Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.



THIS SPEC IS OBSOLETE

Spec No: 002-05044

Spec Title: AN205044 - Startup DevKit16

Replaced by: NONE

Startup Devkit16

This application note describes about multifunctional Development Kit “DevKit16” which has been designed especially to support customers using the Flash and CAN technology on Cypress 16-Bit microcontrollers.

Contents

1 Introduction....1
2 Software....2
3 Installation....2
3.1 Check version of Monitor Debugger....3
3.2 Programming Softune Workbench Monitor Debugger to Flash....3
3.3 Start Debugging Session using Softune Workbench....4
3.4 Memory Mapping....6
3.5 Project Debugging....7
3.6 Starting a new Project....9
4 Document History....10

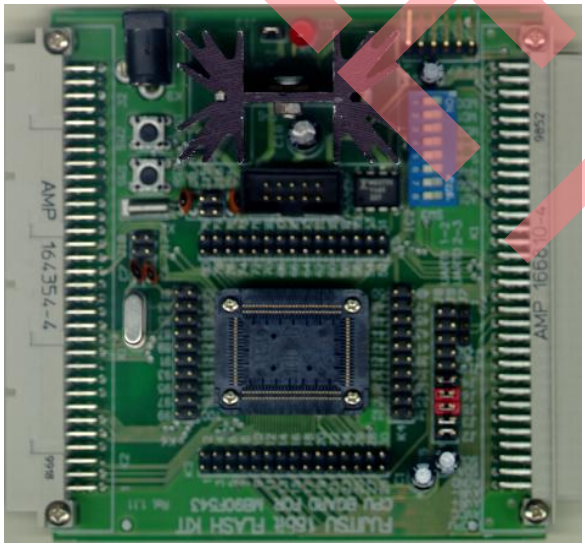
1 Introduction

The multifunctional Development Kit “DevKit16” has been designed especially to support customers using the Flash and CAN technology on Cypress 16-Bit microcontrollers. Due to the modular design of the DevKit16, it is possible to support different microcontrollers in the future.

The DevKit16 is based on a modular design, which consists of a CPU board and a main board. The CPU board can be used standalone as a simple evaluation board or - equipped with a socket - as a simple target board to work with the in-circuit emulator. The CPU board is built with two 96 pin VG connectors, which can be used to connect additional user hardware or the DevKit16 main board.

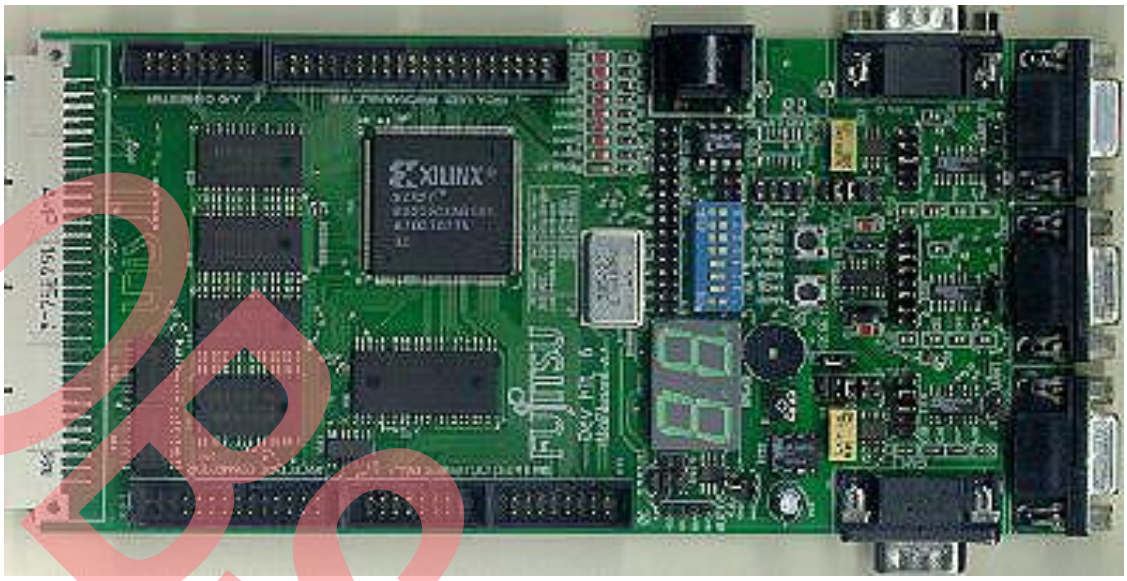
The main board incorporates in the main the monitor debugger and additional external circuits as buzzer, three serial interfaces, two CAN interfaces, two seven segment displays, 8 user LEDs, FPGA logic to simulated I/O ports, external Flash memory and SRAM for code and data storage. Via a 96pin VG interface connector a CPU board can be connected.

DevKit16-F543 (CPU Board)



OBSOLETE

DevKit16-MAIN (Main Board)



OBSOLETE

2 Software

The DevKit16 is supplied with a complete software Integrated Development Environment (IDE), the Softune Workbench, Version 3, which runs under Windows 95/98/NT. It features a complete set of compiler, assembler, linker, editor and debugger. Example projects for DevKit16 can be found on the Fujistu Micros CD 3.1 and also on the Cypress Web page.

A special DevKit16 Flash programming software to program internal and external Flash memory of the DevKit16 is also included in the software package, added to the DevKit16 or can be downloaded from the Software\Utility folder from the Cypress Micros web page.

Additionally a new software tool is provided with the DevKit16 - Processor Expert, from UNIS. This software is included in the shipment of the CPU board and is a development environment which allows rapid prototyping due to its special CPU expert knowledge. The Processor Expert is a code generator, which allows the user to design the software on a higher application level. Software debugging is also possible within Processor expert. Therefore Processor Experts own monitor debugger has to be programmed into the external Flash memory.

3 Installation

In the sections hereafter only the installation and usage of DevKit16 with the Cypress Softune Workbench is explained. For any installation hints regarding Processor Expert, please look at the hardware manual of the DevKit16 or the Processor Expert CD-ROM.

Before installing the software, please check carefully the Devkit16 hardware on causal shipment damages. In case of any damages please contact your Cypress sales office or your local distribution sales partner. If everything looks fine, insert the Cypress Micros CD-ROM V3.1 or higher and install the 16-Bit Softune Workbench software package. Additionally the example projects for DevKit16 should be installed.

To work with Devkit16, Softune Workbench V30L22 or higher is needed. To program the Softune Workbench monitor debugger the DevKit16 Flash programming utility V1.3 or higher must be used. The version of the ROM monitor debugger must be V1.3 or higher. Please check the Cypress (www.cypress.com/cypress-microcontrollers) or UNIS (www.processorexpert.com/) web site for the latest documentation and software updates.

3.1 Check version of Monitor Debugger

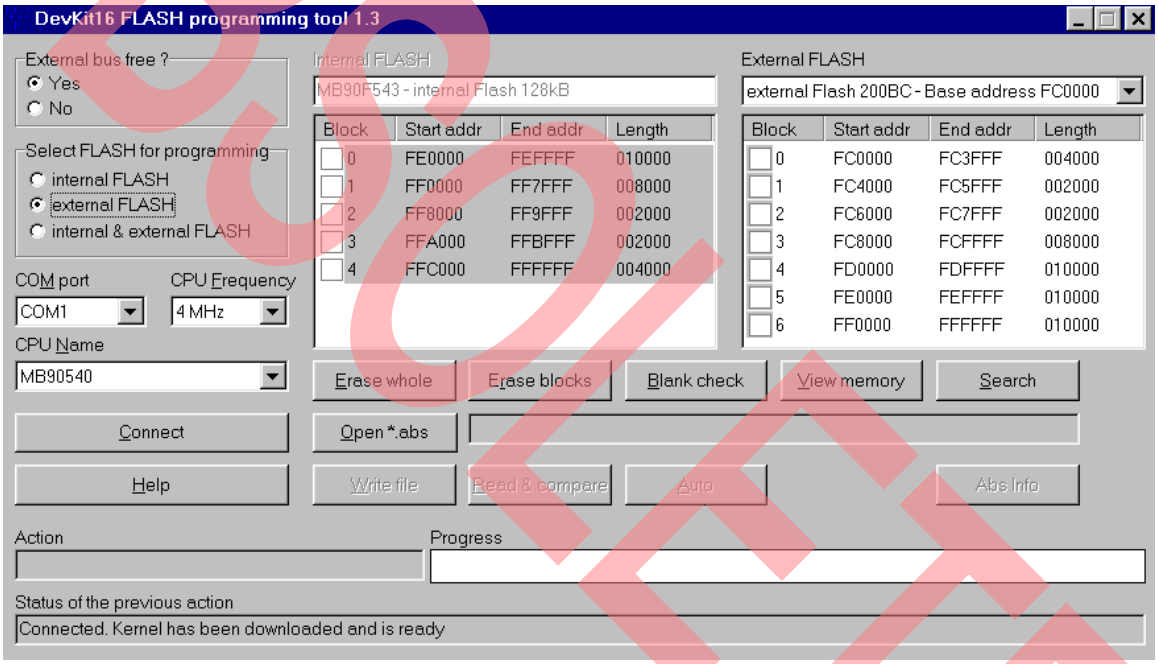
There exist two different Monitor Debugger Kernels. One Kernel is used for Processor Expert and one for Softune Workbench. Depending on which tool is used for the debugging, the corresponding debugger kernel must be programmed into the Flash! By default, the debugger kernel for Softune Workbench is programmed.

To check the current version of the used monitor debugger, set the jumper J1 to position 2-3 and power on the DevKit16. In the display will appear the currently used monitor version, which is located in the external Flash memory. The dots in the display indicate whether the Debugger for Processor Expert or Softune Workbench is used. If both dots are on, the Softune Workbench monitor is used. If only the left dot is on, the debugger for Processor Expert is used.

To use the Devkit16 with Cypress Softune Workbench Development environment please make sure that the Monitor version V1.3 or later is used. For any updates of the softune workbench monitor debugger, please check the Cypress web page www.cypress.com/spansionsupport and use the following link: *Microcontrollers, Development Environment*.

3.2 Programming Softune Workbench Monitor Debugger to Flash

To program the softune workbench monitor debugger, take care that the jumper J1 is set to position 2-3. Start the FlashDevKit16 flash programming utility and configure the settings as shown in the following figure.



Perform the following steps to program the Monitor debbugger:

1. Connect the serial interface of your PC to the User UART of the DevKit16
2. Power-on DevKit16
3. Go on CONNECT to establish the communication between the PC and DevKit16. A dialog box will ask to set jumper J1 to position 2-3 and to set the DIP switches 1 to on, all others to off. After that press Reset. A progess bar will be shown during the download procedure.
4. After the connection is established, the programming kernel has been downloaded to the internal RAM of the CPU. Use now Open *.abs to select the corresponding Softune Workbench Monitor debugger, e.g. for MB90F543:

   SWBMon543_115KV13, for 115K Baud debugging speed

   SWBMon543_38K4V13, for 38K4 Baud debugging speed

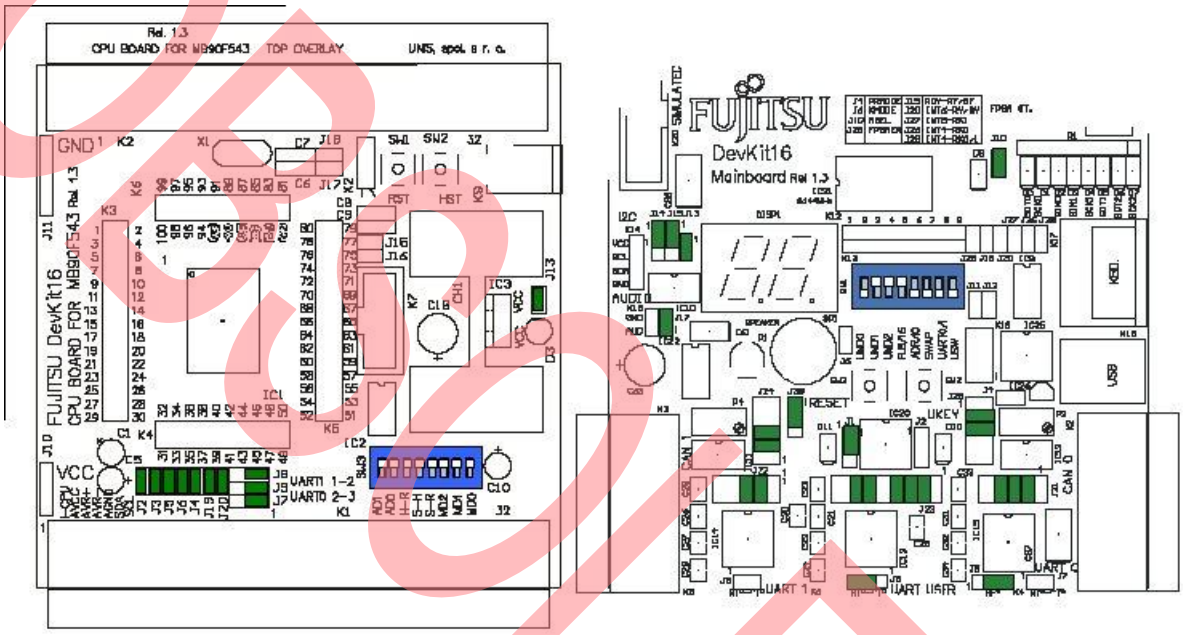
   SWBMon543_9K6V13, for 9K6 Baud debugging speed
5. Start <Auto> to run the automatice Erase, Blank check, Write and Verify commands. A progress bar will be shown during the programming procedure.

PROFET

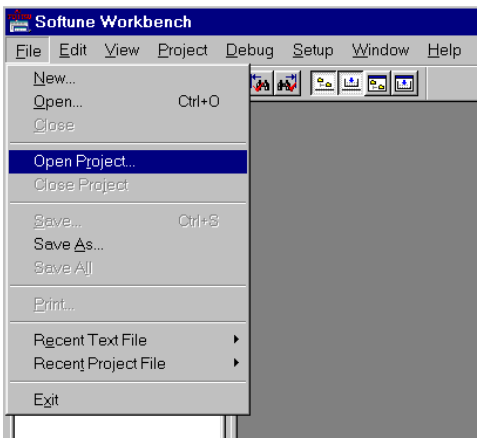
6. Now the DIP switches must be set to: 1 off, 2-4 on, 5-8 off. The Jumper J1 must now be set to position 1-2, otherwise the board will be constantly in reset if the debug session is started.
7. Press Reset button on the board
8. Monitor Debugger update is finished now

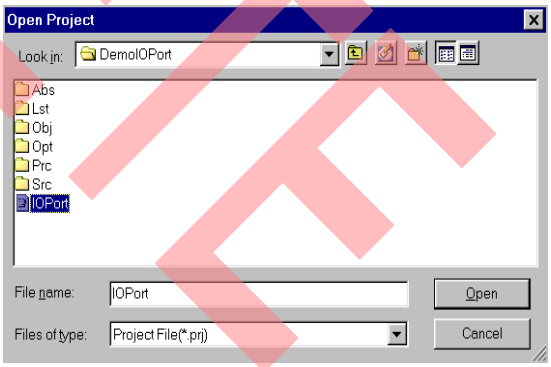
3.3 Start Debugging Session using Softune Workbench

Before starting the Softune Workbench debugging session, make sure that jumper J1 is set to position 1-2. The jumpers should be set to default as shown in the picture, and as described in the Devkit16 Users Guide. The DIP switches should be set to: 1 off, 2-4 on, 5-8 off. Connect the User UART to COM1 or COM2 of your PC. Power on Devkit16 now and start Softune Workbench from the Windows Start menu.

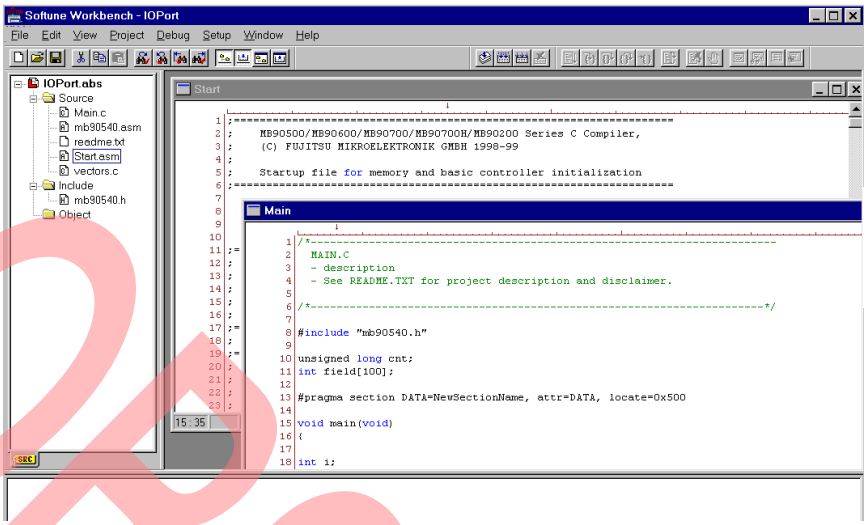


After Softune Workbench is started, use <File>, <Open Project> to open an example project for DevKit16, e.g. DemoIOPort. Click on the IOPort.prj file to open the project.



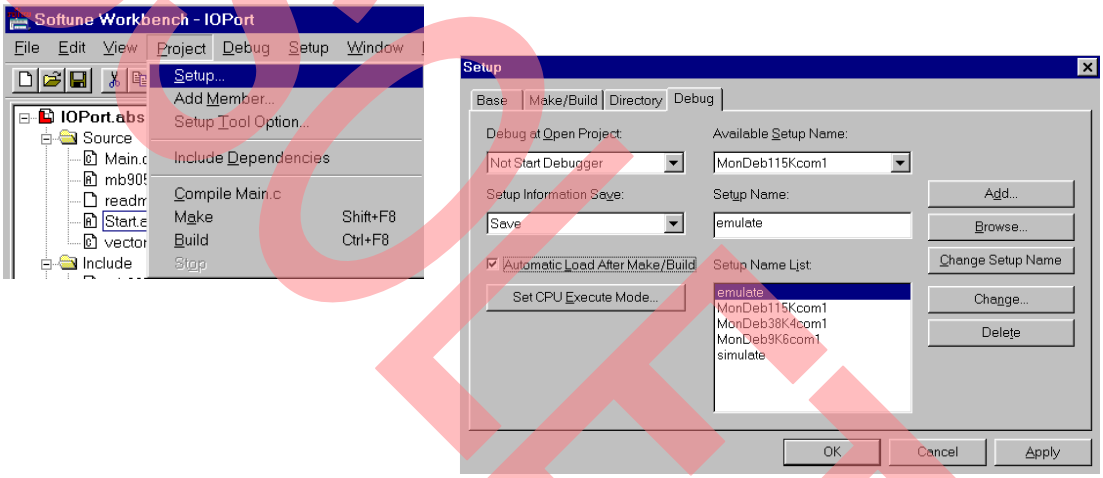


OBSOLETE



Project DemoIOPort for Devkit16 opened in Softune Workbench.

After opening the project, first the Monitor debugger setup should be checked. For that purpose use the <Project>, <Setup> menu.



Here you can select different setups or you can change the setups corresponding to your needs using the change button. Predefined setups can be found as:

MonDeb115Kcom1, Setup for 115Kbaud communication speed using COM port 1

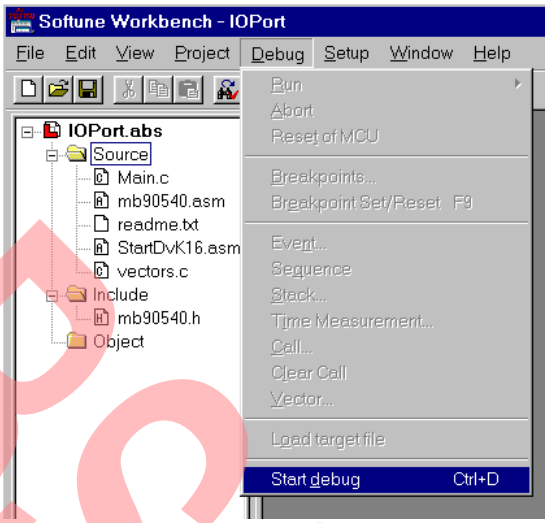
MonDeb38K4com1, Setup for 38K4Kbaud communication speed using COM port 1

MonDeb9K6com1, Setup for 9K6Kbaud communication speed using COM port 1

The setup must be selected according to the used Monitor debugger. E.g. if the Monitor Debugger SWBMon543_115KV13 is used, the setup file MonDeb115Kcom1 must be selected, using the <Available Setup Name> box. If COM2 is used on the PC, the setup **must** be changed using the change button. It is also possible to enter a new setup by entering a new setup name into the <Setup Name> box and using the Add button. This will invoke the Setup Wizard, configuring a new debugger setup.

OBSOLETE

If the correct Debugger setup is selected, the Monitor debugger can be started using the <Debug>, <Start Debug> menu.



After starting the Debugger, a progress bar will indicate the download of the source code to the DevKit16 external SRAM. After that, the source window is opened automatically, displaying the StartDvK16.asm file.

3.4 Memory Mapping

For the user application the address area 0xF8.2000 - 0xFF.FFFF can be used. The interrupt vector table is located at address 0xFF.FC00. In the external Flash memory the area 0x10.0000 - 0x12.FFFF can be used by the user.

The following resources are reserved for the Monitor debugger and cannot be used by the user application:

Interrupt Vector table entries which are used by the debugger:

- 8 - Reset
- 9 - ROM correction, INT9
- 10 - Exception
- 15 - External interrupt 0/1 (for MB90F543, MB90F497)

Note:

The Monitor debugger is located in the external flash memory. After power-on, first the external Flash memory is located in the address are 0xFC.0000-0xFF.FFFF, the SRAM is located at address range 10.0000 - 17.FFFF. Directly after Reset, these address areas are swapped. So the external Flash memory is swapped to the area 0x10.0000 - 0x17.FFFF and the the user SRAM is located at address location 0xF8.0000 - FF.FFFF.

So the monitor debugger is located first in the page 0xFF.0000, which is page 0x13.0000 after the memory swapping.

OBSOLETE

3.5 Project Debugging

When the debugger is started, versatile features exist for source code debugging. With the right mouse button, a context menu can be opened to set Breakpoints, open watch window, use Inline Assembler or edit the source code. In the upper right corner several Icons can be found for Continuous Run, Single Step, Step Over, Step In, set/erase Breakpoints, opening the Watch Window, Register Window, Memory Window and Disassembler Window.

Note:

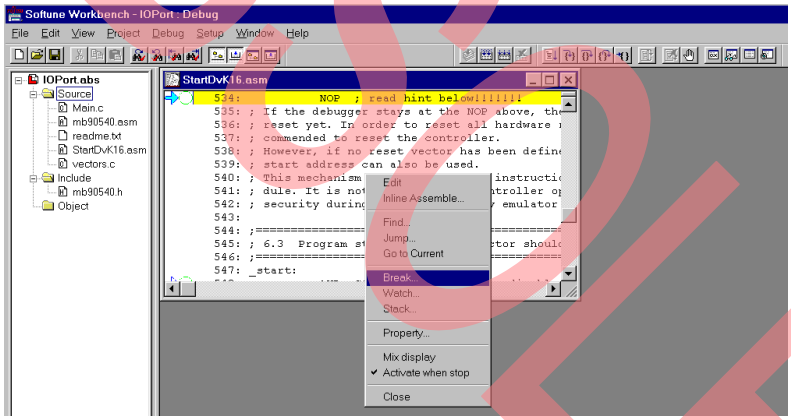
The Abort button does not work! To abort the program, the Abort button on the DevKit16 must be used! The Step Out function is not implemented in the Monitor Debugger, so this function does not work.

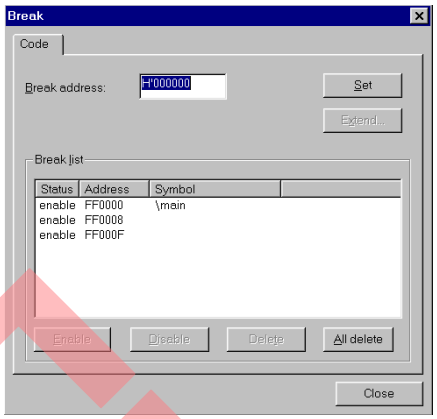
The <Call> function, using the <Debug> menu works only with Softune Workbench V30L24 or higher, the <Time Measurement> function is not implemented and cannot be used.

The monitoring function for the Watch window or Memory window is also not supported by the Monitor Debugger.

3.5.1 Breakpoints

To set breakpoints, use the right mouse button to get the context menu and go on Break, or use the <Debug>, <Breakpoint> menu. After this the breakpoint dialog will be opened. Now the break address or symbol can be entered for the breakpoint. It is also possible to set a breakpoint within the source window by using the right mouse button on the green circles, which are located on the left side of the source window.



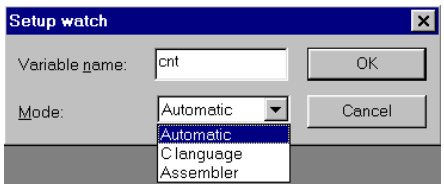


3.5.2 Watch window

Using Context Menu

To open the watch window, use the right mouse button to get the context menu and go on Watch, or use the Display Watch Window icon of the Softune Workbench tool bar. Alternately the menu <View>, <Watch> can be used as well.

With the context menu the following dialog is opened. Now the name of the variable can be entered, which is added to the Watch window. Normally the Mode is set to automatic.



Note:

If registers of the internal resources are added (e.g. data direction register port 0: DDR0) to the watch window, the Mode in the dialog box should be set to Assembly. This allows a correct display of the register data, if the headerfiles are used supplied by FME. The internal I/O registers can only be added to the Watch window as byte or word. It is not possible to add single bits (e.g. DDR_D00) to the watch window.

OBSOLETE

Using Toolbar Watch Window Icon

The watch window can also be opened using the Watch window icon of the Softune Workbench toolbar.



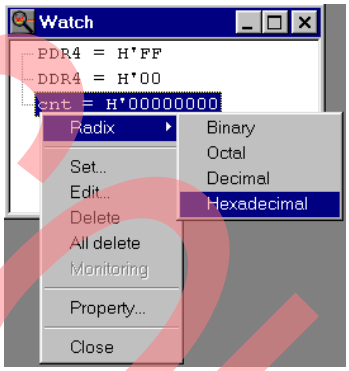
If the icon is used, the Watch window is opened directly. Now the right mouse bbutton is used with the set command to open the above dialog box.

Double click in the source Window

A further method to add variables to the Watch window is to double click the corresponding symbol in the source code. After that, the dialog box can be opened using the right mouse button, or using the menu <View>, <Watch>.

Context dialog of Watch Window

If the Watch window is open, the context menu allows versatile settings as e.g. modify radix of the shown variable, modify the variable value by the edit command, add new variables using the set command.



Note:

The monitoring function is not supported by the Monitor debugger for the watch window. The window is updated after the Debugger stops at a breakpoint.

3.5.3 Display Assembly Code

To see C-Source code and the generated assembly code, use the command <Mix Display> within the context menu. Via the Disassembler icon of the toolbar, the pure assembly listing of the source code can be shown in a separate source window.



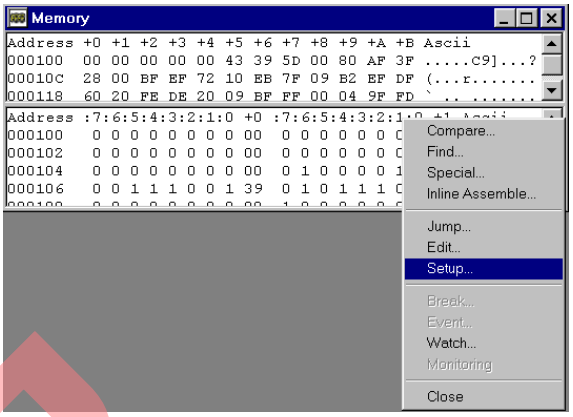
3.5.4 Memory Window

To open the memory window use the icon "Display memory Window" of the toolbar, or the menu <View>, <Memory>.



If the Memory is opened also a context menu exist for special commands within the memory window. So memory areas can be compared using the <Compare> command, the display mode can be changed by the <setup> command (bit, byte, word, long), the Inline Assembler can be started via <Inline Assmebler>, via the <Special> command memory fill and copy routines can be executed and memory locations can be edit by the <Edit> command or just by typing directly in the memory window. With the slider on the right side of the memory box, the display can be splitted, so it is possible to watch two different memory areas within the memory window.

OBSOLETE



Note: The monitoring function is not supported by Monitor Debugger.

3.6 Starting a new Project

To start a new project, a specially prepared Template project can be found in the DevKit16 example folder. This template incorporates some basic microcontroller settings (external bus interface, PLL) which should be checked by the user in detail. These settings are espicially done in the startDVK16.asm file. Right at the beginning of the project, the user should always check these settings, and modify them to his needs. Also some basic settings for C-Compiler, Linker and debugger are included to ease the start of a new project with Devkit16.

So the user can just make a copy of the template project, rename the project folder name and the template.prj file in the new project folder.

After that, the user can start directly to write his own code in the main.c program. The application Note "Softune Workbench Getting Started" gives some more detailed information about project settings.

For further information regarding linker and C-Compiler settings, please look at the "Softune Workbench getting started" and "Startup" Application Notes. Further detailed information can be found in the corresponding Users Manuals and the online help of Softune Workbench.

OBSOLETE

4 Document History

Document Title: AN205044 - Startup DevKit16

Document Number: 002-05044

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	NOFL	06/19/2000 06/20/2000	Initial release Some minor text corrections
*A	5129545	NOFL	03/22/2016	Migrated Spansion Application Note from MCU-AN-390036-E-V11 to Cypress format The Devkit16 is outdated and this AN to be Obsoleted

OBSOLETE

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Lighting & Power Control	cypress.com/powerpsoc
Memory	cypress.com/memory
PSoC	cypress.com/psoc
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless/RF	cypress.com/wireless

PSoC® Solutions

cypress.com/psoc

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

Cypress Developer Community

Community | Forums | Blogs | Video | Training

Technical Support

cypress.com/support

OBSOLETE

PSoC is a registered trademark and PSoC Creator is a trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

© Cypress Semiconductor Corporation, 2000-2016. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you under its copyright rights in the Software, a personal, non-exclusive, nontransferable license (without the right to sublicense) (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. Cypress also grants you a personal, non-exclusive, nontransferable, license (without the right to sublicense) 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 to the minimum extent that is necessary for you to exercise your rights under the copyright license granted in the previous sentence. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. 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 Company shall and hereby does release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. Company shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.