SPANSION®

Tutorial

How to use Keil µVision with Spansion templates

Warranty and Disclaimer

The use of the deliverables (e.g. software, application examples, target boards, evaluation boards, starter kits, schematics, engineering samples of IC's etc.) is subject to the conditions of Spansion as set out in (i) the terms of the License Agreement and/or the Sale and Purchase Agreement under which agreements the Product has been delivered, (ii) the technical descriptions and (iii) all accompanying written materials.

Please note that the deliverables are intended for and must only be used for reference in an evaluation laboratory environment. The software deliverables are provided on an as-is basis without charge and are subject to alterations. It is the user's obligation to fully test the software in its environment and to ensure proper functionality, qualification and compliance with component specifications.

Regarding hardware deliverables, Spansion warrants that they 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. Should a hardware deliverable turn out to be defect, Spansion's entire liability and the customer's exclusive remedy shall be, at Spansion´s sole discretion, either return of the purchase price and the license fee, or replacement of the hardware deliverable or parts thereof, if the deliverable is returned to Spansion 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 Spansion, or abuse or misapplication attributable to the customer or any other third party not relating to Spansion or to unauthorised decompiling and/or reverse engineering and/or disassembling.

Spansion does not warrant that the deliverables do not infringe any third party intellectual property right (IPR). In the event that the deliverables infringe a third party IPR it is the sole responsibility of the customer to obtain necessary licenses to continue the usage of the deliverable.

In the event the software deliverables include the use of open source components, the provisions of the governing open source license agreement shall apply with respect to such software deliverables. To the maximum extent permitted by applicable law Spansion disclaims all other warranties, whether express or implied, in particular, but not limited to, warranties of merchantability and fitness for a particular purpose for which the deliverables are not designated. To the maximum extent permitted by applicable law, Spansion's liability is restricted to intention and gross negligence. Spansion is not liable for consequential damages.

Should one of the above stipulations be or become invalid and/or unenforceable, the remaining stipulations shall stay in full effect. The contents of this document are subject to change without a prior notice, thus contact Spansion about the latest one.

This board and its deliverables must only be used for test applications in an evaluation laboratory environment.

Debugging (Keil µVision)

Debugging (Keil µVision)

Keil Development Tools (Software)

- MDK-ARM-EVALUATION

  Free download

  - 32K C/C++ Compiler, Assembler, µVision IDE/Debugger/Simulator
- MDK-ARM-B (Basic Edition)
  - 256K C/C++ Compiler, Assembler, µVision IDE/Debugger/Simulator
- MDK-ARM (Standard Edition)
  - C/C++ Compiler, Assembler, µVision IDE/Debugger/Sim., RTX Kernel
- ULINK-ME
  - USB-JTAG adapter (only 3.3 Volt support)
- ULINK2
  - USB-JTAG adapter
- ULINKpro
  - High-Speed debug and trace unit

Debugging (Keil µVision)

Installation

- Download latest version of µVision from KEIL Website
  - Evaluation Version
    - https://www.keil.com/demo/eval/arm.htm
    - Registration required

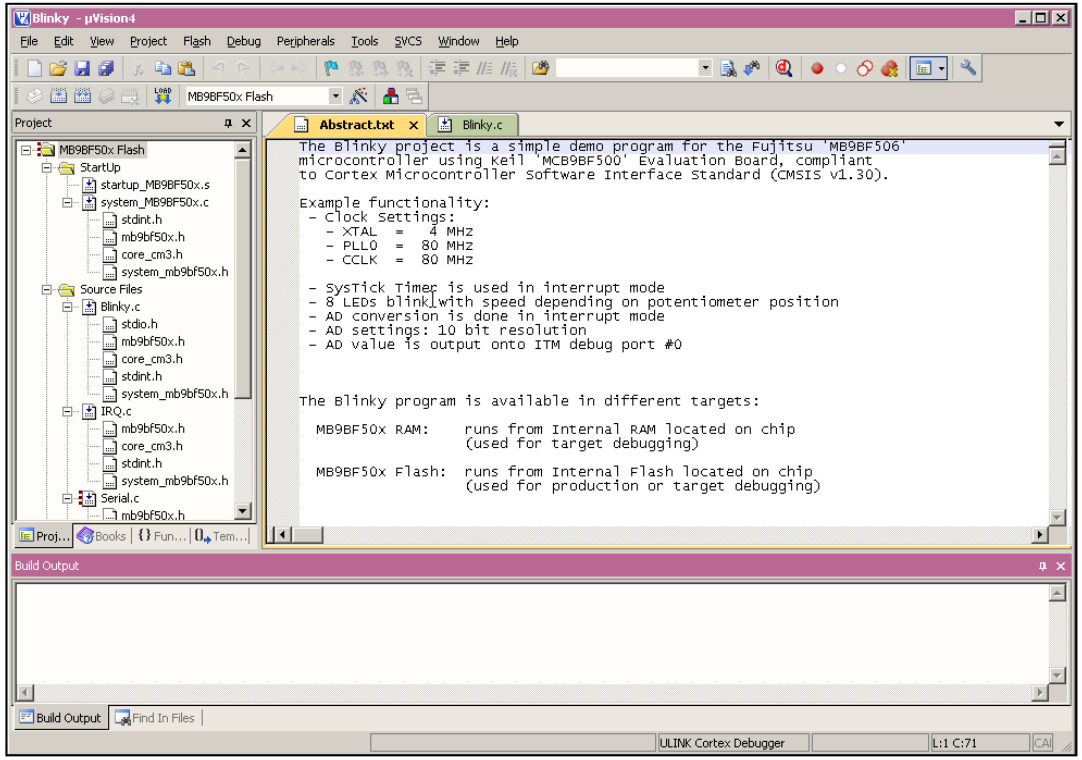
Free download

- Install µVision
- Install ULINK-ME
  - Special installation is not needed, because ULINK-ME acts as a USB Human Interface Device (HID) and thus needs no extra USB driver
- Install ULINK Pro (optional)
  - ULINK Pro needs an own dedicated USB driver located in: *<Installation Path>\Keil\ARM\ULINK*
- Start µVision

Debugging (Keil µVision)

Getting started

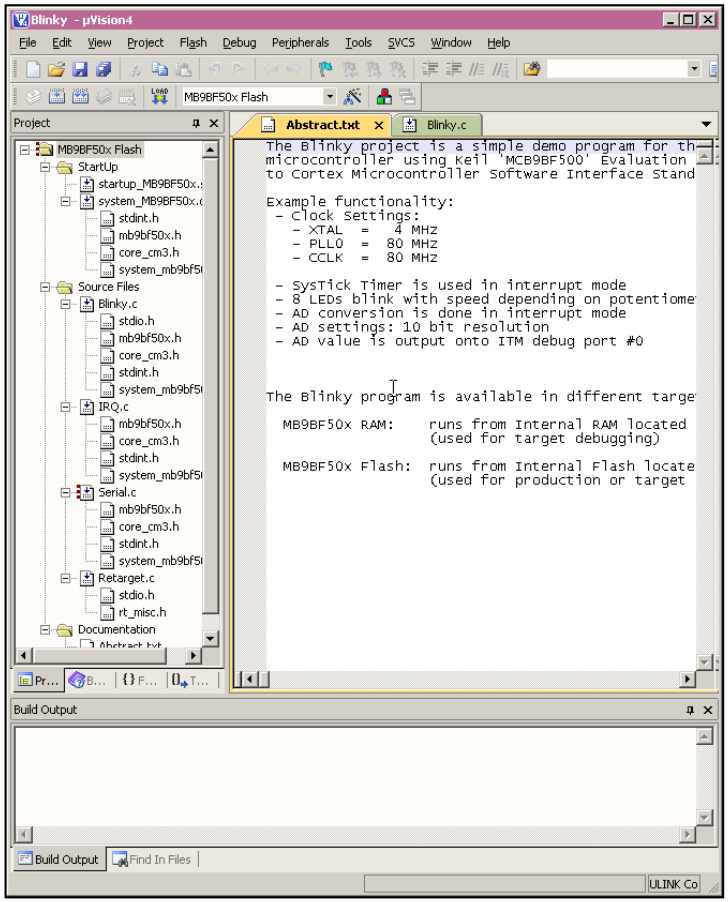
- Choose Menu: *Project→Open Project...*
  - Browse to: *<Installation Path>\Keil\ARM\Boards\Keil\MCB9BF500\Blinky*
  - Choose *Blinky.uvproj*



Debugging (Keil µVision)

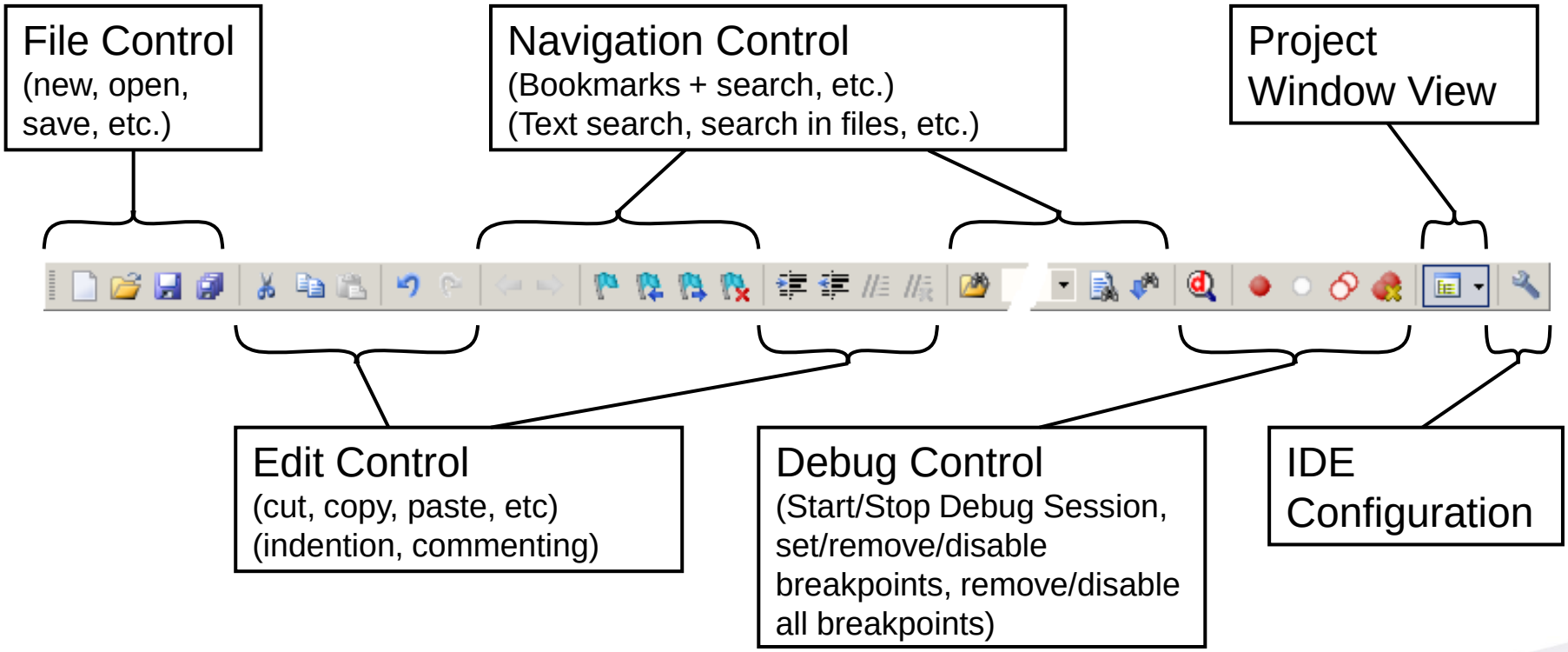
Main Window

- KEIL µVision
  - Project window on left side of IDE window
    - Choose: *View→Project Window* if hidden
  - Source files on right side of IDE window as tabbed windows
  - Output window on bottom side of IDE window



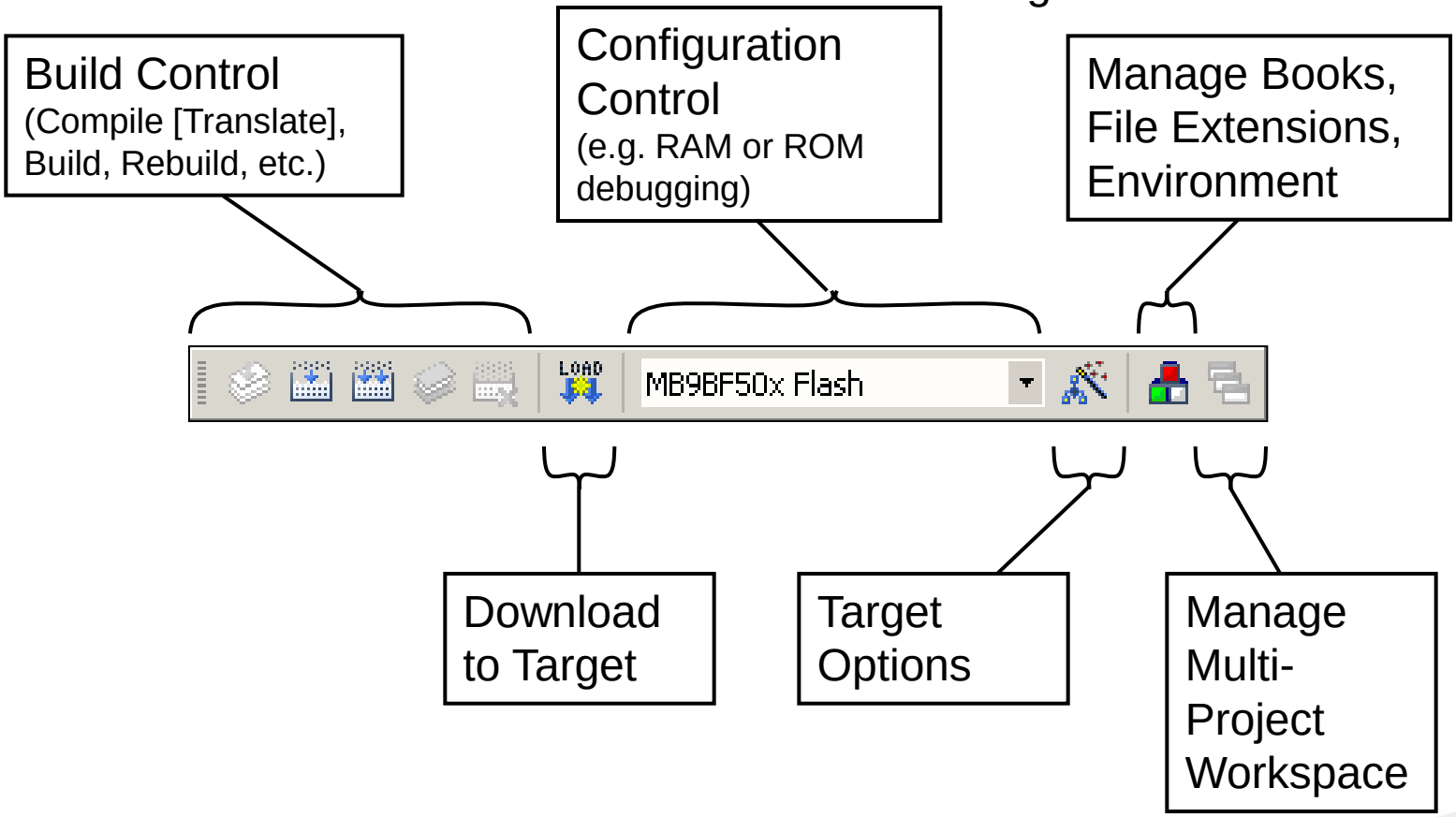
Menu Bars

- Menu Bar 1
  - Can be moved in bar window area or set floating



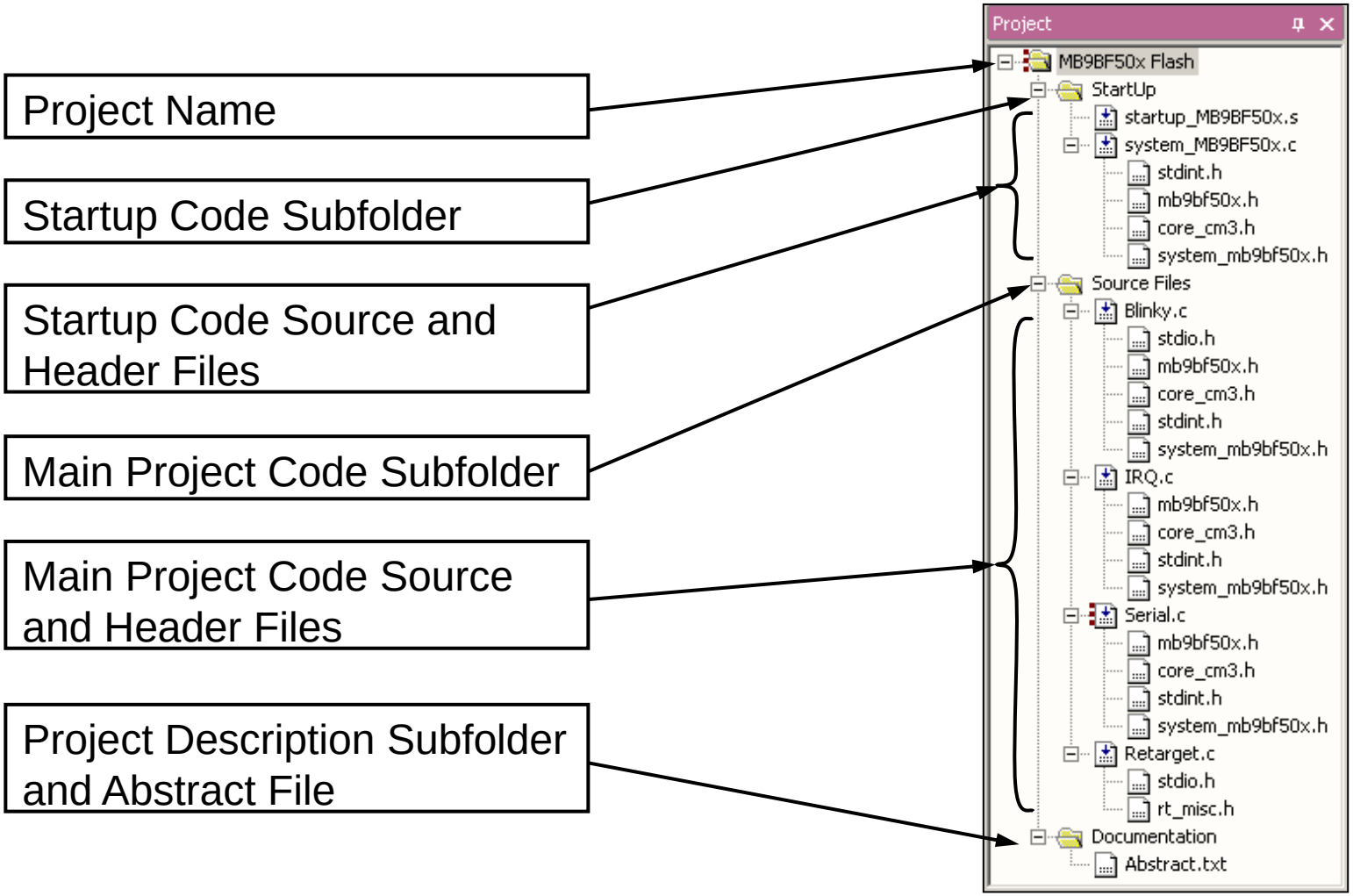
Menu Bars

- Menu Bar 2
  - Can be moved in bar window area or set floating



Debugging (Keil µVision)

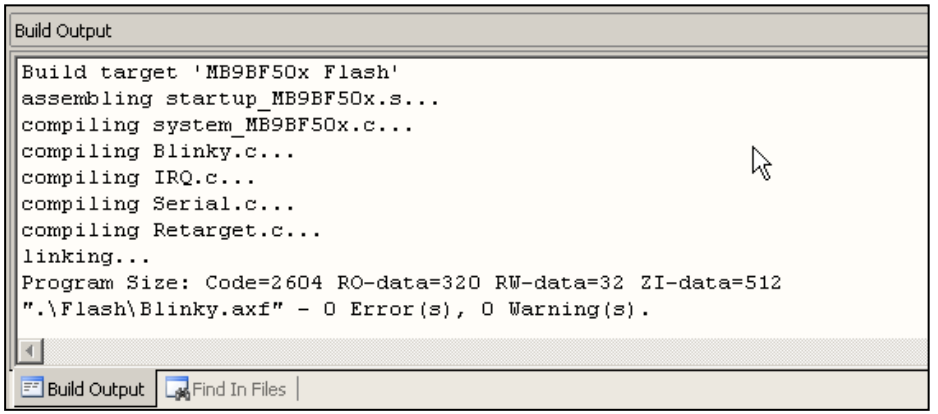
µVision Project Window



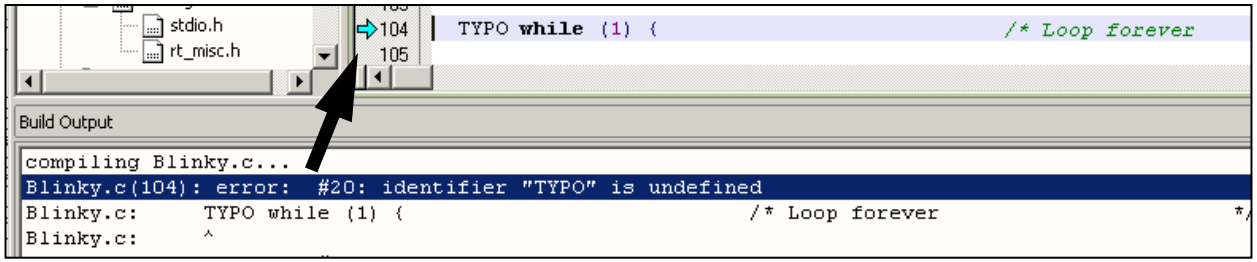
Debugging (Keil µVision)

Making Project

- Making the Project
  - Use Rebuild Icon () or *Project→Rebuild all target files*
  - Check for no errors in Output window below



  - Build errors are shown in Output window.
    - Can be double-clicked by showing the source line with a blue arrow



Debugging (Keil µVision)

Start Debugging

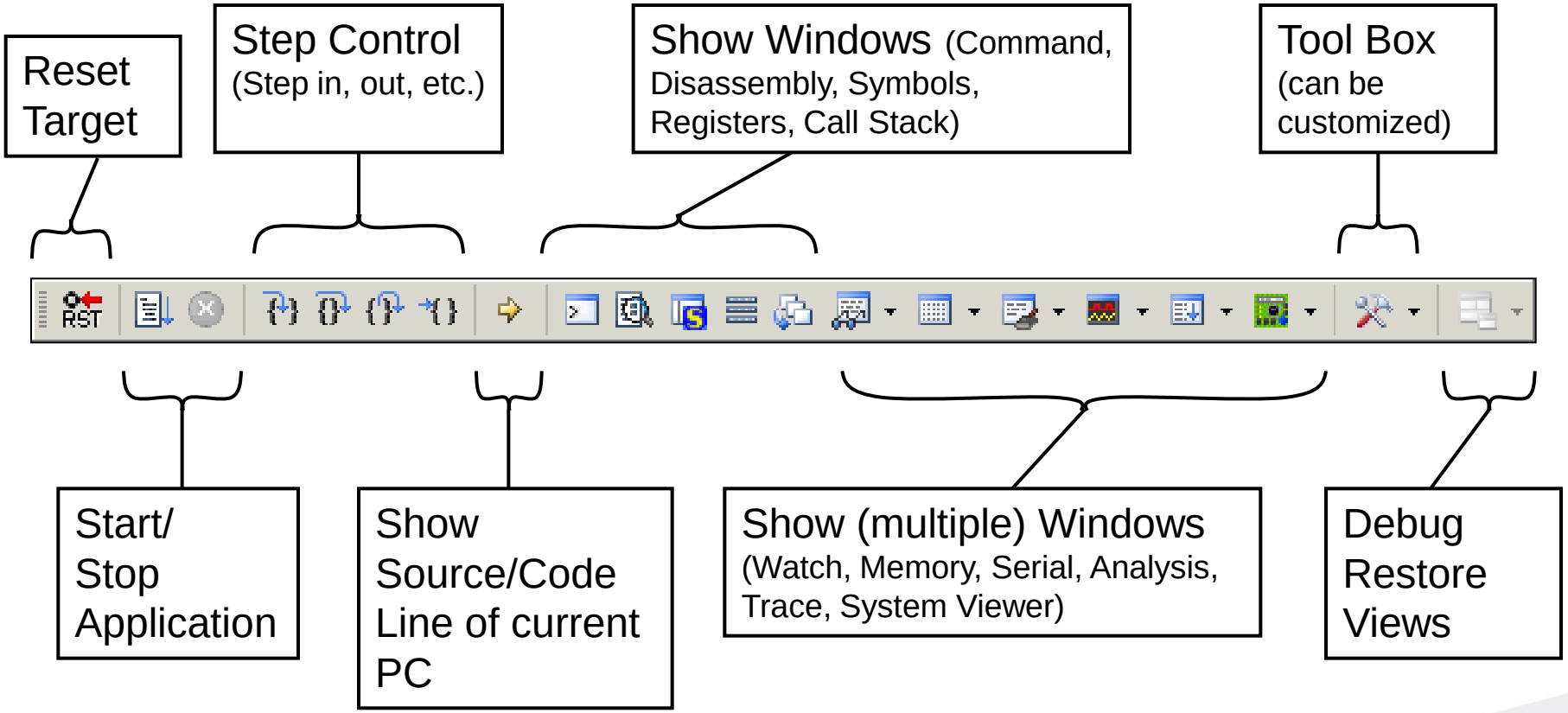
- Download to target first, when MCU Flash does not contain the current application openend and built in the IDE
  - Use Download Icon (

) or Menu: *Flash→Download*
- Start Debug Session
  - Use Start/Stop Debug Icon () or Menu: *Debug→Start/Stop Debug Session*
- Ending Debug Session
  - Use same way as for starting debug session

Debugging (Keil µVision)

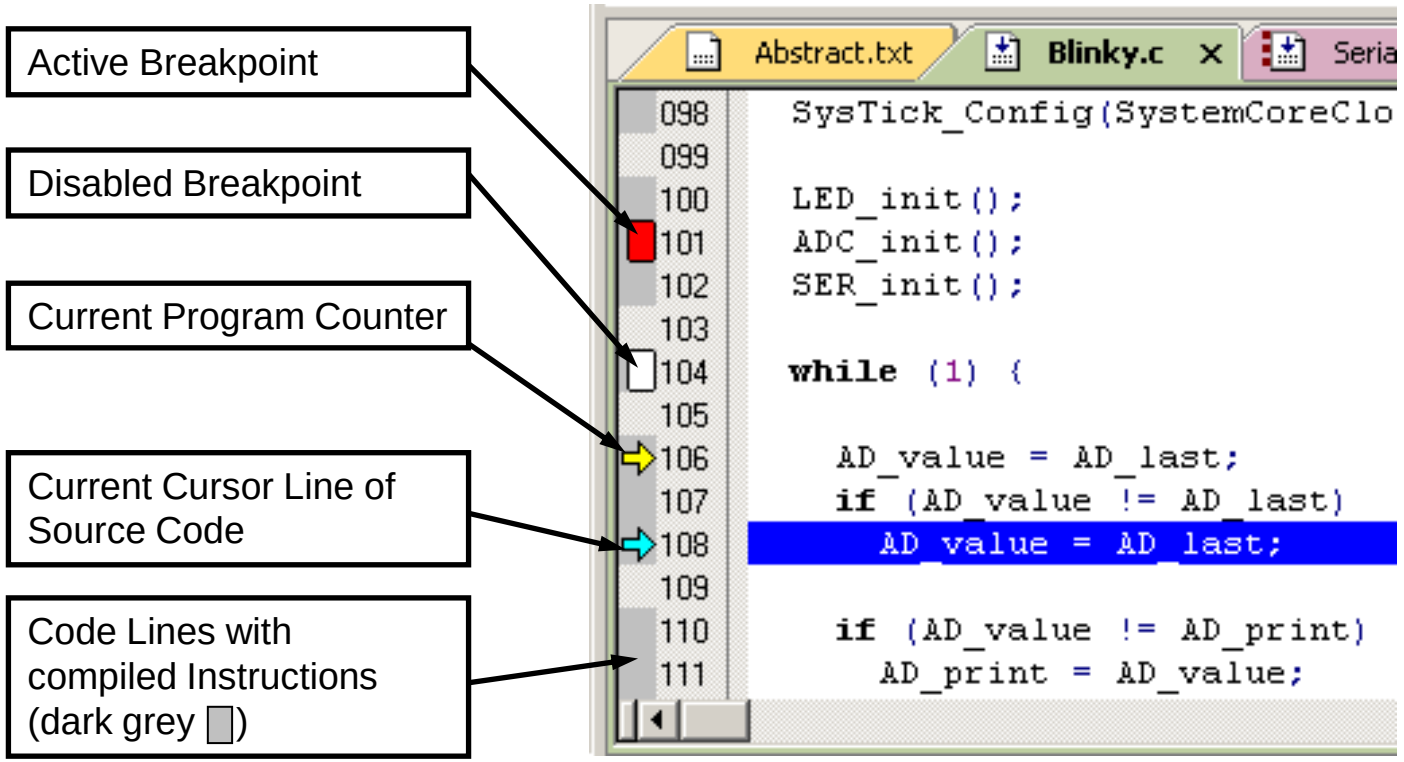
Debugging Icon Bar

- During a Debug Session there will be visible a new icon bar



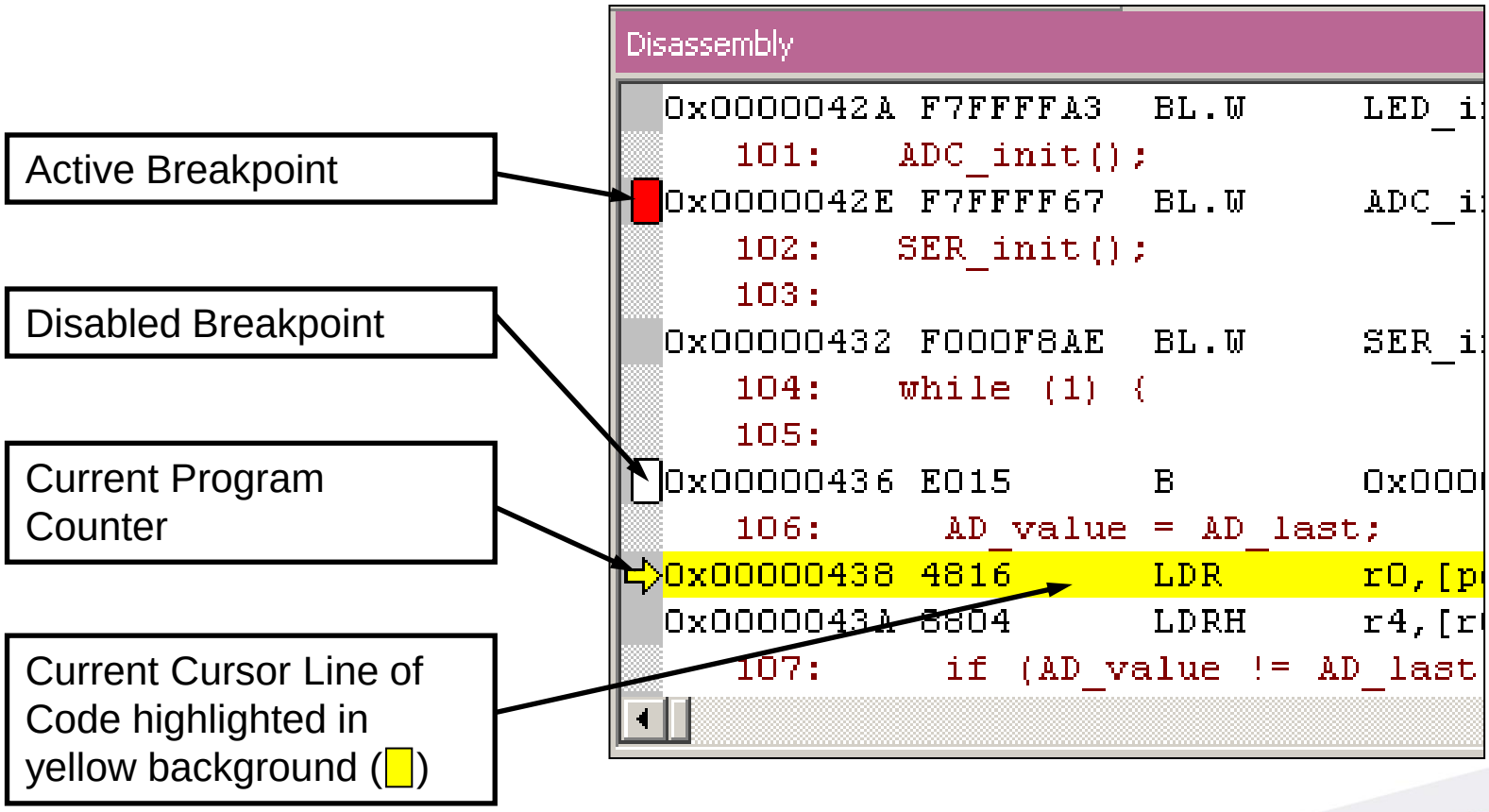
Debug Windows

- Source View
  - The Source windows do not change contents but get additional information



Debug Windows

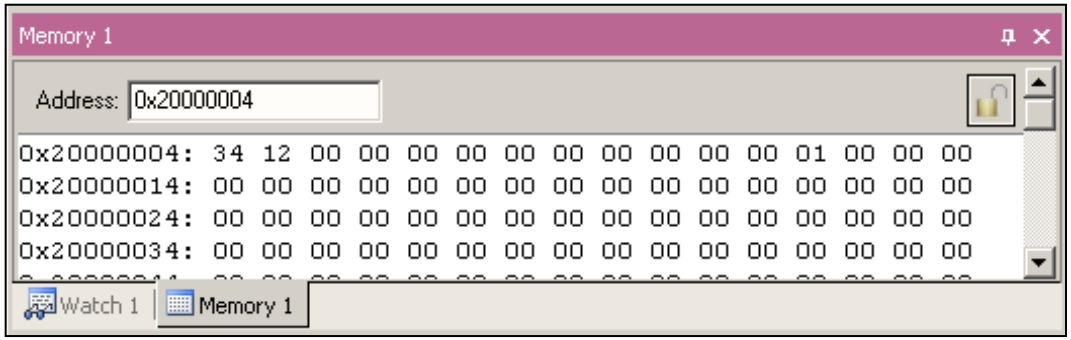
- Disassembly View
  - Mixed mode is selectable and deselectable



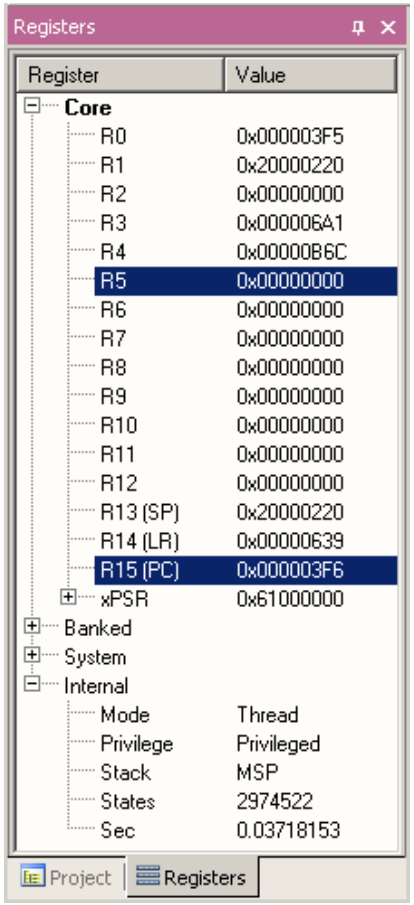
Debugging (Keil µVision)

Debug Windows

- Memory Window
  - Up to 4 Memory windows can be displayed in tabs
  - Memory is updated during runtime
  - Memory window tabs are shared with Watch windows

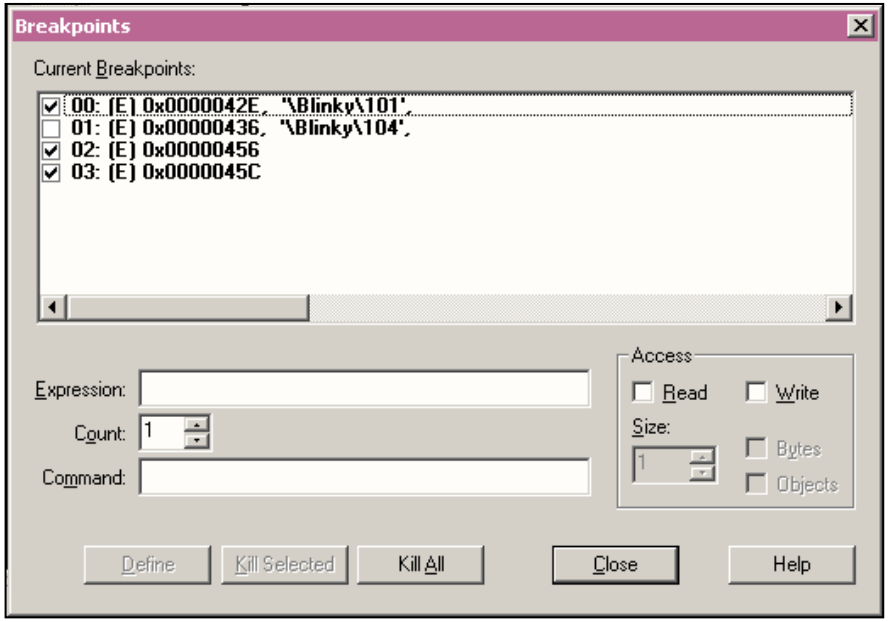


- Register View
  - Register view is a tab of the Project window
  - Changes are highlighted in dark blue text background
  - Register tree knots can be expanded



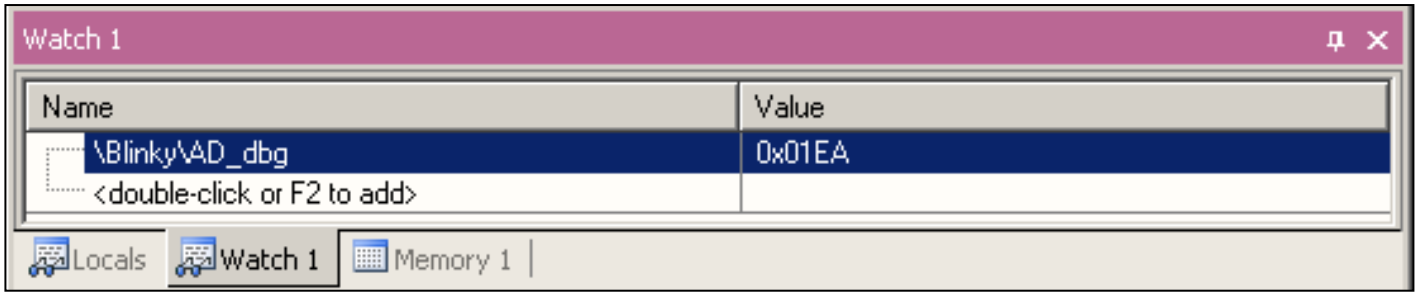
Debug Windows

- Breakpoints
  - Not a Workspace window, but a Dialog window
  - Can opened by menu: *Debug→Breakpoints...* or <Ctrl>-B
  - Any changes can only be seen in Workspace after "Close"
  - Conditional expressions can be entered here



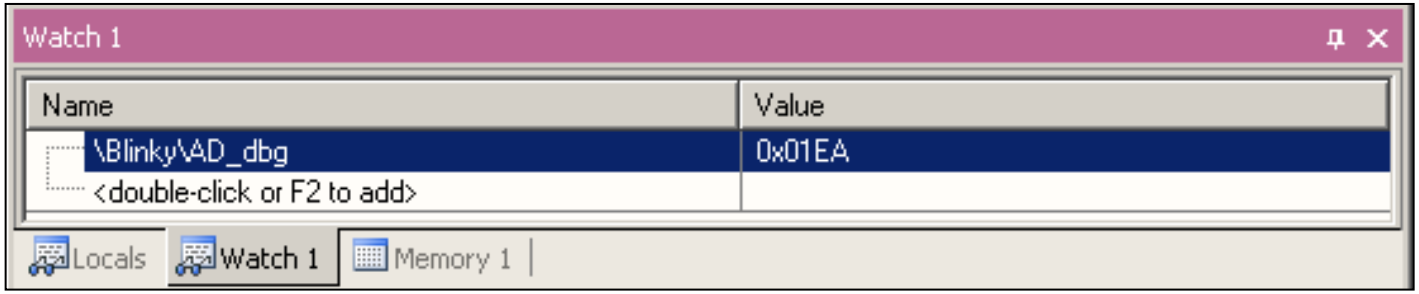
Debug Windows

- Variable Windows (1)
  - Watch Windows
    - Up to 2 Watch windows are sharing their tabs with e.g. Memory and Local views
    - Updated during runtime
    - Any changes are highlighted in dark blue text backround color
    - Displayed values can be changed by user during break



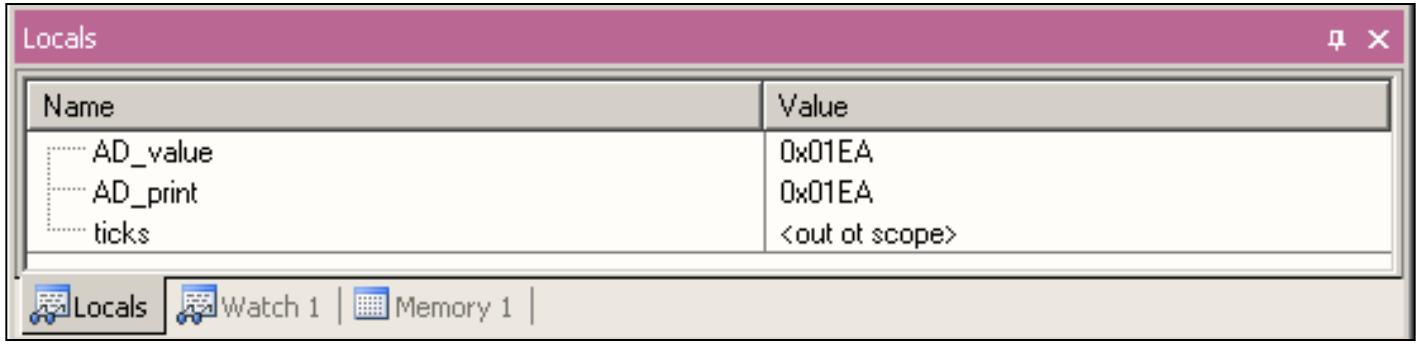
Debug Windows

- Variable Windows (1)
  - Watch Windows
    - Up to 2 Watch windows are sharing their tabs with e.g. Memory and Local views
    - Updated during runtime
    - Any changes are highlighted in dark blue text backround color
    - Displayed values can be changed by user during break



Debug Windows

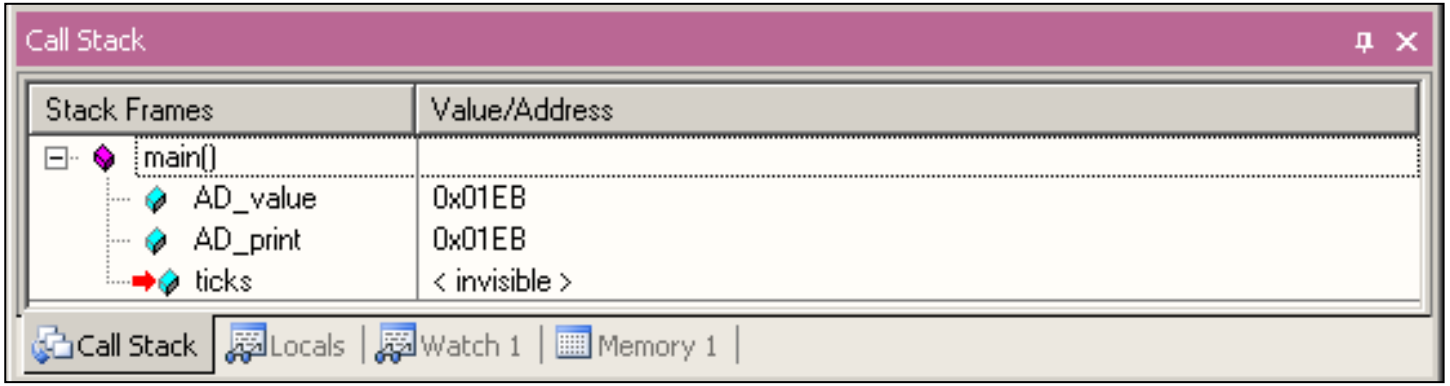
- Variable Windows (2)
  - Local View
    - The local view shares the tab with e.g. Memory and Watch windows
    - Any changes are highlighted in dark blue text backround color
    - Displayed values can be changed by user during break



Debugging (Keil µVision)

Debug Windows

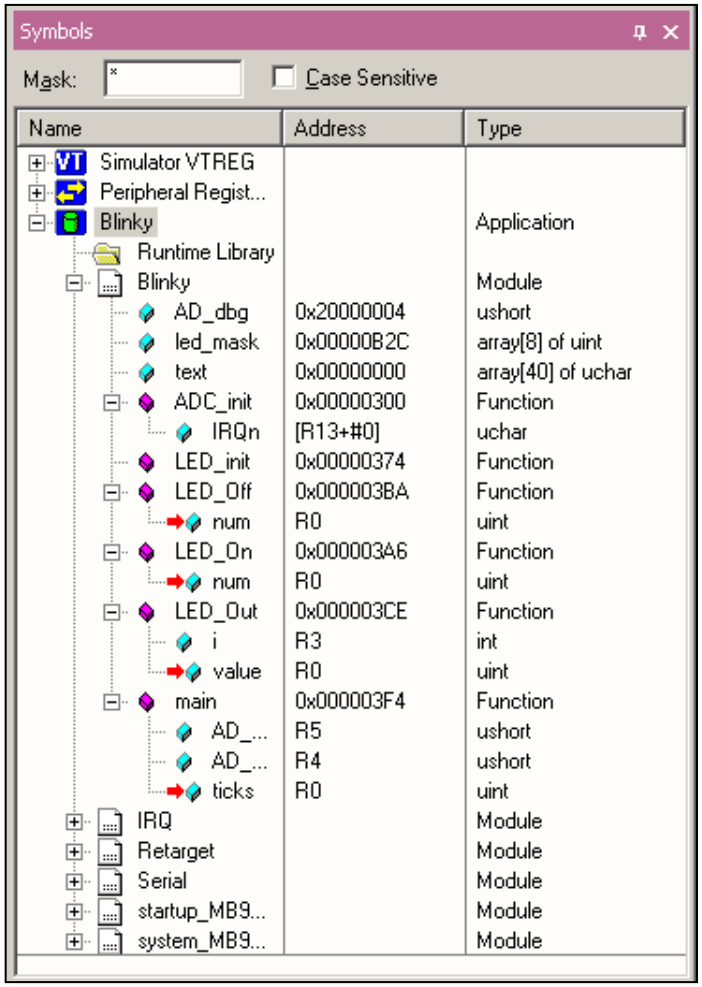
- Callstack
  - The Call Stack view shares its tab with the Variable and Memory window
  - It shows the current called function and their local environment
  - All displayed Call Stack values are not modifyable



Debugging (Keil µVision)

Debug Windows

- Symbol Window
  - The Symbol window shows all symbols of the user‘s project
  - It is sorted in tree form by dependence
    - E.g.
      Project→Module→
      Function→Symbol
  - Peripheral registers can also be displayed
  - Simulator VTREG can be displayed
  - Mouse-over a symbol shows its contents as a tool tip
    - E.g.: `AD_dbg = 0x01EB`

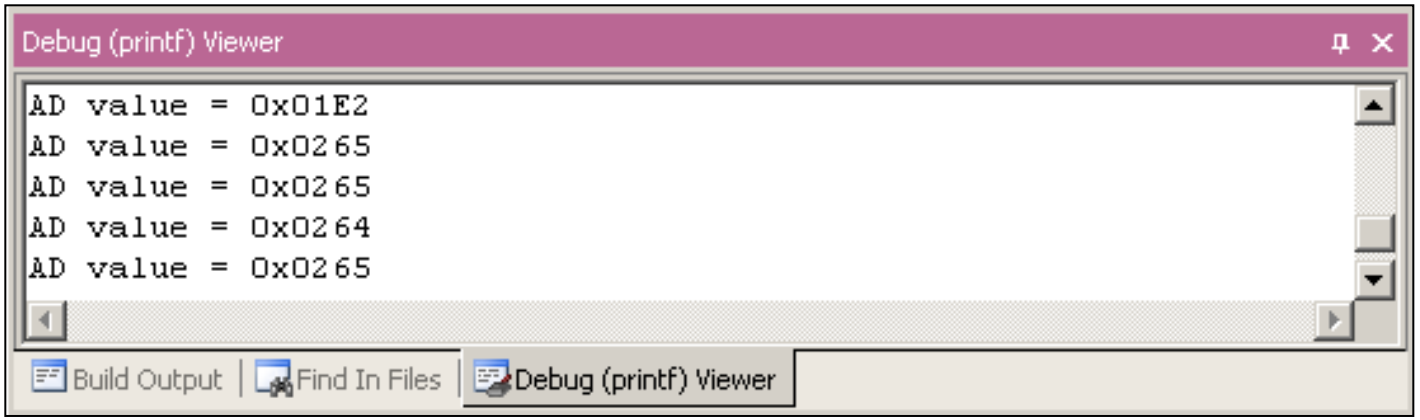


Debugging (Keil µVision)

Debug Windows

- Terminal Windows
  - Up to 3 UART channels can be displayed
  - One direct **`printf`** view is supported (via ITM Viewer)

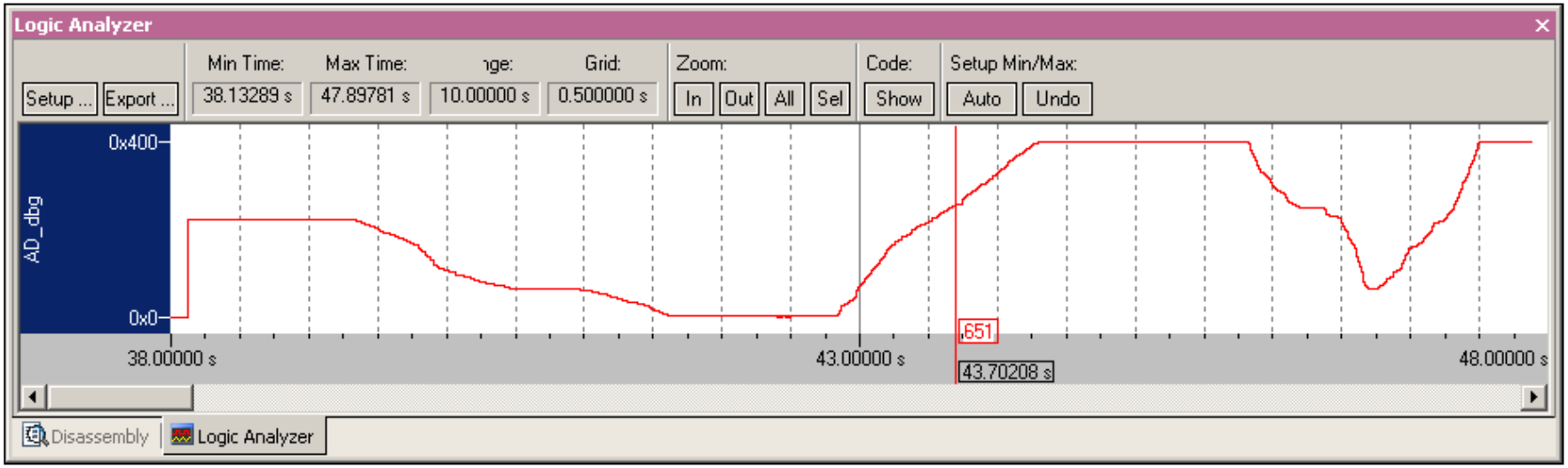
```
116         clock_ls = 0;
117         printf ("AD value = 0x%04X\n\r", AD_print);
118     }
```



Debugging (Keil µVision)

Debug Windows

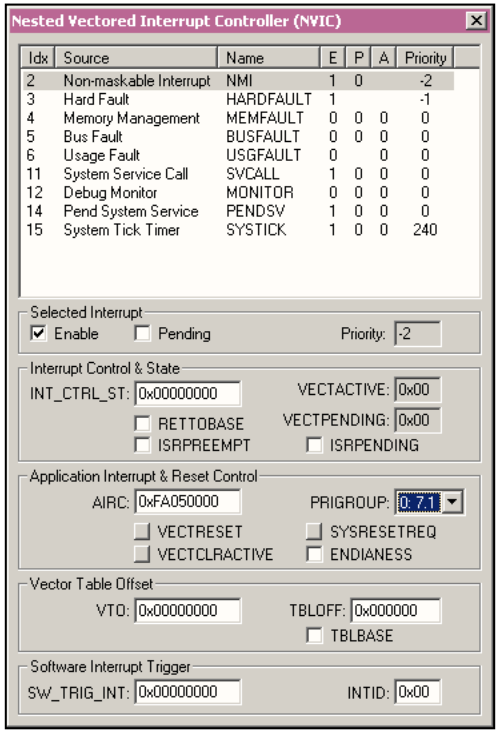
- Logic Analyzer
  - The IDE allows to display graphically the value of a variable
  - Adding a signal is as easy as adding to a Watch window
  - The Logic Analyzer is a tabbed window together with the Disassembly window



Debugging (Keil µVision)

Core Peripherals

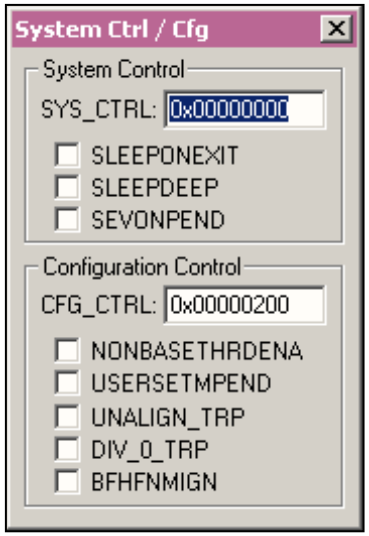
- Nested Vectored Interrupt Controller
  - The NVIC settings for the Core Interrupts can be adjusted in a dialog window
  - Adjustments for:
    - Enable/Pending
    - Priority
    - Interrupt Control & State
    - Application Interrupt & Reset Control (PRIGROUP, ENDIANESS, ...)
    - Vector Table Offset
    - Software Interrupt Trigger



Debugging (Keil µVision)

Core Peripherals

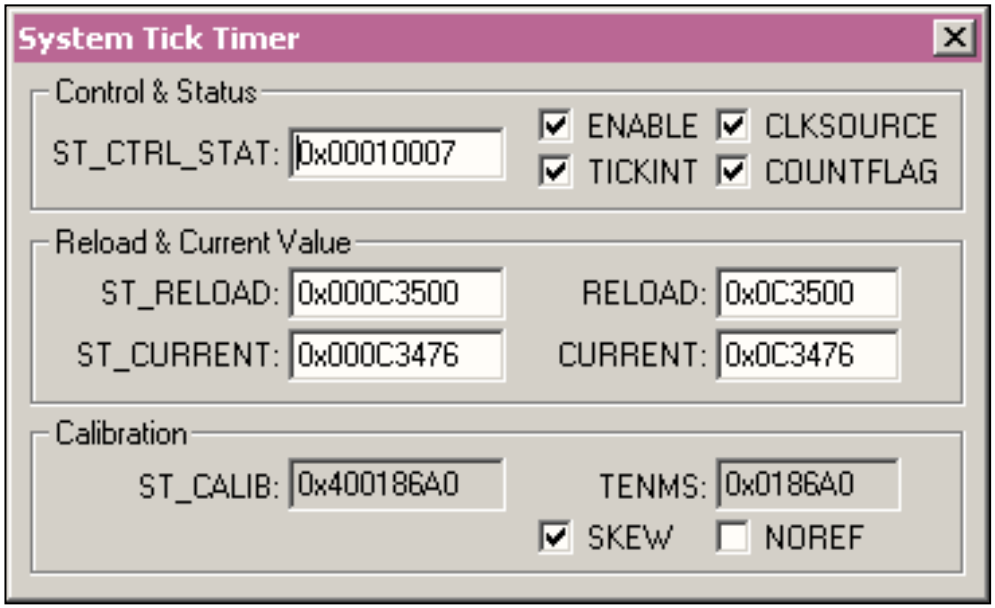
- System Control Register (**0xE000ED10**)
  - Dialog window for setting bits for:
    - SEVONPEND
    - SLEEPDEEP
    - SLEEPONEXIT
- Configuration Control Register (**0xE000ED14**)
  - Dialog window for setting bits for:
    - NONBASETHRDENA
    - USERSETMPEND
    - UNALIGN_TRP
    - DIV_0_TRP
    - BFHFNMIGN



Debugging (Keil µVision)

Core Peripherals

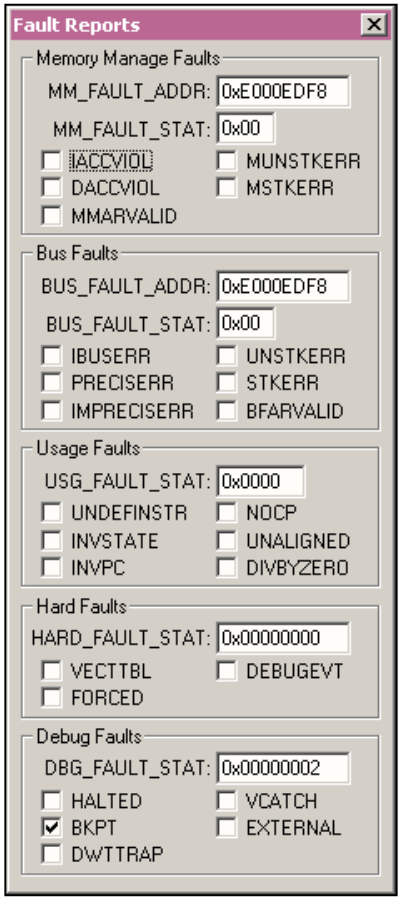
- System Tick Timer (**`0xE000E010`**)
  - Dialog window for setting/watching:
    - Control & Status
    - Reload & Current value
    - Calibration



Debugging (Keil µVision)

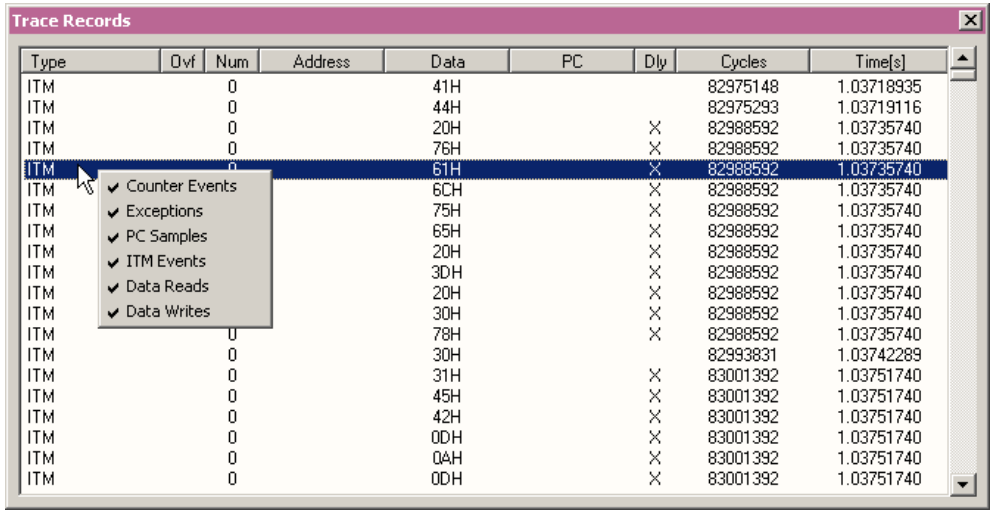
Core Peripherals

- Fault Reports
  - Dialog window for setting/watching:
    - Memory Manage Faults
    - Bus Faults
    - Usage Faults
    - Hard Faults
    - Debug Faults



Trace (ULINK ME)

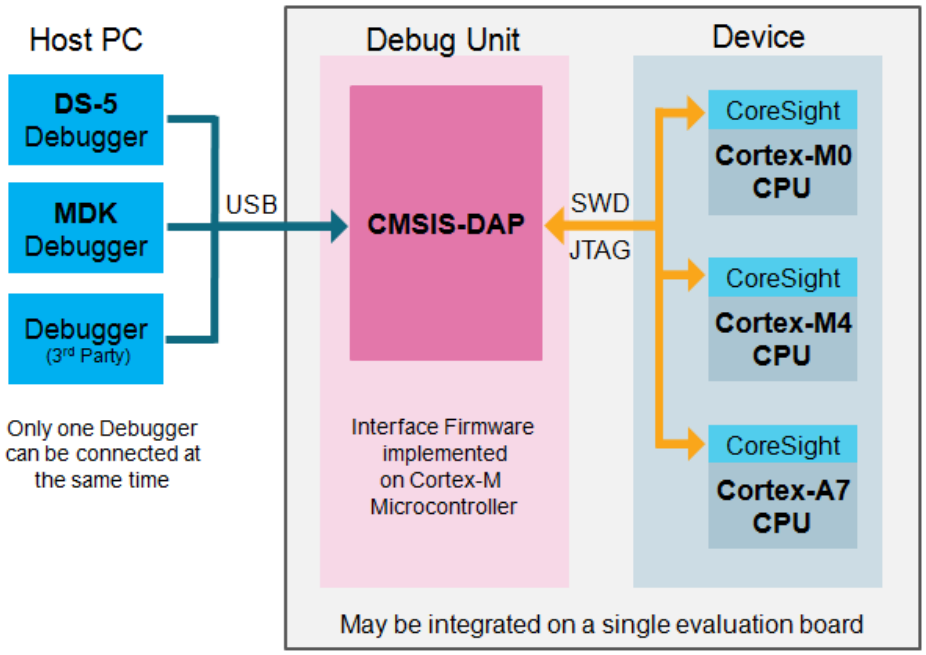
- Trace via ITM
  - Simple Trace views via Instrumentation Trace Macro is supported by µLINK ME
    - Records
    - Exceptions
    - Counters



Debugging with CMSIS-DAP

Definition (ARM-Webpage)

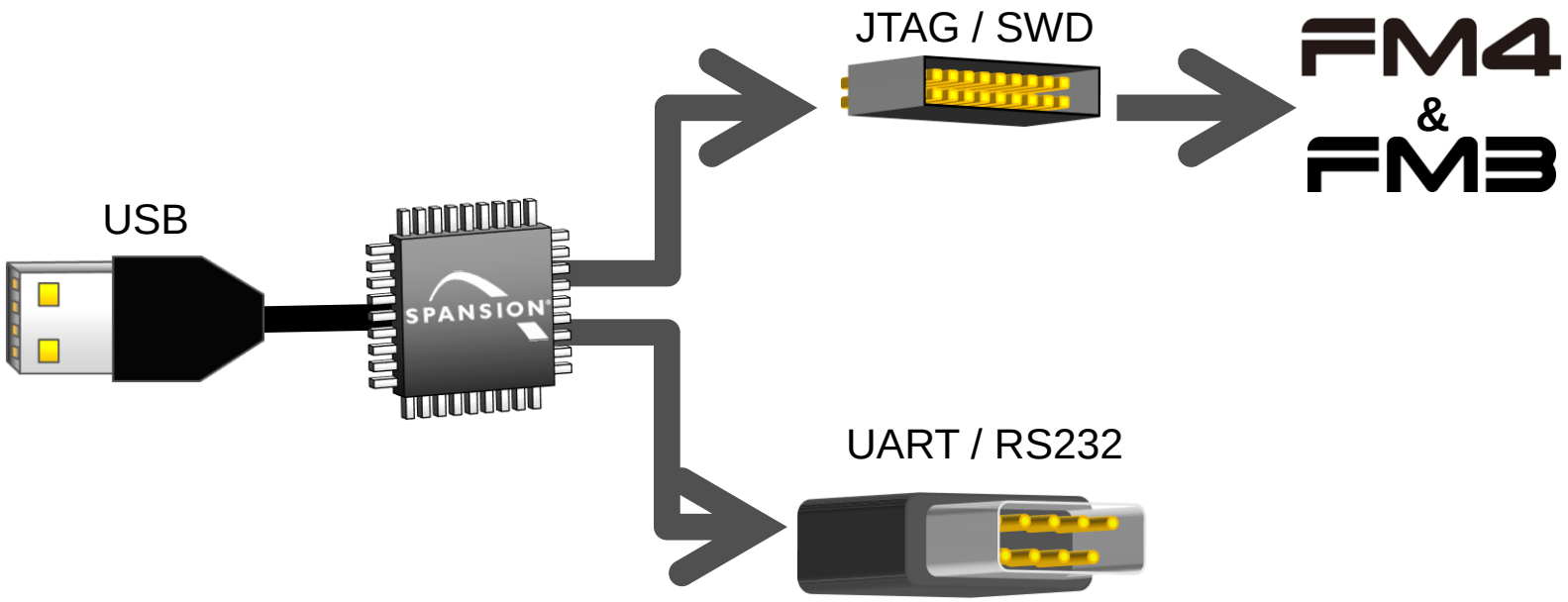
- CMSIS-DAP is the interface firmware for a Debug Unit that connects the Debug Port to USB. Debuggers, which execute on a host computer, connect via USB to the Debug Unit and to the Device that runs the application software. The Debug Unit connects via JTAG or SW to the target Device. ARM Cortex processors provide the CoreSight Debug and Trace Unit. CMSIS-DAP supports target devices that contain one or more Cortex processors.



Debugging with CMSIS-DAP

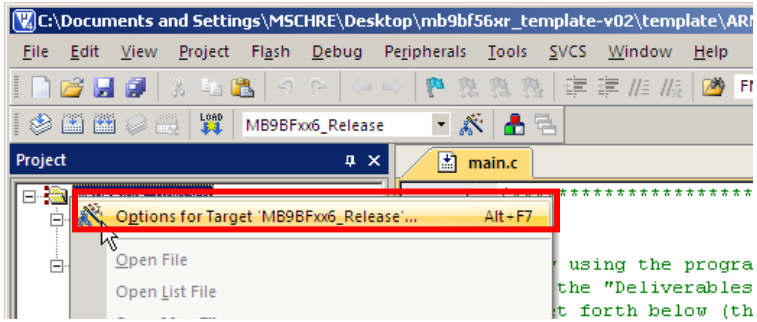
Additional Features

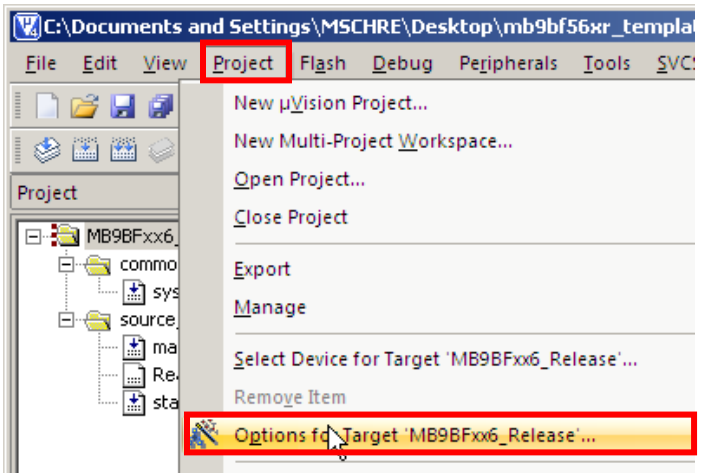
- Spansion CMSIS-DAP implementation offers
  - 1 channel JTAG or SWD (Single Wire Debug)
  - Additional 1 channel UART / RS232



Setup in Keil µVision (1)

- Navigate to project options:
  - Via Project
    - Right-click at the project
    - Select „Options...“
  - Or via menu „Project“
    - Select „Options...“

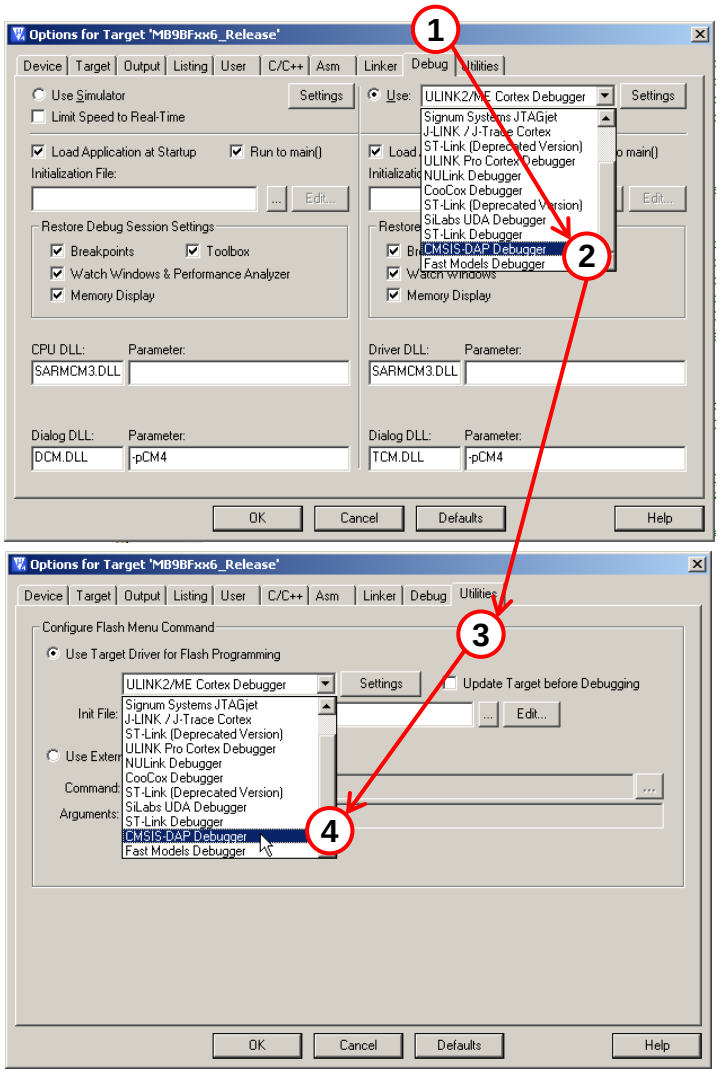




See also Chapter Debugging (Keil µVision)

Setup in Keil µVision (2)

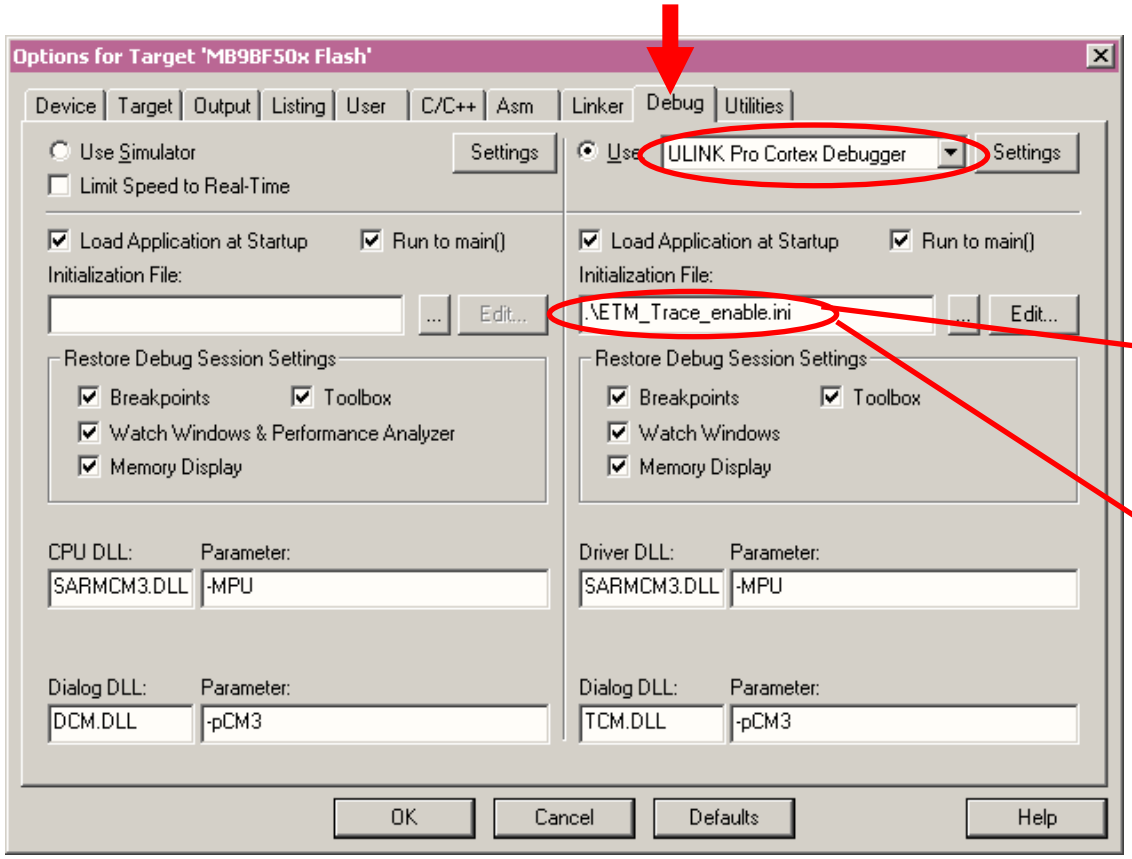
- Setup Debug & Utilities
  - (1) Select tab „Debug“
  - (2) Select „CMSIS-DAP Debugger“
  - (3) Select tab „Utilities“
  - (4) Select „CMSIS-DAP Debugger“

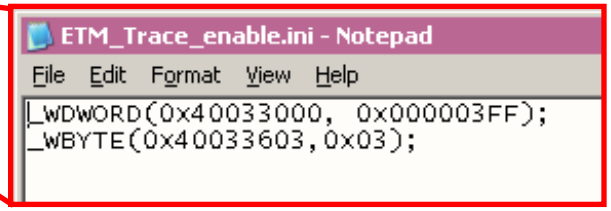


See also Chapter [Debugging (Keil µVision)]

ULINK Pro

- Trace via ETM
  - Check settings in menu:
    Flash→Configure Flash Tools... Tab:*Debug*



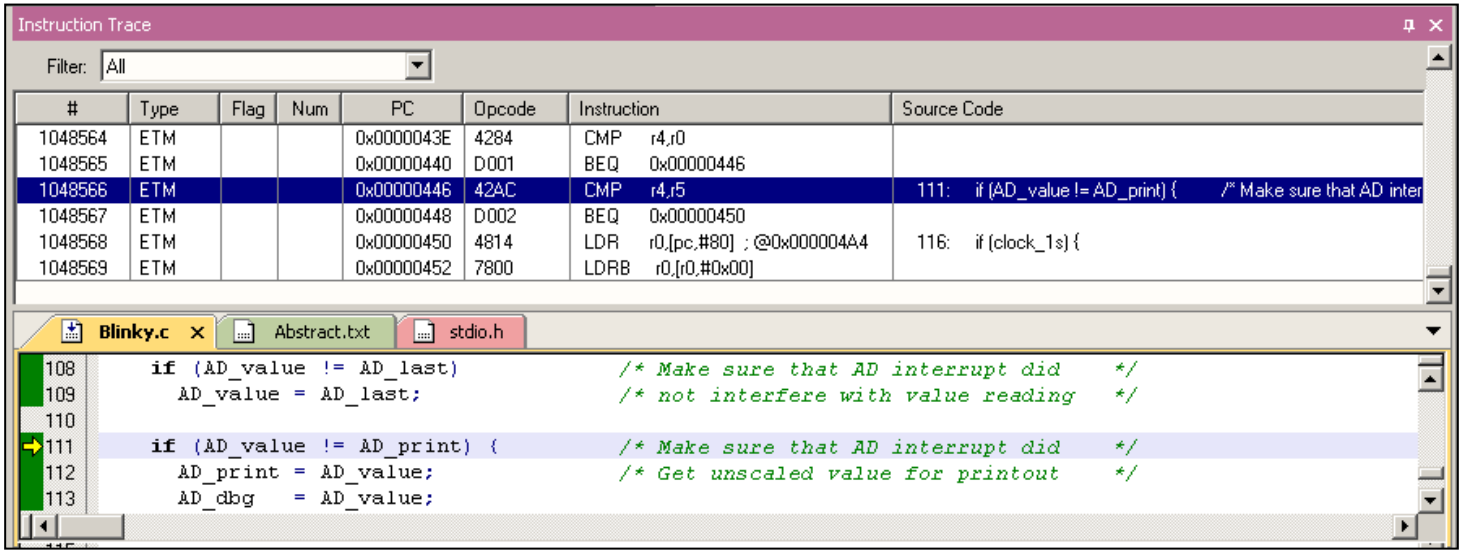


enables ETM pins

Debugging (Keil µVision)

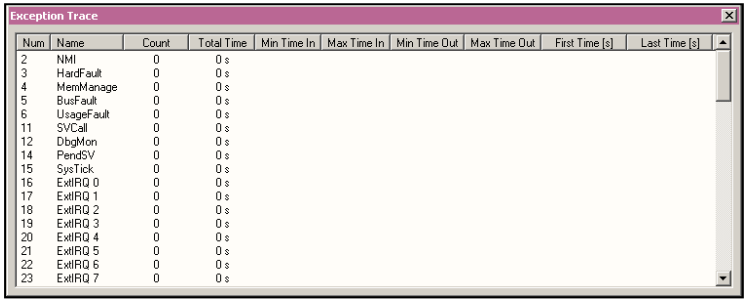
ULINK Pro

- Instruction Trace
  - Real Time Trace recording
  - Output can be filtered by several ETM and ITM events
  - Trace buffer is held in PC memory and transfered to µVision on break



Trace Exceptions

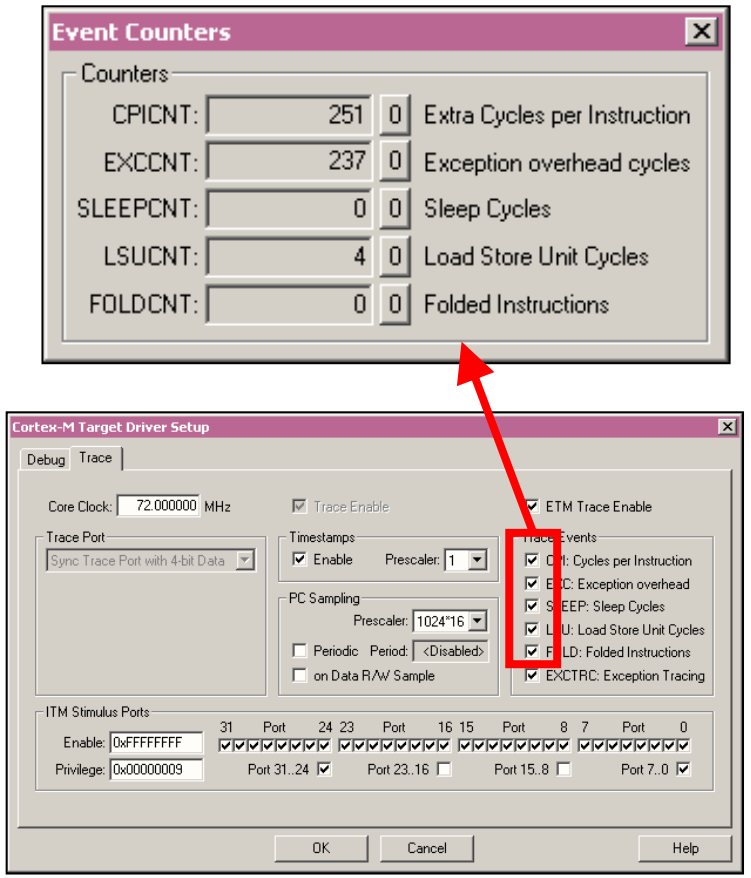
- Exception Trace
  - Exception Trace has an own window, which can‘t be docked
  - Exception Types:
    - System Exceptions
    - User Exceptions (Interrupts)
  - Exception Trace:
    - Count
    - Total Time
    - Several Points in Time: Min., Max., First, Last, etc.
  - Must be configured in: *Debug→Debug Settings...* Tab: *Trace/Trace Events*



Debugging (Keil µVision)

Event Counters

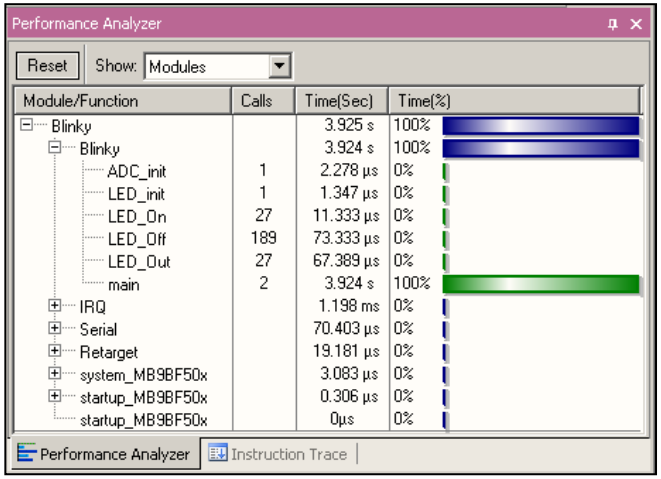
- Following counters are available:
  - Extra cycles per Instruction
  - Exception overhead cycles
  - Sleep cycles
  - Load store unit cycles
  - Folded instructions
- Counters count from 0 to 255
  - Continue counting on overflow starting over from 0
- Must be configured in:
  Debug→Debug Settings...
  Tab: *Trace/Trace Events*
  (Checkboxes)



Debugging (Keil µVision)

Performance Analyzer (ULINK Pro)

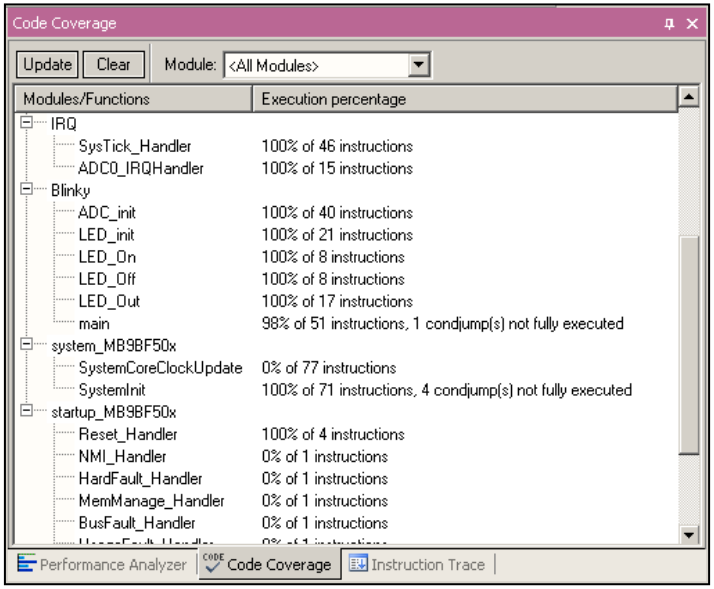
- Shows the execution of any function of an application in:
  - Number of calls
  - Execution time (absolute)
  - Execution time (percentage)
- It shares with a tab the Instruction Trace window and Code Coverage
- Module or function view is supported



Debugging (Keil µVision)

Code Coverage

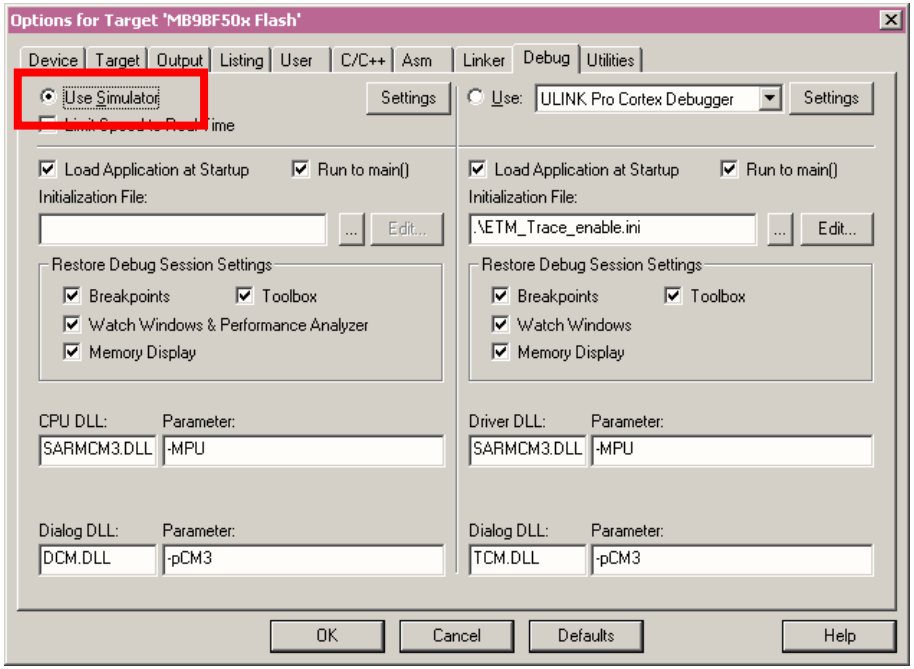
- The Code Coverage shows the function execution by percentage of the function code
- It shares with a tab the Instruction Trace window and Performance view
- All or only a specific module can be chosen for display



Debugging (Keil µVision)

Simulator

- The Core Simulator can be selected by the menu: *Flash→Configure Flash Tools...* and then choosing *Use Simulator*
- Look & feel is like using ULINK debugger
- Controlable also with **.ini* files



www.spansion.com