EUROScope – Global Variables II</td>
<td>41</td>
</tr>
<tr>
<td>6-5</td>
<td>EUROScope – Memory</td>
<td>41</td>
</tr>
<tr>
<td>6-6</td>
<td>EUROScope – Global Variables III</td>
<td>41</td>
</tr>
<tr>
<td>6-7</td>
<td>EUROScope – Memory Fill</td>
<td>42</td>
</tr>
<tr>
<td>7-1</td>
<td>EUROScope – Function Browser</td>
<td>43</td>
</tr>
<tr>
<td>8-1</td>
<td>EUROScope – SFR View</td>
<td>44</td>
</tr>
<tr>
<td>8-2</td>
<td>EUROScope – Using SFR View</td>
<td>44</td>
</tr>
<tr>
<td>9-1</td>
<td>EUROScope – Callstack View</td>
<td>45</td>
</tr>
<tr>
<td>9-2</td>
<td>EUROScope – Callstack View II</td>
<td>45</td>
</tr>
<tr>
<td>10-1</td>
<td>EUROScope – Command Line Interpreter</td>
<td>47</td>
</tr>
<tr>
<td>11-1</td>
<td>EUROScope – Terminal Window</td>
<td>48</td>
</tr>
<tr>
<td>12-1</td>
<td>EUROScope – Connection Configuration Dialog</td>
<td>49</td>
</tr>
<tr>
<td>12-2</td>
<td>EUROScope – Flash Dialog I</td>
<td>50</td>
</tr>
<tr>
<td>12-3</td>
<td>EUROScope – Flash Dialog II</td>
<td>50</td>
</tr>
<tr>
<td>13-1</td>
<td>EUROScope – Connection Configuration Dialog</td>
<td>52</td>
</tr>
<tr>
<td>15-1</td>
<td>USB Support – Device Manager</td>
<td>57</td>
</tr>
<tr>
<td>15-2</td>
<td>USB Support – Port Settings</td>
<td>58</td>
</tr>
</tbody>
</table>
List of Tables

Table 2-1: BDMAM ................................................................................................................... 12
Table 2-2: Recommended setting for BDMAM ........................................................................ 12
Table 2-3: BDMCM ................................................................................................................ 13
Table 2-4: BBRDM ................................................................................................................. 14
Table 2-5: BEAM .................................................................................................................... 15
Table 2-6: BBDM .................................................................................................................... 16
Table 3-1: Mode Switch Setting ............................................................................................. 21
Table 6-1: Processor Status ................................................................................................. 39
Table 6-2: CPU Registers ..................................................................................................... 40
Table 6-3: EUROScope – C Variables ................................................................................... 40
Table 12-1: Mode Switch Setting ........................................................................................... 49
Table 13-1: Mode Switch Setting ........................................................................................... 52
Table 16-1: Devices with no BDM functionality ..................................................................... 59
Table 16-2: Device Specific Maximum PLL Factor ................................................................. 59
Table 16-3: Device Specific PLL Register Access Restriction .................................................. 59
Table 16-4: MB96320 USART mapping .................................................................................. 60
Table 16-5: MB96330 USART mapping .................................................................................. 60
Table 16-6: MB96340 USART mapping .................................................................................. 60
Table 16-7: MB96350 USART mapping .................................................................................. 60
Table 16-8: MB96370 USART mapping .................................................................................. 60
Table 16-9: MB96380 USART mapping .................................................................................. 61
Table 16-10: MB96390 USART mapping ............................................................................... 61
Table 17-1: Lite/Profession version differences ...................................................................... 62