A Guide to

UDE® Universal Debug Engine

Programming on-chip FLASH and external FLASH memory

Integrated Development Environment
for 64-, 32-, 16-bit Microcontrollers and Embedded Processors

AURIX, TriCore, Arm Cortex-M/R/A, Arm7/9/11, S32G/S/V,
Stellar G/P/E, RH850, R-Car, RISC-V, ARC, Power Architecture

pls
Development Tools

This manual contains 49 pages.

Contact us at:

EMail: support@pls-mc.com
info@pls-mc.com
WWW: https://www.pls-mc.com/

PLS Programmierbare Logik & Systeme GmbH
Technologiepark Lauta
DE-02991 Lauta
Germany
Phone: + 49 35722 384 - 0

PLS Development Tools
10080 N. Wolfe Rd., Suite SW3-200
Cupertino, CA 95014
USA
Phone: +1-949-863-0327
Toll Free: +1-877-77-DEBUG

Contents

Introduction

Overview

UDE® Memtool 2025 is designed for On-Chip and On-Board FLASH/OTP programming with microcontroller systems using the 32-bit architectures AURIX™, TriCore™, S32G, S32S, Power Architecture™, Cortex™-M/R/A, ARM™-7/9/11, RH850, SuperH™ SH-2A and using the 16-bit architectures C166, ST10, XC166, XC2000, XE166, C166CBC, C166S V2 derivatives.
UDE® Memtool is a subset of UDE® Universal Debug Engine and uses the same version numbering scheme.

The separated debugging tool **UDE® Universal Debug Engine** features the on-chip or external FLASH / OTP EPROM programming directly from within the Debugging Environment via the UDE® Memtool add-in.

This manual describes the **UDE® Memtool** based on an evaluation board with a ST10F269 microcontroller. However, the UDE® is also working with the other AURIX, TriCore, Power Architecture, Cortex-M/R/A, ARM-7/9/11, RH850, SH-2A, C166, ST10, XC166, XC2000, XE166, and XScale based hardware. Please see the compatibility list in this manual below or the actual list on our Web site for supported MCU's.

You are invited to browse to our Web site at **https://www.pls-mc.com/** to get the newest information or to download the latest version of **UDE® Memtool**.

Feedback

PLS welcomes feedback on our products and documentations. If you have any comments, suggestions or improvements about the products you are using, please use the Feedback Form from our Web Site **https://www.pls-mc.com/**, send an email to **support@pls-mc.com** or call our Support Line.

Safety Instructions for Products and Equipment

Warning! It is critical that you read and follow this safety advice, the product description including technical data and the associated technical documentation. Do not use the product if you cannot read and/or understand the Information for safe operation. If you do have questions for safe operation, please contact the PLS support at support@pls-mc.com.

This PLS product enables users to control systems which accomplish safety functions (e.g., in electronic control systems), to change safety relevant data, or to allocate those for further processing. Hence, the application of this product can be hazardous. Improper use and unskilled application without adequate instruction and experience in handling of such products may cause threats to life and physical conditions as well as damages to property.

Our products have been developed and released exclusively for use in applications defined in the product description.

Fitness and suitability of the products for any intended use beyond the utilization for which the products have been released (e. g. other stresses/strains or technical conditions) need to be verified by the user on his own authority by taking appropriate actions and measures (e. g. by means of tests).

- PLS products made available as **beta versions** of firmware, hardware and software are to be used exclusively in testing and evaluation. These products may have not sufficient technical documentation and may not fulfill all requirements for quality and accuracy for market released series products. Therefore, product performance may differ from the product description and your expectations. The product should be used only in controlled test environments. Do not use data and results from **beta versions** without prior and separate verification and validation and do not pass them to third parties without prior examination.
- Do not use this product if you do not have proper experience and training in using the product.
- Data of any kind, which have been identified or collected by using PLS products, have to be verified with respect to reliability, quality and suitability prior to any use or dissemination.
- When using this product with systems which accomplish safety functions (e.g., in electronic control systems) that influence system behaviour and can affect the safe operation of the system, you must ensure that the system can be transitioned to a safe condition (e.g. emergency shutdown or emergency operation mode) if a malfunction or hazardous incident should occur.
- All applicable regulations and statutes regarding operation must be strictly followed when using this product
- It is recommended to use the products only in closed and designated test environment.

Warning! If you fail to follow this safety advice, there might be a risk of death, serious injury or property damage. PLS and their representatives shall not be liable for any damage or injury caused by improper use of the product. PLS provides trainings regarding the proper and intended use of this product.

Regulatory Compliance and Compliance Statements

The UADx hardware is in conformity with the protection requirements of the EU Council Directive EMC 89/336/EWG, EMC 2004/108/EC, EMC 2014/30/EC. The UADx hardware has been tested and found to comply with the limits for Class B Information Technology Equipment according to the European Standard EN 55022, EN 55024.

The UADx hardware complies with the relevant provisions of the RoHS Directive for the European Union.

Software

- Install the software only on systems which fulfill the minimum requirements both in hard- and software.
- For installation of the software administrator rights are required to copy files in directories which are protected by the Windows OS, to install device drivers and modify the registry.
- The software enables the in-depth control of embedded systems. It should only be operated by persons who have the necessary expertise in the systems.
- Incorrect usage of the software can lead to irreparable destruction of components in the connected systems. This concerns in particular components whose integrated permanent memory (FLASH, PCM) is protected by special mechanisms.
- There is a particular danger if mechanical devices such as motors or actuators are controlled by the embedded systems. In this case, all necessary precautions must be taken to avoid accidents, e.g., emergency shutdown.
- There is also a particular danger if the embedded systems switch voltages that exceed the permissible contact voltages. In this case, all precautions must be taken to avoid accidents, e.g., insulation.

Electrical Safety Instructions

The UDE® Universal Debug Engine shall only be used according to the installing instruction of the **UDE Manual.pdf** and **UDE Manual Appendix.pdf**. Any external power supply used with the Universal Access Device (UAD2^{pro}, UAD2^{+}, UAD2^{next}, UAD3^{+} ...) and its components shall comply with the relevant regulations and standards applicable in the country of intended use.

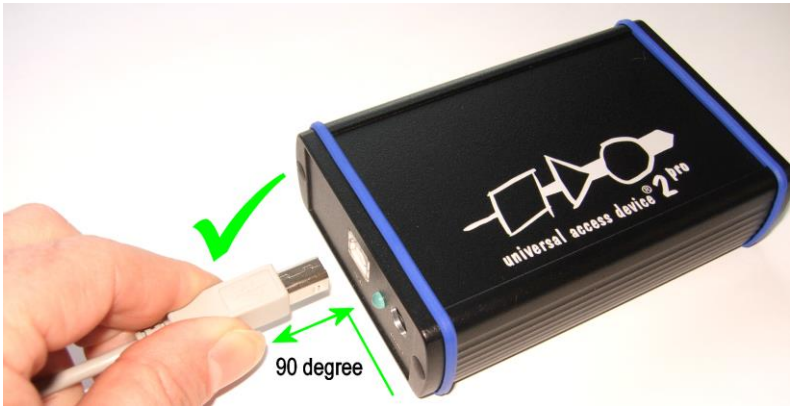
Please observe the following safety instructions when using the power supply:

- Always use the supplied power adapter, and connect it to an AC outlet of the rated voltage and frequency. If an AC adapter other than those specified by PLS is used, it may result in damage to the UADx and its accessories or AC adapter, fire or electric shock.
- Do not insert or disconnect the AC plug with wet hands. Doing so may cause electric shock.
- Insert the power plug fully and securely. Incomplete insertion may cause fire or electric shock.
- The power supply unit should be connected to an easily accessible socket outlet in the immediate vicinity of the unit.
- Always disconnect the power cord by holding the power plug. Pulling the power cord itself may damage it and cause fire or electric shock.
- Ensure that the device connections do not come into contact with liquids and do not touch them with wet or greasy hands or metal objects. If liquid gets into the device, stop using the device immediately and contact support@pls-mc.com.
- Do not store the devices in environments with high humidity or where the temperature may change suddenly. If condensation has formed, switch the devices off immediately and wait until all water drops have evaporated.
- Do not pour liquid substances over the UADx and its accessories or drop other objects on it, this could cause serious damage to the UADx and its components. If this should happen please stop all work with the UADx and its accessories immediately and contact support@pls-mc.com.
- Do not disassemble or attempt to repair the equipment. If a device is damaged, stop using the device immediately and contact support@pls-mc.com. Do not touch damaged areas. Avoid contact with eventually spilled liquids.

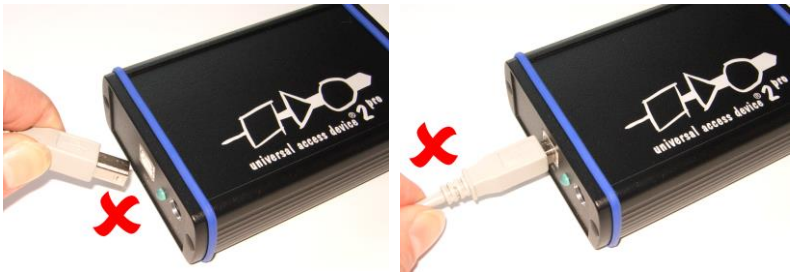
- If the UADx and its accessories is visibly damaged or its functionality is limited, it must not be used without prior instruction from support staff (support@pls-mc.com). Especially if components are damaged where voltage is flowing through them. These must be replaced by the manufacturer in order to avoid hazards.
- Unplug the power cord from the wall outlet during a thunderstorm or prolonged absence! Otherwise, damage to the unit could be caused by overvoltage.

Mechanical Safety Instructions

- Hold the head of the USB cable with your index finger and thumb on both sides and **insert** the cable straight into the USB port as shown in the illustration below. Make sure that you insert it straight and not at an angle.
- Hold both sides of the USB cable with your index finger and thumb at the point where it is connected to the computer and carefully **pull it out** horizontally to remove the cable from the USB port.



- Do not insert or remove a USB plug with excessive force.



- Do not plug in or pull out the USB plug upwards, downwards, left, right or forwards.
- Do not pull or tug on the USB cable when plugged into the port.

Safety Instructions

- Do not use the Universal Access Device (UAD2^{pro}, UAD2^{+}, UAD2^{next}, UAD3^{+} ...) and its accessories in places where flammable or combustible gases (gasoline etc.) are present. Doing so may cause a fire.
- The UADx and its components should be operated in a well-ventilated environment and should not be covered. The UADx and its accessories are only intended for use inside buildings.
- The UADx and its components should be placed on a stable, flat surface in use.
- Do not use excessive force when using the equipment. Do not pull on cables or bend them too much.
- Do not expose the devices to fire, microwaves or high temperatures.
- The UADx and its accessories must not be operated if it is damaged, or if smoke or odd smells occur. Doing so may result in a fire. In such situations, disconnect the power adapter from the AC outlet, and contact support@pls-mc.com.
- Make sure that the UADx and its accessories is stored at ground level and in a position that does not endanger persons and surrounding equipment.
- Do not place the UADx and its accessories on an unstable or sloping surface. Doing so may result in its dropping or overturning, causing injury. Be careful not to drop the UADx and its accessories when carrying it.
- Before cleaning, remove all connected cables to avoid the risk of electric shock. Clean the outside of the devices only, using a soft, damp cloth. Do not use chemicals or abrasives. Avoid under all circumstances the penetration of moisture into the device.
- The use of spare parts, accessories and special equipment which have not been tested and approved by PLS can have a negative influence on the function and properties of the UADx and its components. Therefore, PLS is not liable for any resulting damage.
- Improper operation of the UADx and its accessories may cause damage to the devices or other property. It may therefore only be used in technically perfect condition and for its intended purpose in accordance with the operating instructions given in the manual.
- Safe use of UADx and its accessories is only possible if the user manual is read completely and the instructions are followed completely. Non-observance of the instructions can lead to considerable damage or accidents.
- Anyone using UADx and its accessories must have access to the user manual. The user manual can be found here: in the delivery content of the UDE® as printed manual, UDE® Software installation as PDF.
- **Keep these operating instructions in a safe place for later use.**
- The product may only be used by persons instructed in the safe use of the product and understand the resulting dangers.
- Children should be supervised to ensure that they do not play with the UADx and its components.
- Keep the devices, all accessories and packaging materials out of reach of younger children and pets. Small objects such as the packaging materials could be accidentally swallowed. Cables could be tied around the neck.

System Requirements

To run **UDE® Memtool 2025** at least the following minimum system configuration is required:

	Minimum	Recommended
CPU	Intel or AMD x86_64 (64-bit) processor	Intel Core i7™ or AMD R7 processor
RAM	4 GByte	8 GByte
Free disk space	2 GByte HDD	8 GByte SSD
Display	SXGA	WUXGA
Operating System	Windows®10 64-bit or Windows®11 64-bit	Windows®10 64-bit or Windows®11 64-bit

Dependencies

- Microsoft Visual C++ 2015-2022 Redistributable (x64)

Note: Installations packages of these components are include with and installed by UDE® setup package. However to avoid side effects on other applications these components are not uninstalled when UDE® is uninstalled.

Additional requirements

- Optional: CD-, DVD- or BD-drive for installation from CD-ROM
- Microsoft .NET™ Framework 4.8.x
- Microsoft Windows® Scripting Host V5.6 or higher required for scripting support
- Microsoft Internet Explorer® 10 or higher
- Adobe® Acrobat Reader 10 or higher
- Administrator permissions for the current login during installation.

Depending on the type of target access you will additionally need one of the following interface ports:

- an USB port interface for the Standard version with UAD2^{pro} or UAD2^{+} or UAD2^{next} or UAD3^{+} or for the Demo version for Easy Kits XC166, XC2000, XE166 or for the Simulator version with USB-Key or the Standard version with USB-JTAG-Port
- or an Ethernet interface for the Standard version UAD2^{+} or UAD2^{next} or UAD3^{+}
- or an IEEE1394-OHCI interface for the Standard version with UAD2^{+} or UAD3^{+} via IEEE1394 or an IEEE1394b-OHCI interface for the Standard version with UAD3^{+} via IEEE1394b
- or a standard serial port interface for UDE® Memtool using the Host PC serial COM port (ASC supported).

When a new version of UDE® Memtool is started the first time a firmware update may be executed on the access device (UAD2pro, UAD3+, ...). This may take some more time than usual for the target connect operation. Please **DO NOT** power off or unplug the access device while this time!

Versions of UDE® Memtool

UDE® Memtool .. comes with a separate front-end interface as standalone tool outside of UDE®. All programming functions are available also via standard COM automation interfaces. Using these interfaces, the features of Memtool may be integrated into automatic production and test systems or can be executed via scripts and batches.

UDE® Memtool Add-In .. is a part of the UDE® Universal Debug Engine and allows the FLASH/OTP programming during the development cycle inside of UDE®. Memtool observes the download path and runs the programming process when requested.

General Features

- High-speed data communication by using the debug channel between target system and the debugging tool UDE® Universal Debug Engine based on the Universal Access Device Communication Hardware or Host interfaces
- Automatic adaptation of data to be programmed to the smallest memory block size by automatic read operations
- **Easy-to-Use**: FLASH programming is integrated in the debugger-to-target program download in the UDE® Memtool Add-In. User created front-ends are possible with the UDE® standalone Memtool.
- **Transparent Erase Mode**: previous bank read and bank erase simulate random access for non-bytewise erasable FLASH memories.

Supported Functions

UDE® Memtool offers following functions (depending on the type of memory):

- Erasing the entire memory module
- Erasing selected sectors of the memory module
- Loading Intel Hex and Motorola S-Record files
- Programming all or selected portions of the file into the memory module
- Comparing all or selected portions of the file to the current contents of the memory module
- Read back and save-to-disk of the current memory content
- Setting and Resetting the Chip/Sector Protection (On-Chip only)
- Calculating CRC sums of sections
- Watchdog handling for some derivatives
- **UCB** (User Configuration Blocks) , ABM (Alternate Boot Mode) , BMI (Boot Mode Index) header handling for some derivatives
- **ABM** (Alternate Boot Mode) header handling for some derivatives
- **BMI** (Boot Mode Index) header handling for some derivatives
- Support of Protection Function
- Support of the single-chip Reset Mode of the new C16x derivatives.

UDE® Memtool can handle more than one memory module on the target system and is only using on-chip RAM for execution (IRAM; XRAM when available). UDE® Memtool supports the single-chip reset mode of the new C166 derivatives.

Supported Microcontroller with On-Chip FLASH/OTP Memories

- AURIX and TriCore
- Cortex-M0, Cortex-M3, Cortex-M33, Cortex-M4, Cortex-M7, Cortex-R4, Cortex-R52, Cortex-R5F, Cortex-A53, Cortex-A72, Cortex-A9, Cortex-A8, XMC1000, XMC4500
- RH850 G3K, G3M, G4M
- RISC-V
- SuperH SH-2A
- Synopsys ARC
- ARM7, ARM9, ARM11
- Power Architecture and PowerPC
- XC2000, XE166, XC166, XC800, C166, ST10Fxx.

See complete list at website https://www.pls-mc.com/.

Supported External FLASH Memories

- Parallel-NOR-FLASHs (AMD AM29x, M29x, Intel i28Fx)
- Serial EEPROMs (I2C, SPI, SPIFI)
- NAND-FLASH (on demand)
- Further devices under development or on request.

It is possible to extend the database of the supported external FLASH memories by the user. For the list of supported external memory devices, see the file `EXTFLASH.DAT` within the UDE® installation folder. If newer FLASH devices are available, which are not supported by UDE® Memtool, please contact the **PLS Support Team** at support@pls-mc.com to receive a newer version of the FLASH database.

Please see the FAQ list of UDE® on the PLS' website https://www.pls-mc.com/.

Communication between Host PC and Target System

UDE® Memtool supports the communication channels ASC, SSC, CAN, DXCPL and JTAG via Access Devices and can use the ASC bootstrap loader and CAN bootstrap loader.

The UDE® standalone Memtool offers also the host-target communication via ASC communication and a standard serial RS232 host interface (COM). Beside RS232, the usage of the K-Line interface is possible.

Communication via JTAG

- Supported for target microcontrollers of the AURIX, TriCore, Power Architecture, Cortex, SuperH SH-2A, RH850, ARM7, ARM9, ARM11, XC166 and XC2000 families.
- Communication channel via JTAG/DAP/SWD interface supported.
- Uses only on-chip memory for programming. Single-chip systems without external RAM are supported.

Communication via DXCPL (DAP over CAN Physical Layer)

- Supported for target microcontrollers of the AURIX, TriCore families
- Communication channel via JTAG/DAP/SWD interface supported.

ASC-Bootstrap loader via mini monitor

- Supported for all target microcontrollers of the AURIX, TriCore, Power Architecture, Cortex, C166, ST10, XC166, XC2000, XE166 families.
- Communication channel via ASC interface supported.
- Communication on Host PC via serial COM port with up to 57600 Baud supported, optionally via Universal Access Device up to 156 kbit/s.
- Uses only on-chip memory for programming. Single-chip systems without external RAM are supported.

CAN-Bootstrap loader via mini monitor

- Supported for AURIX, TriCore, Power Architecture, ST10F276, XC2000, XE166, XC166 derivatives.
- Communication channel via CAN interface supported.
- Communication Universal Access Device.
- Uses only on-chip memory for programming. Single-chip systems without external RAM are supported.

Overview about target communication

Target Controller	Communication Channel	Monitor	Universal Access Device 2+	Universal Access Device 2pro	Universal Access Device 3+	Host Serial ASC
AURIX, TriCore	ASC-BSL	Mini	✓	✓		✓
AURIX, TriCore	CAN-BSL	Mini	✓	✓		
AURIX, TriCore	JTAG/DAP	w/o	✓	✓	✓	
AURIX, TriCore	DXCPL	w/o	✓	✓		
PowerPC	ASC-BSL	Mini	✓	✓		✓
PowerPC	CAN-BSL	Mini	✓	✓		
PowerPC	JTAG	w/o	✓	✓	✓	
XMC1000/4000	ASC-BSL	Mini	✓	✓		✓
XMC1000/4000	CAN-BSL	Mini	✓	✓		
STM32	ASC-BSL	Mini	✓	✓		✓
STM32	CAN-BSL	Mini	✓	✓		
Cortex	JTAG/SWD	w/o	✓	✓	✓	
ARM7	JTAG/SWD	w/o	✓	✓	✓	
ARM9	JTAG/SWD	w/o	✓	✓	✓	
ARM11	JTAG/SWD	w/o	✓	✓	✓	
XC166	ASC-BSL	Mini	✓	✓		✓
XC166	CAN-BSL	Mini	✓	✓		
XC166	JTAG	w/o	✓	✓	✓	
XC2000/ XE166	ASC-BSL	Mini	✓	✓		✓
XC2000/ XE166	CAN-BSL	Mini	✓	✓		
XC2000/ XE166	JTAG	w/o	✓	✓	✓	
C166/ST10	ASC-BSL	Mini	✓	✓		✓
ST10F276	CAN-BSL	Mini	✓	✓		
RH850	JTAG	w/o	✓	✓	✓	
SuperH SH-2A	JTAG	w/o	✓	✓	✓	

Installing of UDE® Memtool

Installing Hardware

The UDE® Memtool as part of the Universal Debug Engine can use all communication channels featured by this development workbench.

The hardware installation of the communication hardware is described in the UDE® **Manual Appendix** you have gotten with the UDE® Memtool delivery. Please refer to the chapter 'Installing Hardware' for more information.

When a new version of UDE® Memtool is started the first time a firmware update may be executed on the access device (UAD2pro, UAD3+, ...). This may take some more time than usual for the target connect operation. Please **DO NOT** power off or unplug the access device while this time!

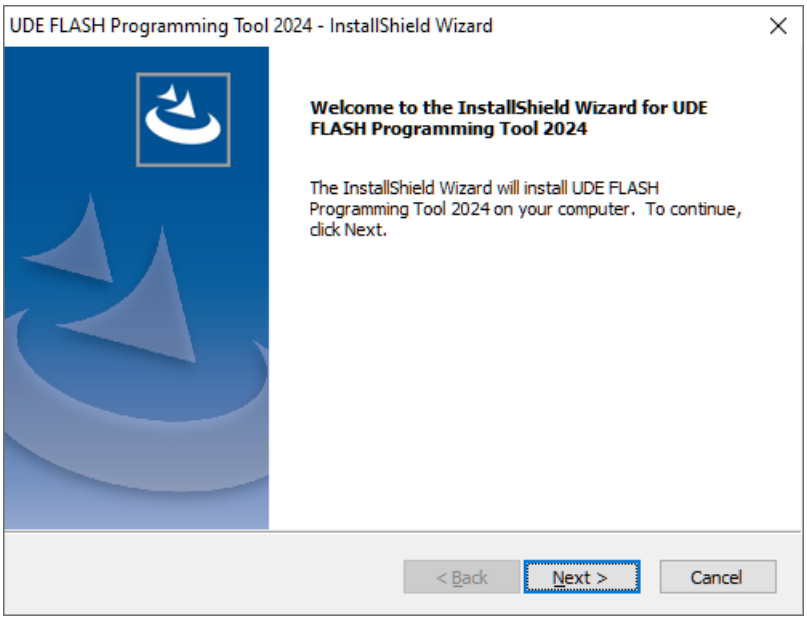
Installing Software

The UDE® Memtool Add-In is a part of the UDE® Universal Debug Engine and won't described at this chapter. If there are question please refer to the UDE® Manual. Furthermore, the installation of UDE® Standalone Memtool will be described.

For installing **UDE® Standalone Memtool** change to the UDE® Memtool CD-ROM.

When a new version of UDE® Memtool is started the first time a firmware update may be executed on the access device (UAD2pro, UAD3+, ...). This may take some more time than usual for the target connect operation. Please **DO NOT** power off or unplug the access device while this time!

1. Read the file `RELEASE.TXT` for the latest software changes.
2. Start `SETUP.EXE` from the UDE® Memtool CD-ROM.



3. Click **Next** to continue the installing process or click **Cancel** for aborting.
4. Accept the terms of license agreement and click **Next - Next**.
5. Choose the destination location installation via the **Browse** button. Please advise an empty or new directory for the UDE® software. Click **Next**.
6. Select the Program Folder and click **Next** to continue.
7. Click **Install** to continue and end the installation process.

The further installation process includes driver installations of the communication devices. Please refer to the **UDE Manual Appendix.pdf** for further information.

Getting Started

Introduction

This tutorial explains the configuration of **UDE® Memtool** for programming the on-chip FLASH module of the supported microcontroller derivatives.

A further description of the UDE® Memtool Add-In can be found in the **UDE Manual.pdf**.

In this example, an evaluation board featuring a **TriCore AURIX TC375** microcontroller is assumed. Communication between the host PC and the target system is established via the JTAG interface. The usage of the DAP interface is described additionally.

When a new version of UDE® Memtool is started the first time a firmware update may be executed on the access device (UAD2pro, UAD3+, ...). This may take some more time than usual for the target connect operation. Please **DO NOT** power off or unplug the access device while this time!

Protection Function

The Protection Function of UDE® Memtool allows to control read and write protection features provided by the FLASH/OTP memory device. Which feature is supported and how it is handled by UDE® Memtool largely depends on the type of memory device.

Supported protection features:

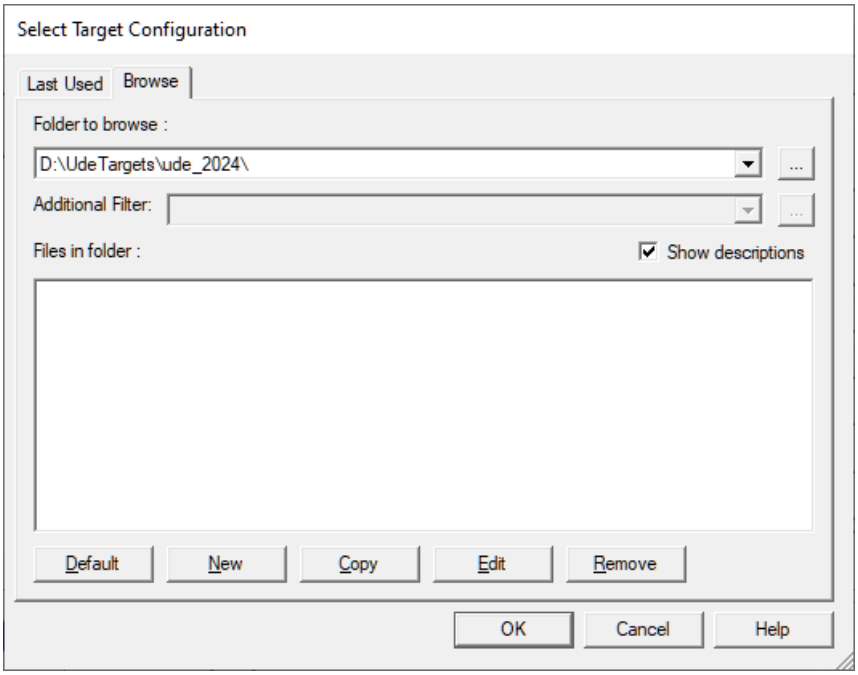
- Global read protection
- Sector write protection.

Programming the On-chip FLASH Module

Starting UDE® Memtool

To launch UDE® Memtool as a stand-alone tool, execute Memtool.exe via the Windows Main menu **Start** - **All Programs** - **UDE FLASH Programming Tool** - **UDE Memtool** or `<UDE_DIRECTORY>\Memtool.exe`.

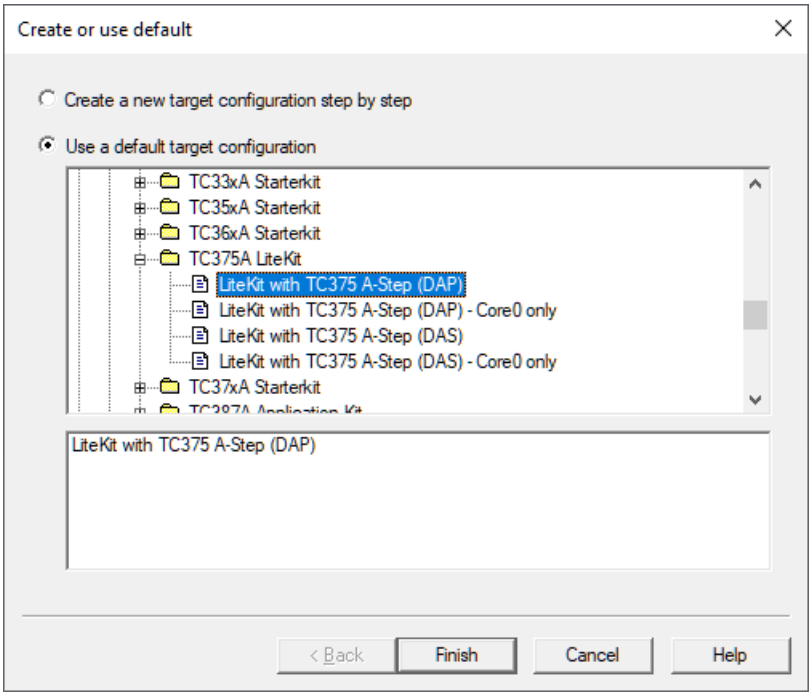
When starting UDE® Memtool for the first time, the **Select Target Configuration** dialog is displayed. Otherwise, this dialog can be reached via menu **Target** - **Change...** .



Using a Default Target Configuration

UDE® Memtool comes with a numbers of default target configuration for Starterkits and evaluation boards. These configurations were created and tested by PLS. Open the **Select Target Configuration** dialog via menu **Target** - **Change...** , click on **Default**, enable **Use a default configuration** and select the microcontroller family and a suitable target configuration.

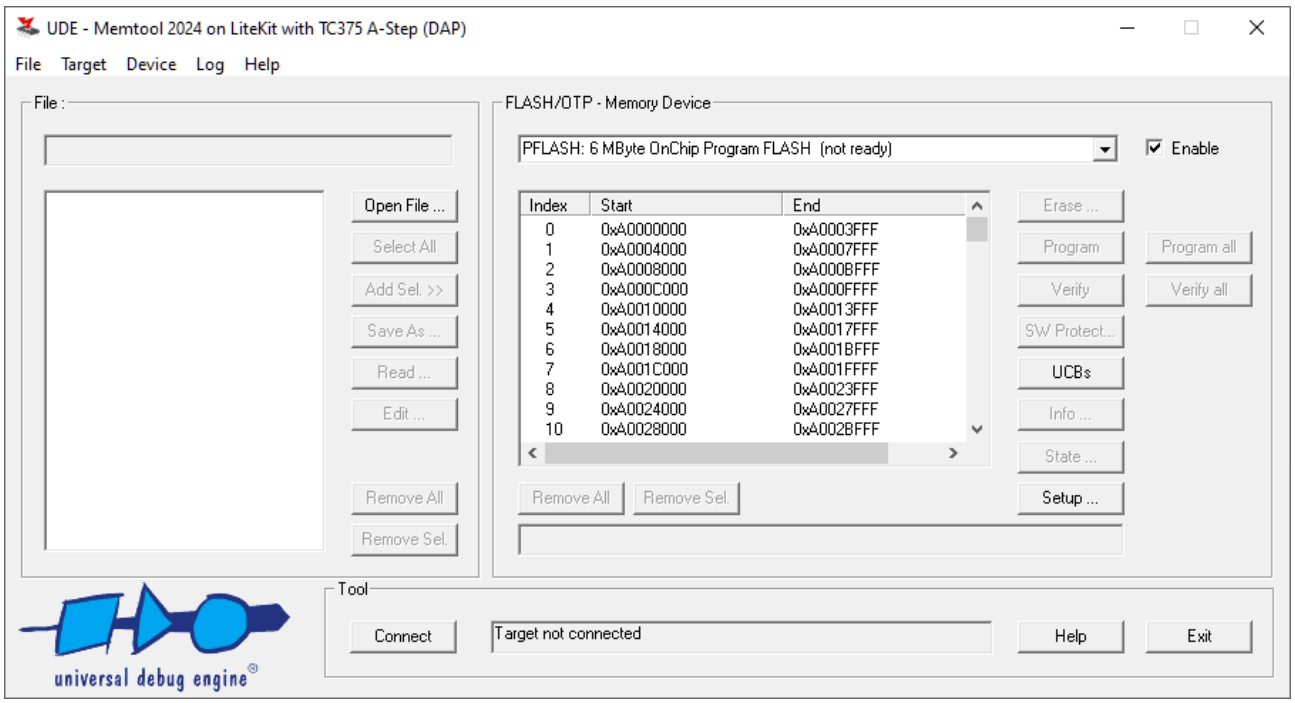
For usage of the **TC375A LiteKit** choose **TriCore Aurix TC3xx - Infineon - TC375A LiteKit - LiteKit with TC375 A-Step (DAP)**.



Click on the **Finish** button to close the selection dialog. Define the path and name of the resulting configuration file.

The location for saving this file is not limited while it is recommended to save configuration files in `<My Documents>\pls\UDE 2025\Targets`. In this example, the configuration file is saved in `D:\UdeTargets\ude_2025\LiteKit_TC375A_dap.cfg`.

Select the target configuration and leave this dialog with **OK** to load the new configuration file. UDE® Memtool will now use this target configuration for the further process.



Creating a New Target Configuration

Alternatively, a new target configuration can be required. For a new configuration file, click on **New**, select **Create a new target configuration** and click on **Next**.

In **Target Description** a short description of the target system has to be entered. This description will be displayed in the **Select Target Configuration** dialog. In this example, **TC375 Evaluation Board** is used.

In the **Family** list box, the microcontroller family can be selected to which the target system's microcontroller belongs to - in this example the **Infineon TriCore Family**.

Click on the **Next** button to change to the **New Target Controller** dialog page.

In the **Type** list box please select the exact microcontroller derivative. In this example, the **TC375A** is used.

Remark: Some microcontroller derivatives show different behaviour between different manufacturing steps. If applicable, in the **Step** list box the manufacturing step can be selected.

Click on the **Next** button to change to the **New Target Master Core** dialog page.

In **Communication Device/Protocol** the type of target access is selected. In this example, the **TriCore2 JTAG/OCDS Interface** is used because communication is established via the JTAG or DAP interface.

Communication between the target and the host PC requires some more configuration settings. Please click on the **Setup** button to open the **TriCore2 JTAG/OCDS Interface Setup**.

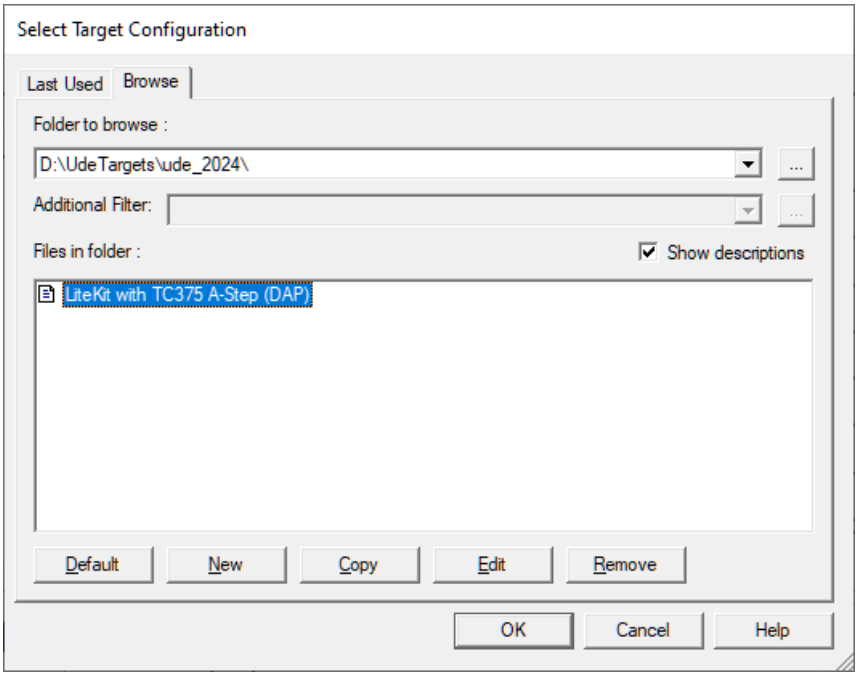
Change to the **General** dialog page. Please select the used communication port/device in **Access Device**. In this example, an UAD2pro is used. When DAP will be used, the **Use DAP for target access** check box must be enabled.

Click on the **Next** button to change to the **New Target Master Core** dialog page. Select **Core0** as **Master** for Core1, CoreX, GTM, ED and so on..

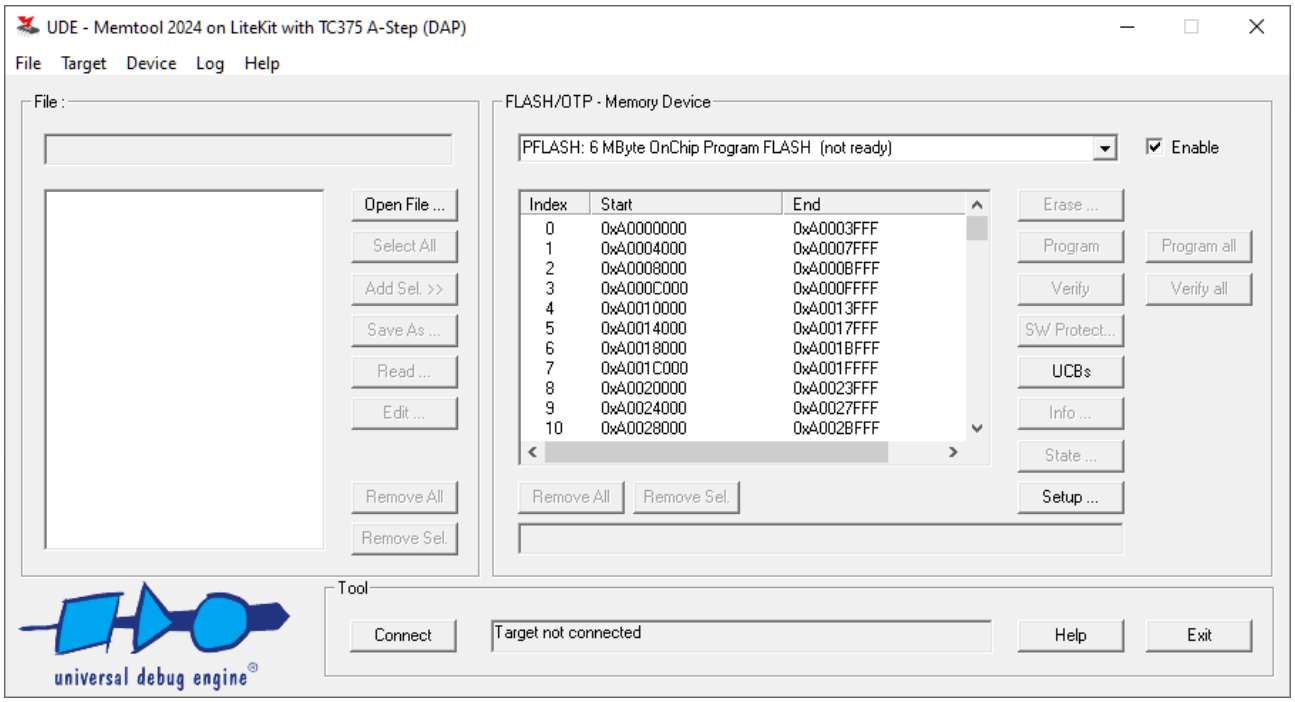
Click on the **Next** button to change to the **New Target Specify Memory** dialog page. Leave the **Number of Memories** as **0**.

Click on the **Next** button to change to the **New Target Finished** dialog page. Enable the memory device which should supported by UDE® Memtool.

Click on the **Next** button to change to the **New Target Configuration** dialog page. Define the path and name of the resulting configuration file. The location for saving this file is not limited while it is recommended to save configuration files in `<My Documents>\pls\UDE X.XX\Targets`. In this example, the configuration file is saved instead in `D:\UdeTargets\ude_2025\LiteKit_TC375A_dap.cfg`.



Select the target configuration and leave this dialog with **OK** to load the new configuration file. UDE® Memtool will now use this target configuration for the further process.



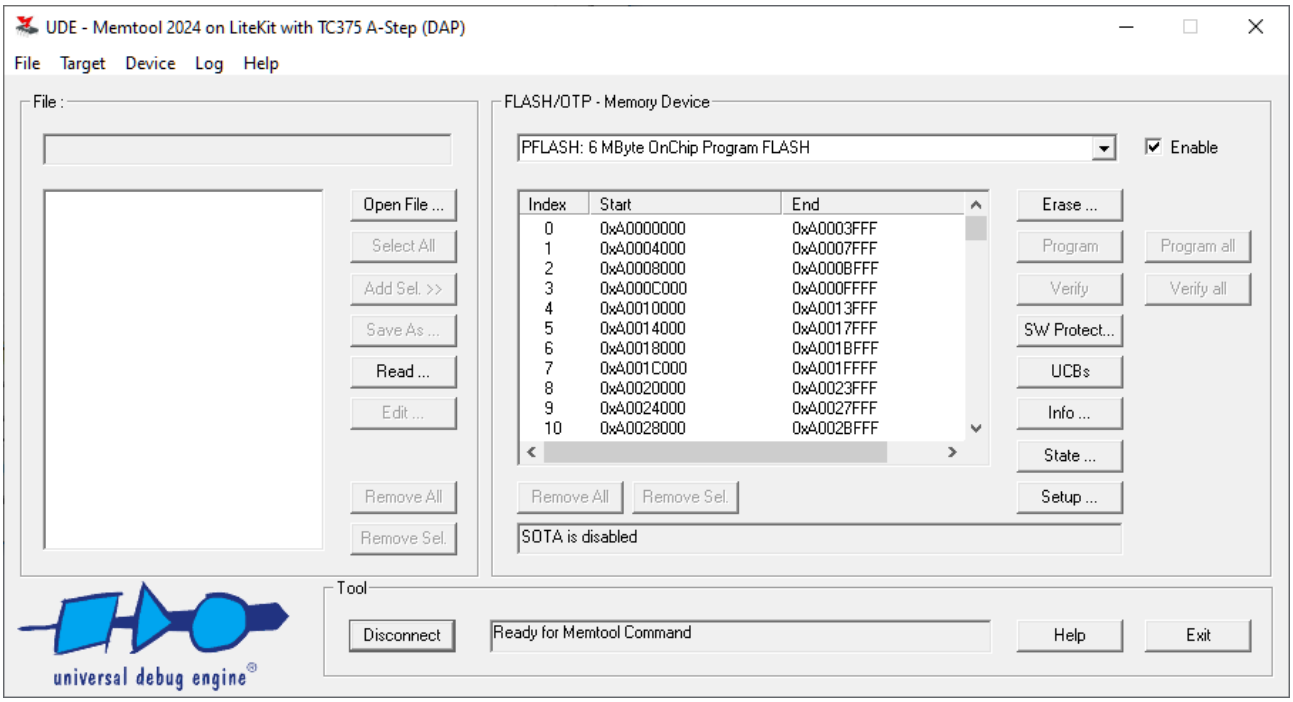
Connecting to the Target

When a new version of UDE® Memtool is started the first time a firmware update may be executed on the access device (UAD2pro, UAD3+, ...). This may take some more time than usual for the target connect operation. Please **DO NOT** power off or unplug the access device while this time!

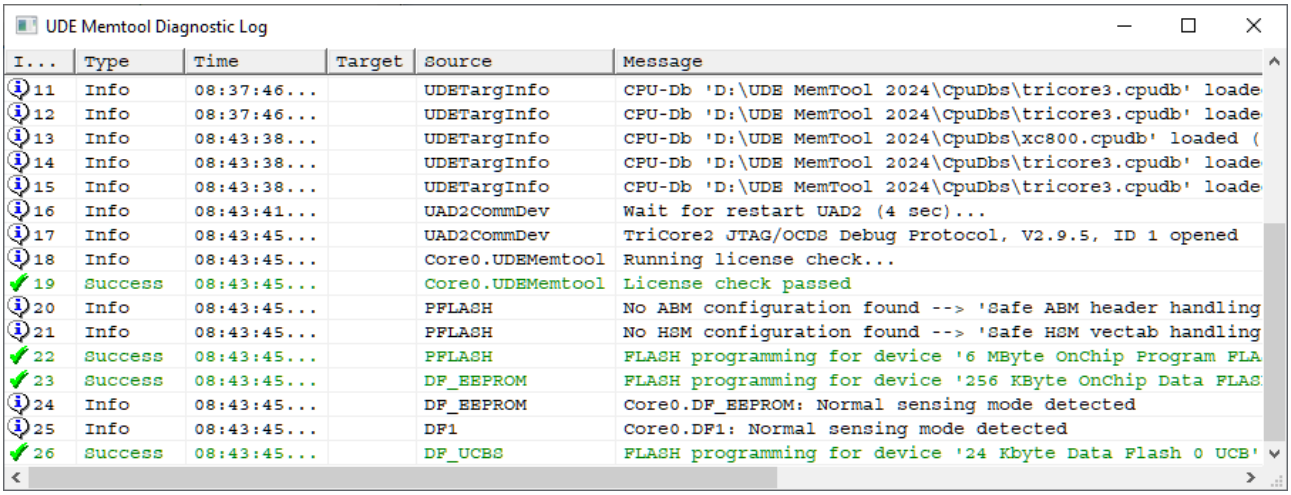
After selecting a configuration file UDE® Memtool is ready to establish a connection to the target system. In the title bar a description of the currently loaded configuration file is displayed.

In the combo box under **FLASH/OTP-Memory Devices** all available FLASH modules are listed. In this example the **PFLASH: 16 MByte on-chip Program FLASH** is available and enabled. The **Sector** list box contains the sector table of the selected FLASH module.

Please click on the **Connect** button to establish connection to the target system.

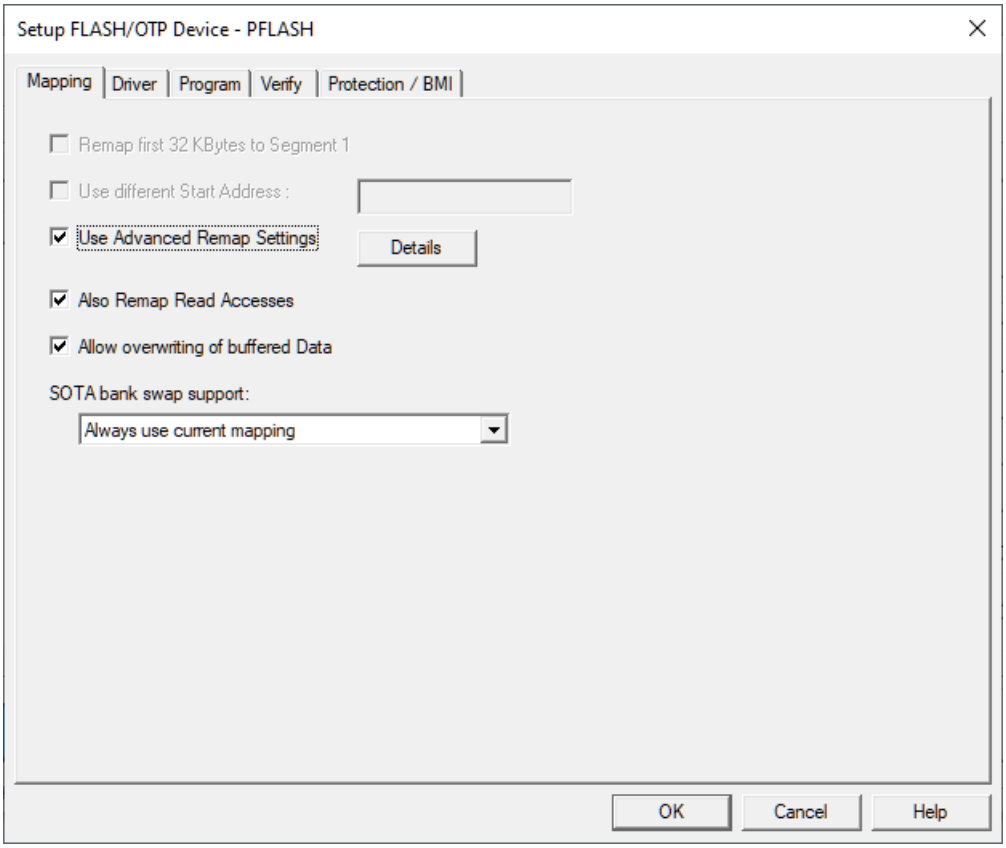


For diagnosis of problems while establishing the target system connection and for the FLASH driver initialization a log window is available. Please click on menu **Log** - **Show Log** to open this window.



Configuring the On-Chip FLASH Module

Before the first FLASH programming cycle can be executed a number of additional configuration settings have to be investigated. Please click on the **Setup** button to open the **Setup FLASH/OTP Device** dialog, and then change to the **Mapping** dialog page.

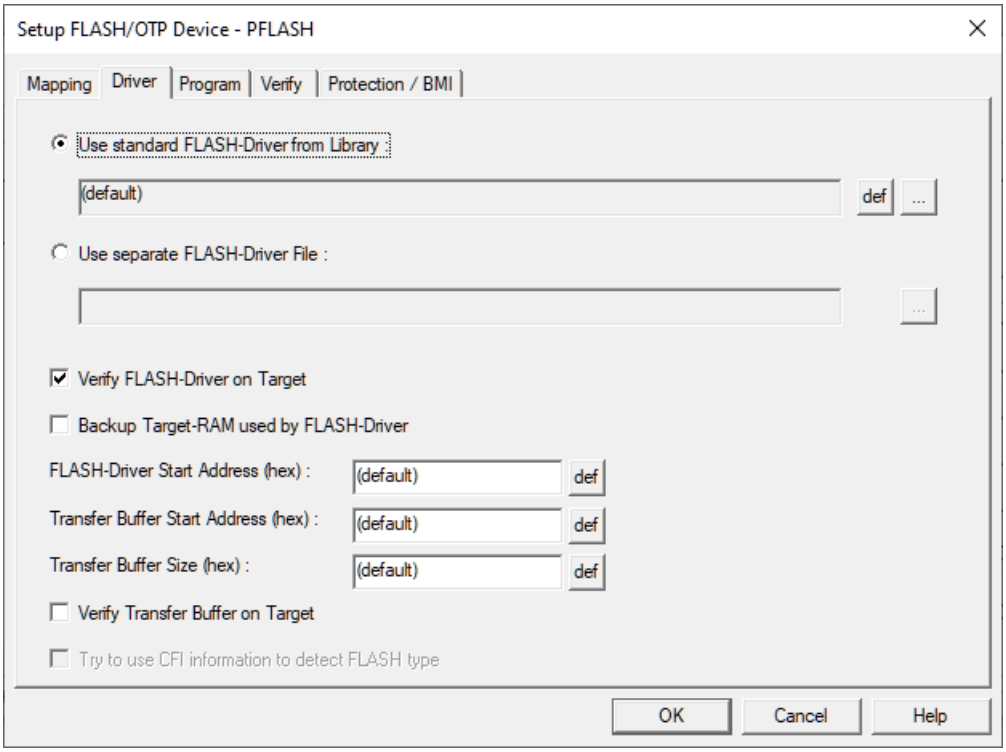


The **Use Advanced Remap Settings** option activates the advanced remap mode. If set the defined remap ranges are used to define the run time mapping of the memory device.

If the **Also Remap Read Accesses** option is set the run time mapping is taken into consideration when reading data from target. Otherwise all data is read from its program time addresses.

In advanced remap mode data section with different run time addresses may have overlapping program time addresses. Or when successive loading several hex files they may have overlapping sections. The **Allow overwriting of buffered Data** option defines what happens when new data would overwrite data already stored in the memory device handler's buffer. If set the new data overwrites the old data. Only a warning is logged when report level is greater than min. Otherwise an error message is generated and the new data is not stored.

The dialog page **Driver** controls the selection of the FLASH Driver for the Memory Device and all other options regarding the driver handling.



The **Use standard FLASH Driver from Library** option is the opposite of option Use separate FLASH Driver File. When set a selected FLASH driver from FLASH driver library FlashDrv.stg is used for the memory device.

The **Use separate FLASH Driver File** option is the opposite of option Use standard FLASH Driver from Library. When set a selected separate file is used as FLASH driver for the memory device.

After the FLASH driver has been loaded into target memory with the **Verify FLASH Driver on Target** option set its correctness within the target memory is verified. To carry out this verification either a CRC checksum calculation or a read back and compare is used.

Before loading the FLASH driver into target memory with the **Backup Target RAM used by FLASH Driver** option set each memory range occupied by the FLASH driver and its transfer buffer is read from target and stored on the host PC. When the FLASH operation is finished, the original memory contents is restored. Please note: Register contents is not saved and restored.

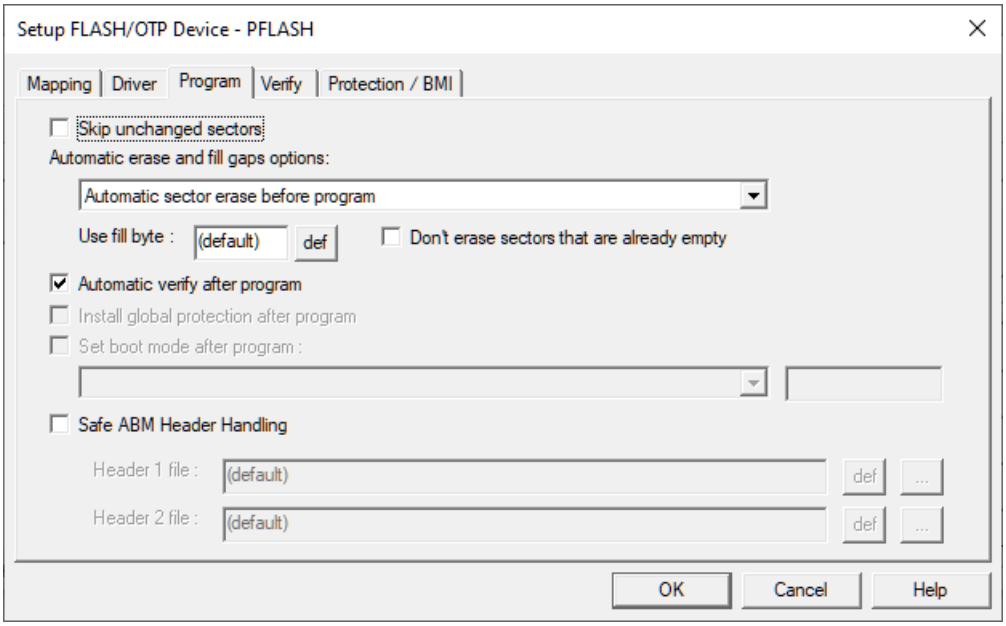
The **FLASH Driver Start Address** option defines the load and start address of the FLASH driver within target memory. By default, this address is fixed and defined by the mapping of the FLASH driver. If the FLASH driver is relocatable within target memory this option can be set by the user. Otherwise setting this option is disabled.
Currently FLASH drivers running on TriCore based targets are relocatable.

The **Transfer Buffer Start Address** option defines the start address of the transfer buffer within target memory which is used to exchange programming data between host PC and FLASH driver. By default, this address is defined by the FLASH driver.

The **Transfer Buffer Size** option defines the size in bytes of the transfer buffer within target memory which is used to exchange programming data between host PC and FLASH driver. By default, this address is defined by the FLASH driver.

Please note: Data to be programmed into the FLASH is usually an executable binary starting at address `0x0`. Please make sure that the **Remap first 32 kBytes to Segment 1** option is unchecked because otherwise data in the address range `0x0-0x7FFF` will not be programmed into the FLASH.

The dialog tab **Program** controls the behaviour of the Program Function.



When executing the program function with the option **Automatic Chip Erase before Program** set the chip erase function is executed before programming any data. When no chip erase function is provided by the memory device the sector erase function is executed for all sectors.

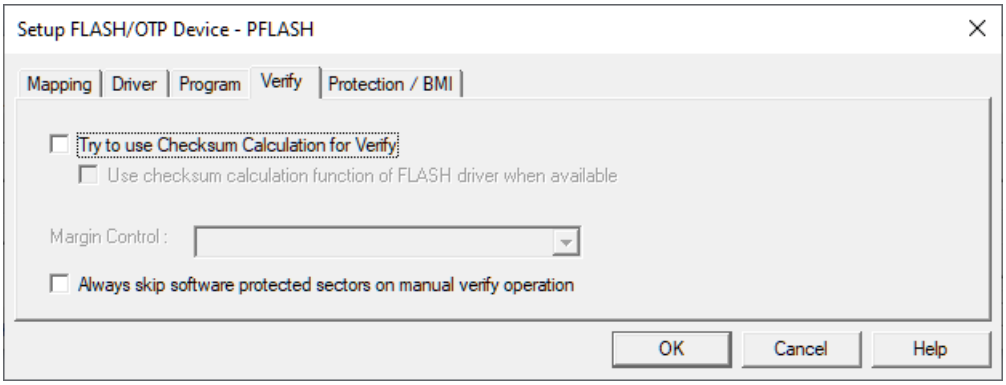
When executing the program function with the **Automatic Sector Erase before Program** option set the sector erase function is executed before programming data into the sector. If no data is buffered for a sector it will be not erased.

When executing the program function with the **Simulate Random Access Mode** option set for each sector which has new data the following sequence of operation is done:

1. Each range of the sector which will not be programmed with new data is read from the memory device and buffered.
2. The sector erase function is executed.
3. All buffered data (new and read) is programmed into the sector.

When executing the program function with the **Automatic Verify after Program** option set the verify function is executed after programming has finished for all sectors

Change to the **Verify** dialog page.



If the **Try to use Checksum Calculation for Verify** option is set Memtool tries to use a checksum calculation function to execute the Verify Function. This checksum calculation may be provided by the FLASH Driver or by the Target Interface that is used for communication. If this option is not set or if no checksum calculation is provided Memtool does a read back and compare to execute the Verify Function.

Some FLASH drivers feature a function for calculating a check sum of target memory areas. This function may be used for verifying the programming cycle. If such a function is

not available all data is read back by the target. Depending on the communication type and speed used verifying this data may take a while.

Remark: Not all FLASH drivers contain a Verify function. Drivers which are entirely executed in the microcontroller's IRAM do not have this function implemented. Therefore, data will be read back upon Verify.

Preparing a binary File

The following output formats of binary and debug symbol data are supported by UDE®:

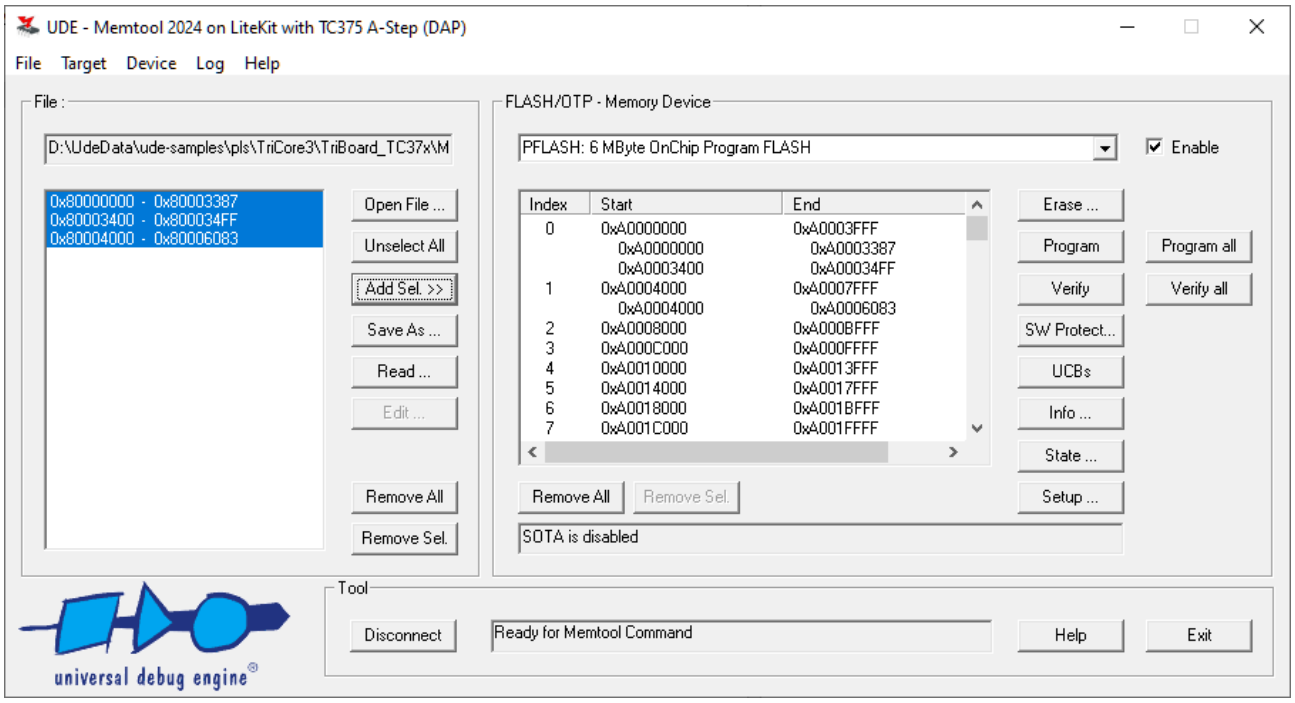
Output format	Expected content
`*.elf`	ELF/DWARF binary object file with debug information
`*.out`	Binary objects file with debug information
`*.axf`	ELF/DWARF binary objects file with debug information
`*.abs`	Binary objects file
`*.hex *.h66 *.h86`	Intel HEX file, ASCII text
`*.bin`	Intel binary objects file
`*.sre`	Motorola S records file, ASCII text
`*.s19`	Motorola S records file, ASCII text

Programming the Application into the FLASH Module

For demonstrating FLASH programming the sample program `MulticoreDemo.hex` will be programmed into the on-chip FLASH module. This program executes a timer loop and creates periodic alternation of the controller's pins, which are connected with LEDs on the TriBoard.

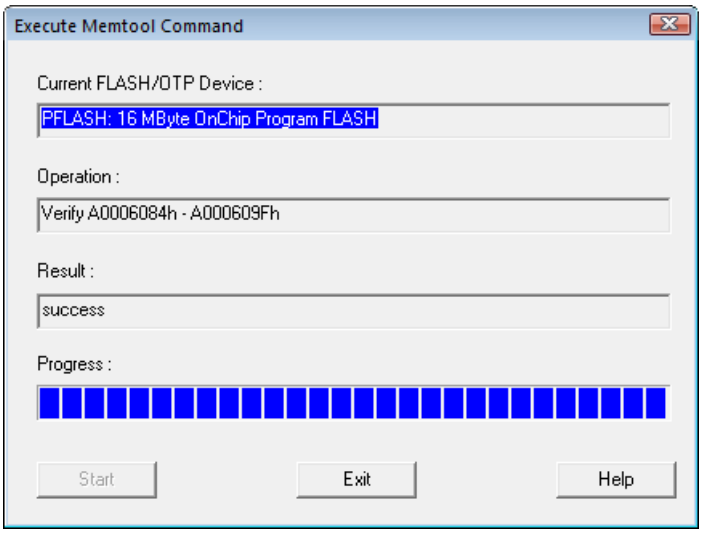
Please click on the **Open** button and select the file `MulticoreDemo.hex`. After loading this file, in the left part of the Memtool window the file name and a list of sections of the application are displayed.

Now please click on **Select All** and then on **Add Selected**.



The sections of the application are now displayed (according to the sectors they belong to) in the list box on the right-hand side.

Please click on the **Program** button to start the programming cycle. The **Execute Memtool Command** dialog will be opened to show the programming progress.



As defined in chapter 'Configuring the On-Chip FLASH Module' of this tutorial a number of sectors will be erased first. Following, the new data will be programmed into the FLASH module and checked after programming has finished.

You may now switch the evaluation board into Single-Chip Mode to watch the blinking LEDs.

Programming an external FLASH Module

Introduction

This tutorial explains the UDE® standalone Memtool configuration to program external FLASH modules on a C166/ST10 target system.

As an example, an evaluation board containing a C167CR microcontroller and one 16-bit FLASH devices of the AM29F200 type is used. Communication between host PC and the target system is established via the standard RS232 serial interface.

Starting point

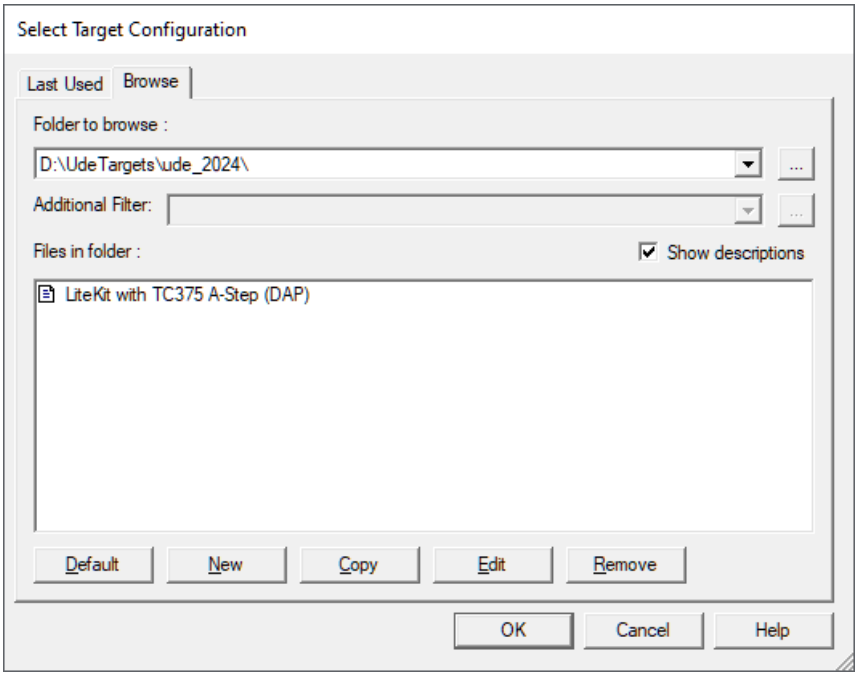
- UDE® standalone Memtool installed or
- UDE® Debugger including UDE® Memtool Add-In installed.

Starting UDE® Memtool

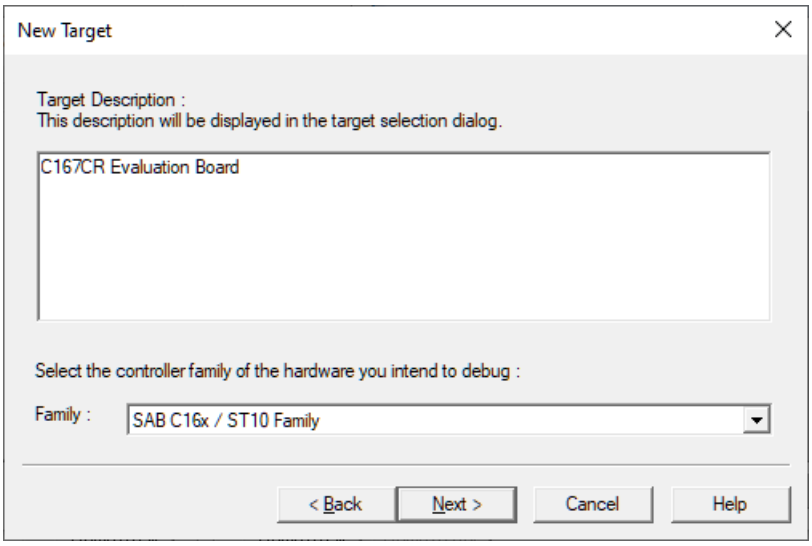
To launch UDE® Memtool in the stand-alone version simply execute Memtool via the Windows main menu **Start** - **Programs** - **UDE Memtool** or via `<UDE_DIRECTORY>\Memtool.exe`.

Creating a New Configuration File

When UDE® Memtool is started for the first time the **Select Target Configuration** dialog is displayed automatically. Otherwise, this dialog can be accessed via menu **Target** - **Change**.



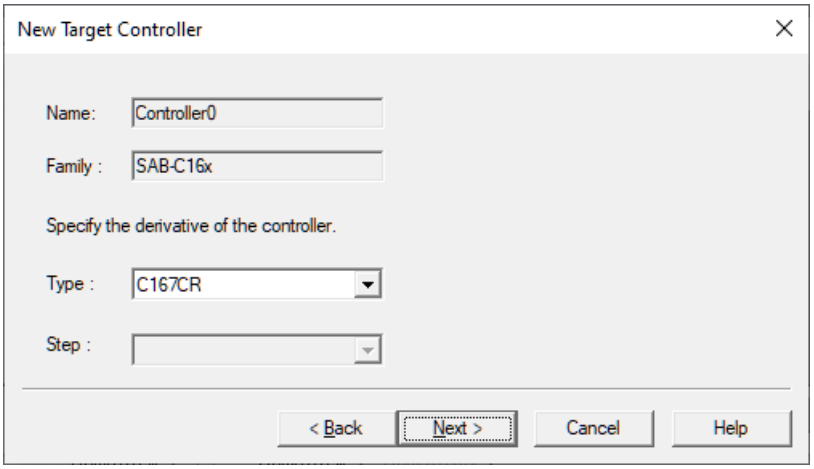
Depending on the type of UDE® installation a number of configuration files can be displayed already. For a new configuration file, click on **New**, select **Create a new target configuration** and click on **Next**.



In **Target Description** a short description of the target system has to be entered. This description will be displayed in the **Select Target Configuration** dialog. In this example, **C167CR Evaluation Board** is used.

In the **Family** list box, the microcontroller family can be selected to which the target system's microcontroller belongs to - in this example the **SAB-C166 / ST10** family.

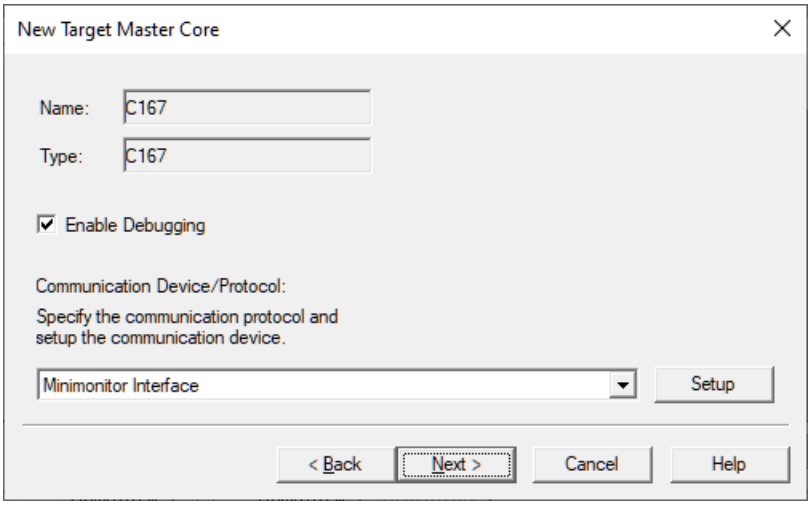
Click on the **Next** button to change to the **New Target Controller** dialog page.



In the **Type** list box please select the exact microcontroller derivative. In this example, the **C167CR** is used.

Remark: Some microcontroller derivatives show different behaviour between different manufacturing steps. If applicable, in the **Step** list box the manufacturing step can be selected.

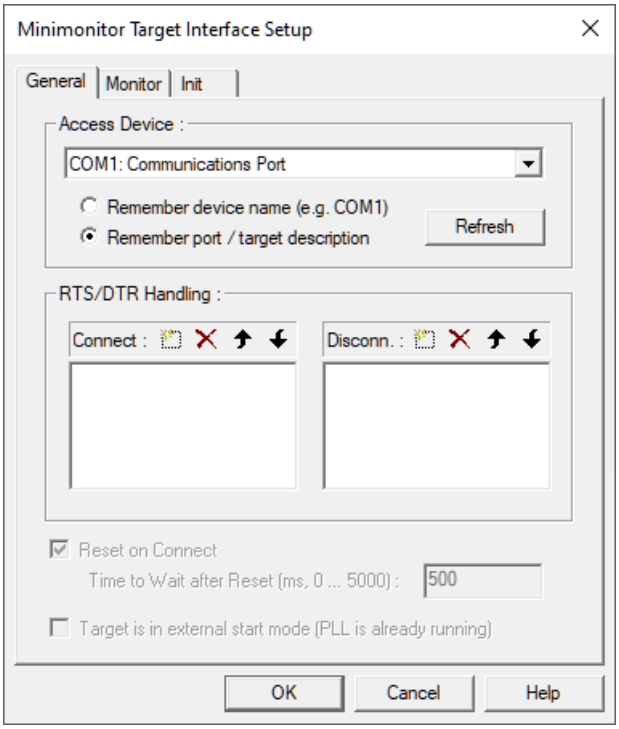
Click on the **Next** button to change to the **New Target Master Core** dialog page.



In the **Communication Device/Protocol** the type of target access is selected. In this example, the **Minimonitor Interface** is used because communication is established via the standard RS232 serial interface.

Communication between the target and the host PC requires some more configuration settings. Please click on the **Setup** button to open the **Minimonitor Target Interface Setup**.

Change to the **Port** dialog page.



Please select the used communication port/device in **Access Device**. In this example, the Host PC Serial Port COM1 is used.

When using the Host PC Serial Port, the RTS and DTR signal of the port can be programmed on user's request. The following commands are available on **RTS/DTR Handling**:

Command	Description
SET [RTS | DTR]	activate RTS/DTR line (switch to +12V)
CLR [RTS | DTR]	deactivate RTS/DTR line (switch to -12V)
WAIT <time>	wait for <time> ms, 10 ... 1000 allowed

The example below demonstrates a use case:

RTS is used to switch the BSL mode. DTR is used to switch the RESET line. (Additional hardware required !)

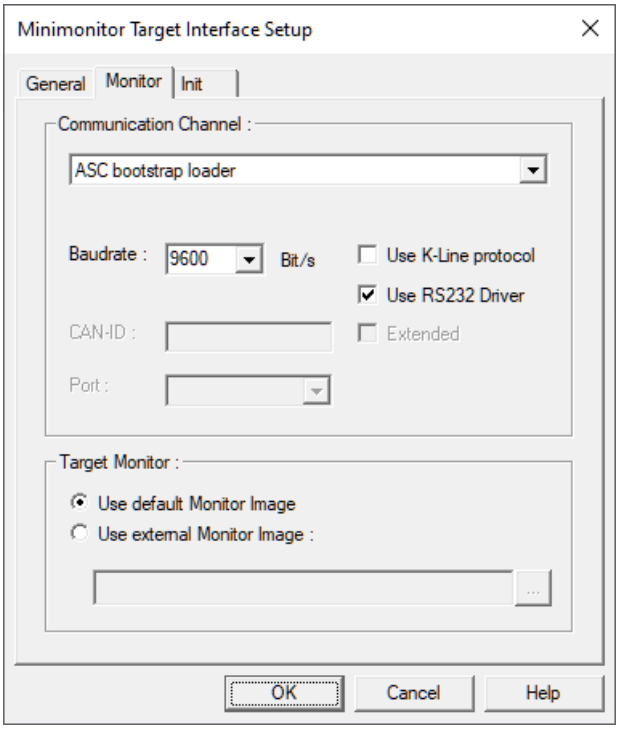
Connect : switch BSL mode on, give a 10 ms RESET pulse and wait 100 ms before connecting

```
SET  RTS
SET  DTR
WAIT 10
CLR  DTR
WAIT 100
```

Disconnect : switch BSL mode off and give a 10 ms RESET pulse

```
CLR  RTS
SET  DTR
WAIT 10
CLR  DTR
```

Please change to the **Monitor** dialog page.

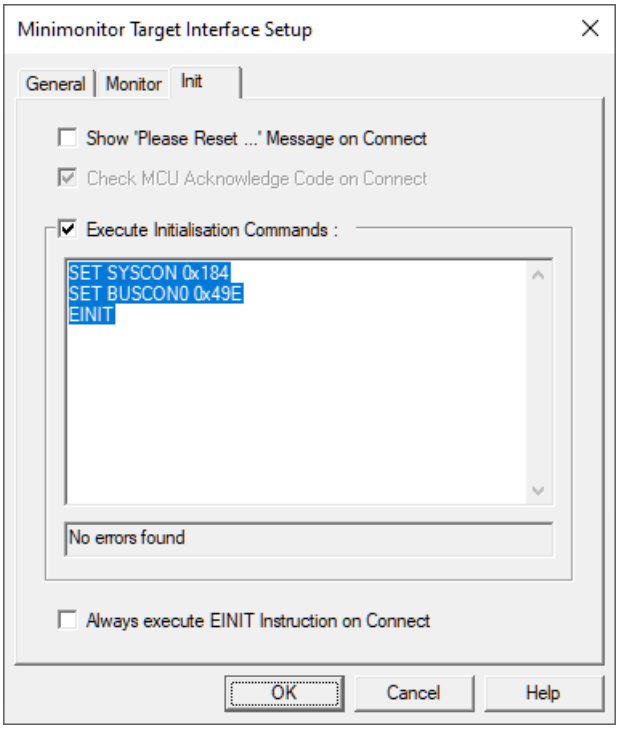


Select the **ASC bootstrap loader** and the baudrate for your target system.

Remark: Target systems featuring a clock speed of 20 MHz and communicating via the standard RS232 serial interface allow a communication speed up to 57600 baud. If a Universal Access Device is used as the communication interface, a communication speed of up to 200.000 baud can be achieved.

The real communication speed depends on type and clock speed of the target system, the communication interface and the provided link cable between the target system and the host hardware. Activate the **Use default Monitor image** to automatically select the monitor software. It is only necessary in special cases to use an external monitor.

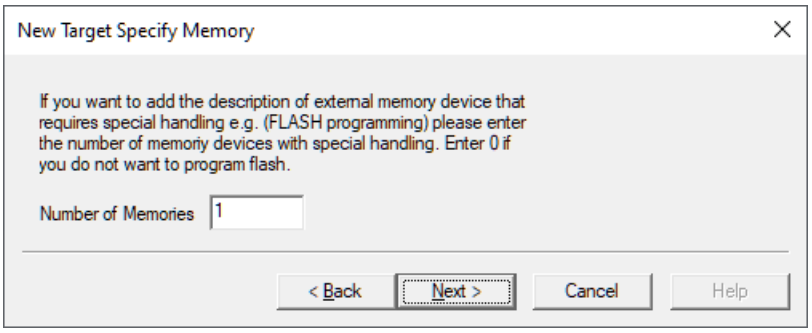
Change to the **Init** dialog page and set up some microcontroller registers upon communication startup. Enable **Execute Initialization Commands**.



In the target system of this example, the external FLASH module to be programmed consists of one 16-bit device of the AM29F200 type connected to CS0 of the microcontroller. Therefore, the BUSCON register have to be set to 0x049E. Following, the EINIT instruction is executed.

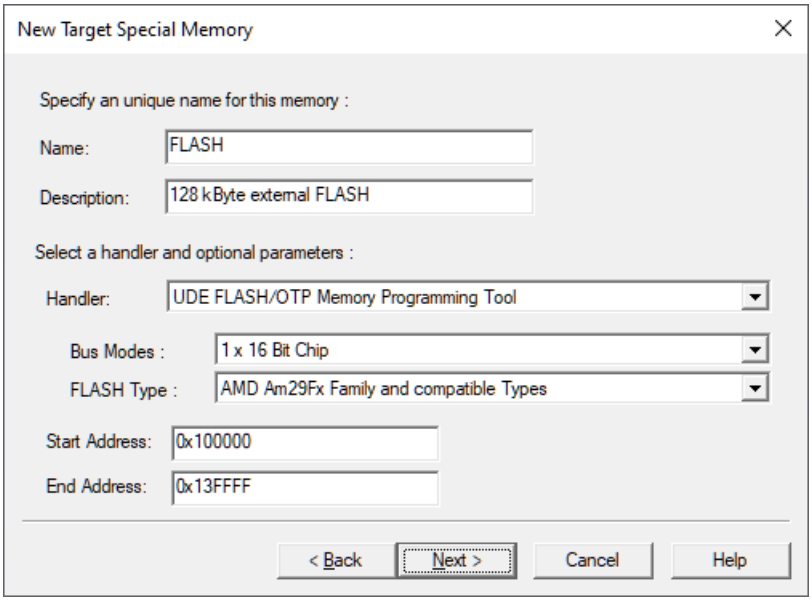
Please leave the **Minimonitor Target Interface Setup** dialog with **OK** to return to the **New Target Master Core** dialog page of the **New Target Wizards**.

Click on **Next** to change to the **New Target Specify Memory** dialog page.



In this example, an external FLASH module has to be defined. Therefore, please fill in '1' in the edit field.

Please click on the **Next** button to change to the **New Target Special Memory** dialog page.



At first, for internal purposes a unique name has to be defined.

Next, please enter a short description of the FLASH module. This description will be used by Memtool later for identifying the FLASH module.

For **Handler** please select **UDE FLASH/OTP Memory Programming Tool** from the list box.

For **Bus Mode** please select **1 x 16 Bit Chips** from the list box.

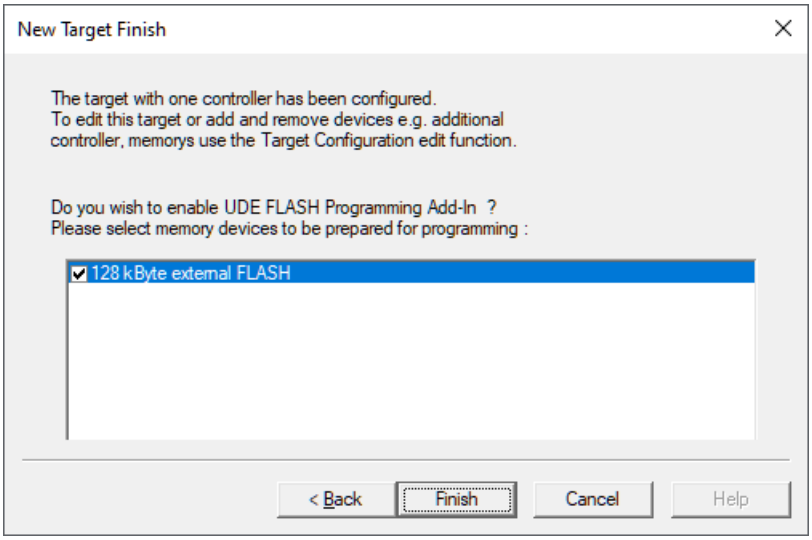
For **FLASH Type** please select **AMD Am29Fx Family and compatible** from the list box.

Then enter `0x10'0000` for **Start Address** and `0x13'FFFF` for **End Address**.

Remark: For start address, please type in the start address of the FLASH module in programming mode. To program the FLASH, it must be completely visible in the address range. If **Start Address** is set to `0x0` parts of the FLASH module are overlapped by the microcontroller's internal memory and can therefore not be programmed. Because in this example the FLASH module is connected to `CS0` and no other `BUSCONx` registers are activated, the FLASH will be visible at all addresses which are multiples of the FLASH module size.

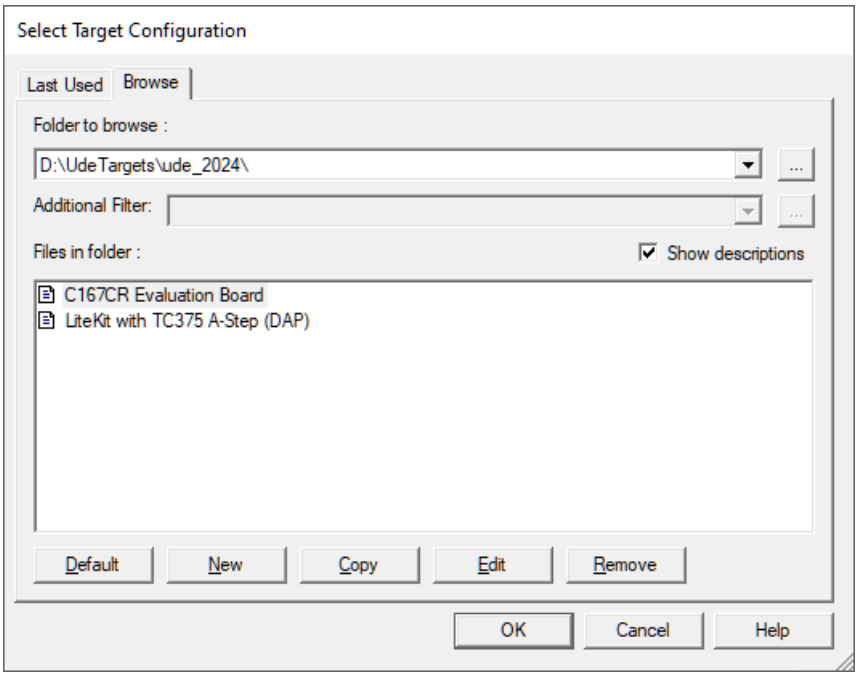
This setting does not interfere with demands to program data to address 0x0. For this, all settings required are described in chapter 4 of this tutorial.

Click on the **Next** button to change to the **New Target Finish** dialog page.



Now the New Target Wizard is completed.

Click on the **Finish** button to close the wizard. The **Select Target Configuration** is displayed again containing the newly created configuration file which is highlighted.

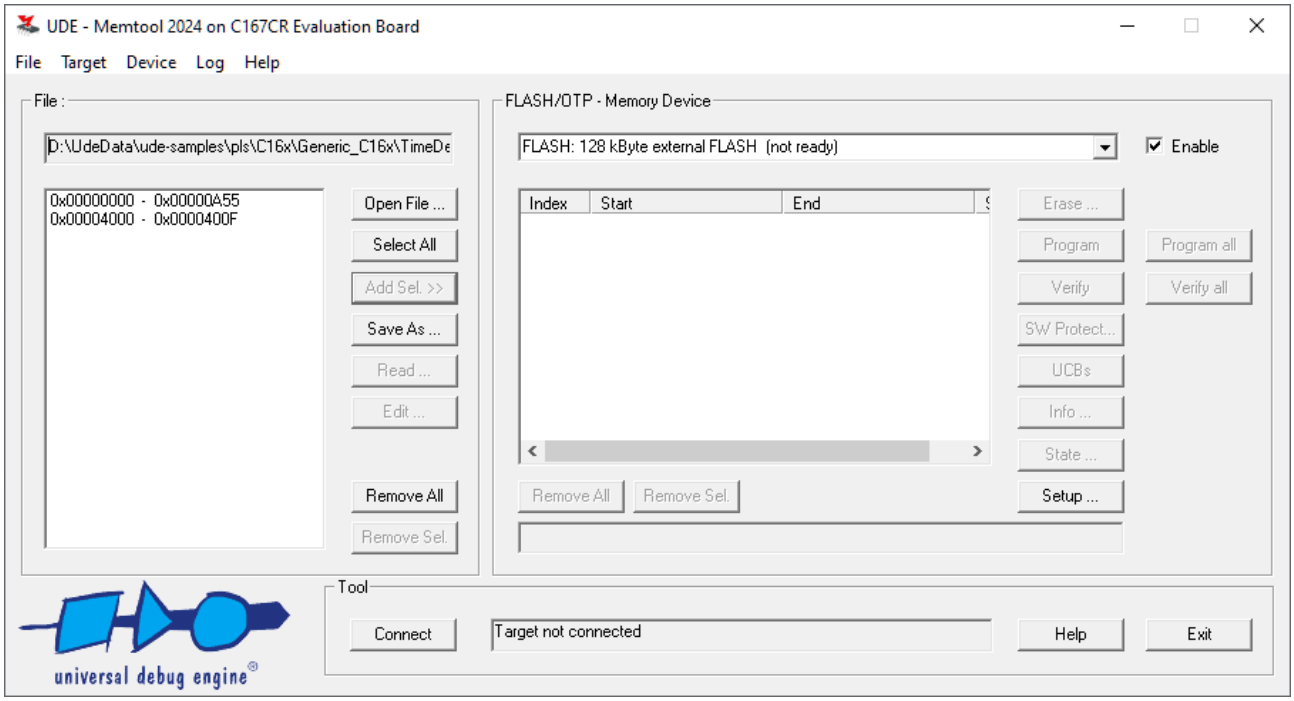


Please leave this dialog with **OK** to load the new configuration file.

Connecting to the Target

When a new version of UDE® Memtool is started the first time a firmware update may be executed on the access device (UAD2pro, UAD3+, ...). This may take some more time than usual for the target connect operation. Please **DO NOT** power off or unplug the access device while this time!

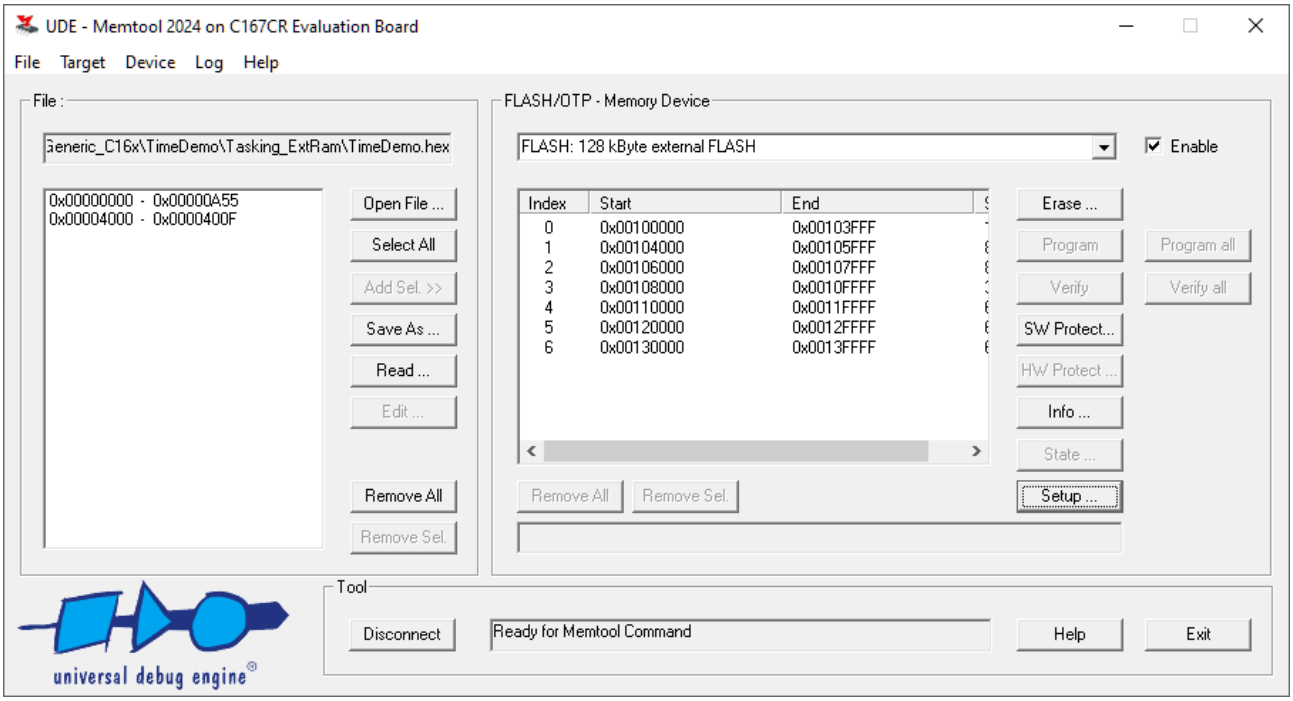
After selecting a configuration file UDE® Memtool is ready to establish a connection to the target system.



In the title bar a description of the currently loaded configuration file is displayed. In the combo box under **FLASH/OTP-Memory Devices** all available FLASH modules are listed.

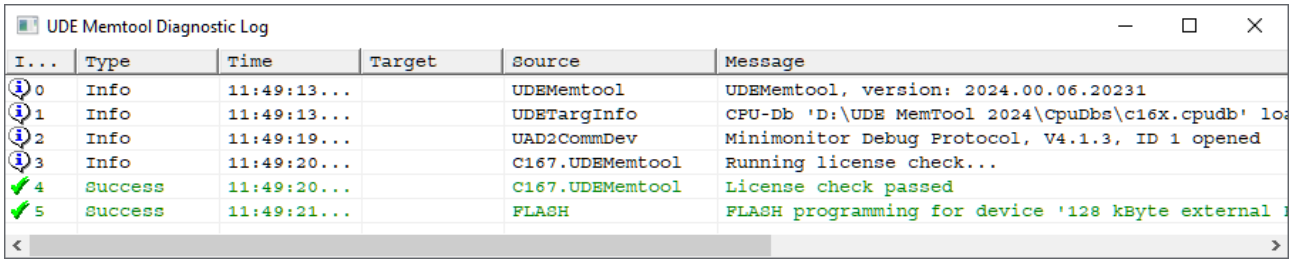
In this example, only the **128 kByte external FLASH** is available. For external FLASH modules the sector table is crated on connect after determining the actual type of the FLASH. So, the **Sector** list box is empty at this time.

Please click on the **Connect** button to establish connection to the target system.



Now the **Sector** list box contains the sector table of the selected FLASH module.

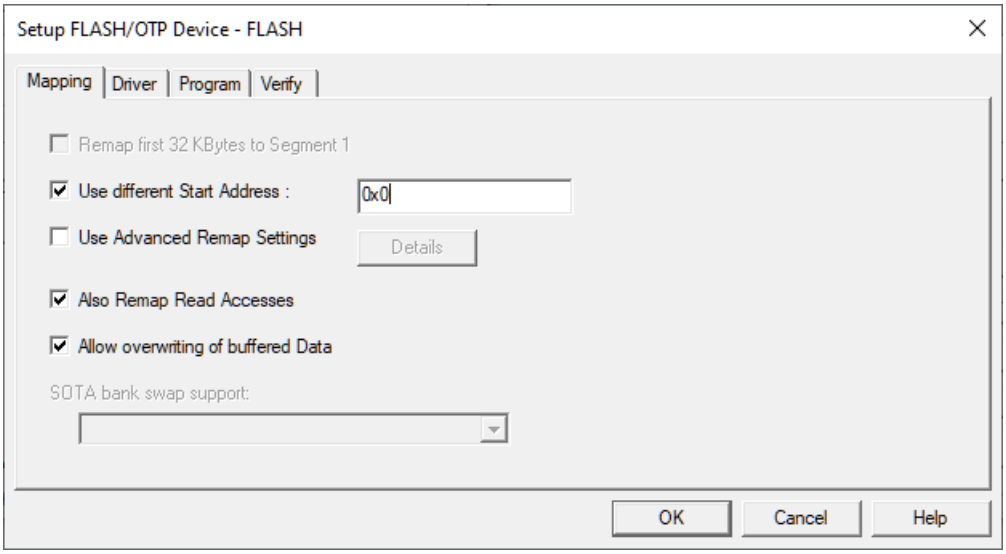
For diagnosis of problems while establishing the target system connection and for the FLASH driver initialization a log window is available. Please click on menu **Help** - **Show Log** to open this window.



Configuring the external FLASH Module

Before the first FLASH programming cycle is executed a number of additional configuration settings have to be investigated.

Please click on the **Setup** button to open the **Setup FLASH/OTP Device** dialog, and then change to the **Mapping** dialog page and set the **Use different start Address** to `0x0`. This is required, because the program code starts at `0x0`, but the appropriate FLASH device start at a higher address. That's why a mirroring must be done.



A detailed description about the setup of a FLASH module can be found in the chapter **Configuring the On-Chip FLASH Module**.

Preparing a binary File

The UDE® Memtool supports the compiler tool chain of various compiler manufacturers. Please see the compatibility list for a selection of suitable compilers and the compiler manual for correct usage of the compiler.

The following output formats of binary and debug symbol data are supported by UDE® Memtool:

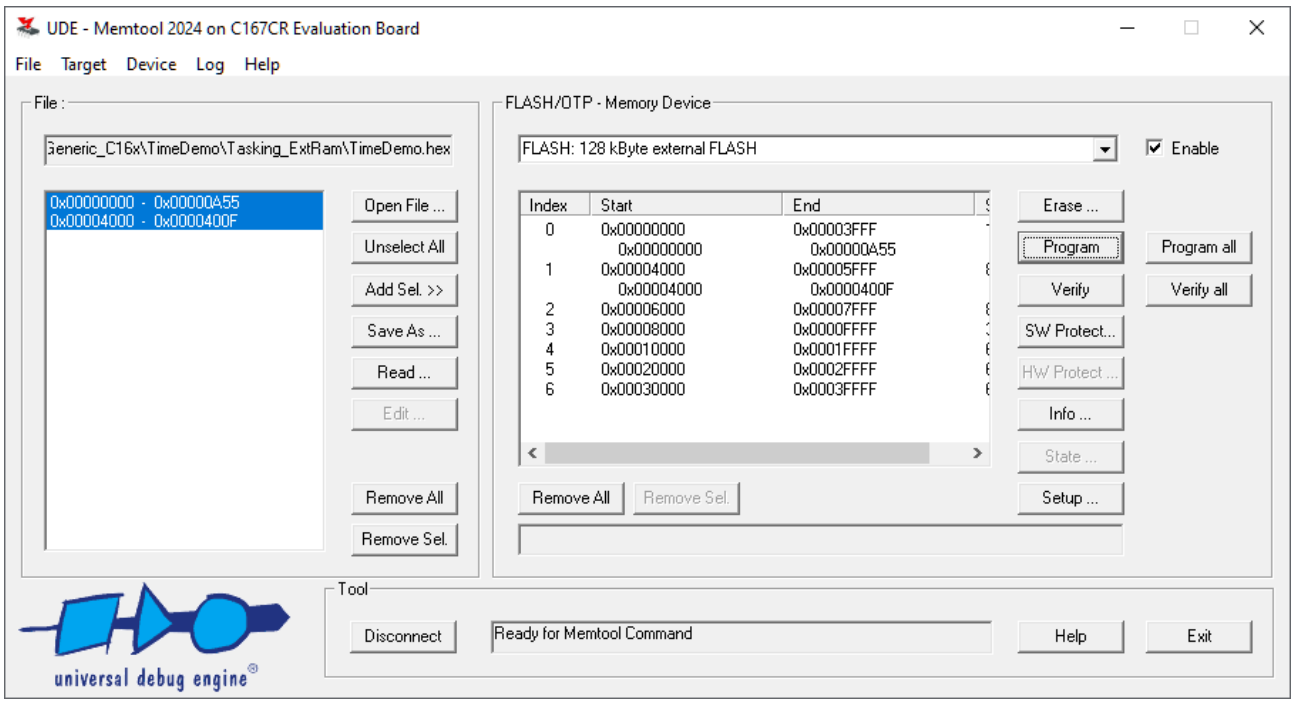
Output format	Expected content
`*.axf *.elf`	ELF/DWARF binary object file with debug information
`*.out`	Binary objects file with debug information
`*.abs`	Binary objects file
`*.hex *.h66 *.h86`	Intel HEX file, ASCII text
`*.bin`	Intel binary objects file
`*.sre *.s19`	Motorola S records file, ASCII text

Programming the Application into the FLASH Module

For demonstrating the FLASH programming the sample program `TimeDemo.hex` will be programmed into the external FLASH module. This program executes a timer loop and the SWRST instruction to create periodic alternation of the controller's RESETOUT pin. At this pin, a LED is connected on many evaluation boards which can be lit this way.

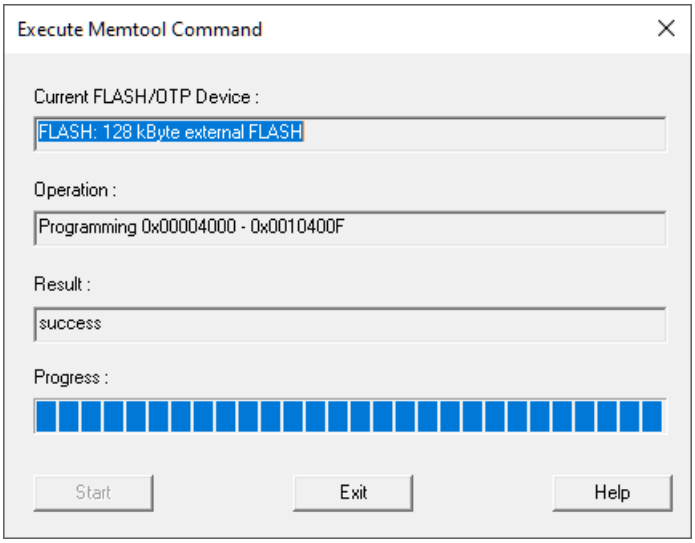
Please click on the **Open File** button and select the file `TimeDemo.hex`. After loading this file, in the left part of the Memtool window the file name and a list of sections of the application is displayed.

Now please click on **Select All** and then on **Add Selected**.



The sections of the application are now displayed (according to the sectors they belong to) in the list box on the right-hand side.

Please click on the **Program** button to start the programming cycle. The **Execute Memtool Command** dialog will be opened to show the programming progress.



As defined in chapter **Configuring the external FLASH Module** of this tutorial a number of sectors will be erased first. Following, the new data will be programmed into the FLASH module and checked after programming has finished.

You may now switch the evaluation board into none bootstrap loader mode to watch the blinking LED at the RESETOUT pin.

Automation of UDE® Memtool

Introduction

For automation purposes of FLASH programming services, the UDE® Memtool can be used as server. The UDE® Memtool server component `Memtool.dll` works as COM-in-process server and provides an object model for accessing the function of FLASH programming.

When a new version of UDE® Memtool is started the first time a firmware update may be executed on the access device (UAD2pro, UAD3+, ...). This may take some more time than usual for the target connect operation. Please **DO NOT** power off or unplug the access device while this time!

Methods of Automation

The following chapter contains examples of using the UDE® Memtool object model. The code demonstrates the basic access to objects on various ways.

The **Batch Mode** example shows a simple automation via a batch, executed by UDE® Memtool.

A further simple implementation of a command line interpreter is delivered with UDE® Memtool, called **MemtoolCli**. It provides an access interface to the object model and the most relevant FLASH programming functions.

Full access to the object model is possible via using a script or programming language. Example scripts for **Visual Basic Script, JScript, Perl** and **Python** are available at the end of the chapter.

UDE® Object Model

The object model of UDE® Memtool server is described in detail in the UDE® Memtool Help. Please refer to

`<UDE_MEMTOOL_DIRECTORY>\Help\Memtool.chm`

in the UDE Help system (or press **<F1>**) and look for the topic 'Object Model'.

Batch Mode of UDE® Memtool

The client application `UDEMemtool.exe` is equipped with a graphical user interface for Microsoft™ Windows. It provides a batch mode where multiple commands can be assembled to a batch file. To execute such batch file, select in the menu **File** - **Run Batch** command and choose the batch file.

Alternatively, you can run the batch file using a command line parameter while starting the Memtool from the command line.

Synopsis:

```
UDEMemtool[.exe] [<batchfile>][-c <cfgfile>]
```

Please note: When using option '-c' to specify the configuration file to be used always pass in the absolute path of the configuration file!

Batch commands:

`#`	Start of comment line
`connect`	Establish connection to target system. If the target is already connected nothing is done.
`disconnect`	Shutdown the connection to the target system
`set IgnoreErrorsInHexFiles | IgnoreMappingErrors`	Activate error handling option - IgnoreErrorsInHexFiles ... Ignore errors in hex files (bad checksum, no end record) - IgnoreMappingErrors ... Ignore mapping errors while loading sections Note: Command has to be executed before open_file
`open_file <hexfile> [,[+|-]<offset>]`	Open a hex file and load it into the File Section List. <hexfile> is the absolute path name of the file to be loaded <offset> is an optional numeric value that is added to all section addresses in the file
`select_all_sections`	Selects all sections in the left listbox.
`select_sections <index> [, <index> [, ...]]`	Select a single section or a list of sections given its zero-based indices.
`deselect_sections <index> [, <index> [, ...]]`	Deselect a single section or a list of sections given its zero-based indices.
`add_selected_sections`	Take all selected sections in the left list box and apply them to the right one.
`load_bin_file <path>, <address>`	Load binary file into FLASH section list starting at <address>. <path> is the absolute path name of the file to be loaded Please note: It is not required to execute 'select_all_sections' and 'add_selected_sections'.
`program`	Starts the programming of the flash device.
`verify`	Execute the 'Verify all devices' function
`exit`	Stop batch file execution and exit UDEMemtool.exe

Example for FLASH programming via Batch mode

```
# FLASH programming via Batch

# connect target system
connect

# open starter software file
open_file c:\Data\starter.hex

# load sections into FLASH section list
select_all_sections
add_selected_sections

# open firmware file, original location
open_file c:\Data\firmware.hex

# load sections into FLASH section list
select_all_sections
add_selected_sections

# open firmware file, copy location
open_file c:\Data\firmware.hex , +40000h

# load sections into FLASH section list
select_all_sections
add_selected_sections

# open serial number file
open_file c:\Data\sernum.hex

# load sections into FLASH section list
select_all_sections
add_selected_sections

# program all loaded sections to FLASH
program

# disconnect target
disconnect

# exit UDEMemtool
exit
```

This example batch file connects the target system. Than it loads a starter software (`starter.hex`), a firmware and a device specific serial number.

The firmware is loaded twice. The second time it is loaded using an offset of `0x40000`. So, the firmware image is programmed twice into the FLASH memory device.

Please note that after each open_file operation the loaded sections have to be added to the FLASH Buffers before the next open_file operation can take place. Then all loaded sections are programmed into the FLASH memory device.

Finally, `UDEMemtool.exe` is closed when no error has occurred.

Command Line Interface

The MemtoolCli V1.12.0 application realizes a command line interface of the UDE® Memtool server, so that a FLASH programming is possible via a batch file.

Synopsis:

```
MemtoolCli[.exe] <command>
                 [<hexfile> [<hexfile> [..]]]
                 [<section> [<section> [..]]] [-s]
                 [-c <configfile>] [-l <logfile>]
                 [-v <verifyfile >] [-f <flash>]
                 [-g] [-p <password>]
                 [-m <new mode>]
                 [-o] [-R] [-U]
                 [-d <output path>] [-i]
                 [<range> [,<range>,...]]
```

Options:

`<command>`	`ERASE`	Execute Erase Function
	`PROGRAM`	Execute Program Function
	`VERIFY`	Execute Verify Function
	`PROGRAM+VERIFY`	Execute Program and Verify Function (independently of 'verify after program' configuration)
	`PROTECT`	Execute protection function (option -f required)
	`REMOVEPROTECTION`	Remove current protection settings (option -f required)
	`GETBOOTMODE`	Read current boot mode
	`SETBOOTMODE`	Set new boot mode
	`READ <range> [,<range>,...]`	Read ranges from target memory
	The command is executed for all memory devices that are enabled in the configuration file.	
`<hexfile>`	Path of a hex file containing data for program or verify.	
	For commands PROGRAM and VERIFY at least one file has to be specified. More than one files can be specified. When there are overlapping sections in different files data is taken form the file that was specified last.	
`<section>`	Zero based index of a sector to be erased when executing the ERASE command. When omitted all sectors are erased.	
`-c <configfile>`	Path of target configuration file.	
	This argument is optional. When omitted the configuration file is used that was lastly used with UDEMemtool.exe. (UDEMemtool.ini: [Main].TargInfoFile)	
`-s`	Silent mode. Suppress all output regardless of the current log level.	
`-l <logfile>`	Set the log level for diagnostic output. Range: 0 ... 100. This argument is optional. When omitted the value is taken, that was lastly used with UDEMemtool.exe. (UDEMemtool.ini: [Main].LogLevel)	
`-v <verifyfile>`	Create a file containing the differences found when executing the verify function. When the file already exists it will be overwritten. This argument is optional. When omitted	

	the value is taken that was lastly used with UDEMemtool.exe. (UDEMemtool.ini: [Main].VerifyProtFile)
`-f <flash>`	Execute the command for the memory device <flash> only. When omitted the command is executed for all memory devices that are enabled currently. With command PROTECT or REMOVEPROTECTION this is not an option but required.
`-g`	Install global read/write protection (PROTECT and REMOVEPROTECTION command only)
`-p <password>`	Password used by protection function. When omitted the password is taken from configuration file. When no password is specified in configuration file the password is set to 0.
`-m <new mode>`	New boot mode used by SETBOOTMODE command.
`-o`	Look for other communication device when checking license.
`-R`	Check for node locked or floating license.
`-U`	Check USB license key.
`<range>` `[,<range>,...]`	Define list of address ranges for READ command. e.g. 0xA0000000-0xA0007FFF, 0xA0010000-0xA0017FFF
`-d <output file path>`	Option for READ command. Save read sections to file.
`-i`	Option for READ command. Ignore user defined mapping.

Script programming with Visual Basic Script

```
'-------------------------------------------------------------------------------
' UDEMemtool automation basics - VBScript demo script
'
' Command line run: cscript MemtoolAutomationDemo.vbs
'-------------------------------------------------------------------------------

WScript.echo "UDEMemtool VBScript Sample"

ProgId = "UDE.Memtool"
'ProgId = "UDE.Memtool.2025"  ' use a special version of UDE

WorkDir = cwd()
'WScript.Echo "WorkDir : " & WorkDir

CfgFile = WorkDir & "\TriBoard_TC27x_dap.cfg" ' target configuration file
HexFile = WorkDir & "\MulticoreDemo.hex"      ' target application file
LogFile = WorkDir & "\Memtool.log"            ' Memtool log file
'WScript.Echo "CfgFile: " & CfgFile
'WScript.Echo "HexFile: " & HexFile
'WScript.Echo "LogFile: " & LogFile

WScript.Echo "Create new UDEMemtool object ..."
dim Memtool
set Memtool = CreateObject(ProgId)
WScript.echo "  Version: " & Memtool.Version

' prepare logging
WScript.echo "Prepare logging..."
Memtool.CreateLogFile LogFile, 50

' initialize Memtool object using config file
WScript.echo "Init Memtool..."
Memtool.Init(CfgFile)

' connect target system
WScript.echo "Connect target..."
Memtool.ConnectTarget()

' load target application file
WScript.echo "Load target application file..."
Memtool.OpenFile HexFile, true

' program FLASH
WScript.echo "Program FLASH..."
Memtool.ProgramSections()

' verify FLASH
WScript.echo "Verify FLASH..."
Diffs = Memtool.VerifySections("")
WScript.echo "  " & Diffs & " differences found"

' disconnect target and cleanup Memtool object
WScript.echo "Disconnect and cleanup..."
Memtool.DisconnectTarget()
Memtool.Cleanup()

WScript.echo "finished"

' Get current directory
function cwd
     dim fso
     set fso = CreateObject("Scripting.FileSystemObject")
     cwd=fso.GetAbsolutePathName(".")
end function
```

Script programming with JScript

```
// ---------------------------------------------------------------------------
// UDEMemtool automation basics - JScript demo script
//
//  Command line run: cscript MemtoolAutomationDemo.js
// ---------------------------------------------------------------------------

WScript.echo("UDEMemtool JScript Sample");

var ProgId = "UDE.Memtool";
//var ProgId = "UDE.Memtool.2025";    // use a special version of UDE

var WorkDir = cwd();
//WScript.echo("WorkDir : " + WorkDir );

var CfgFile = WorkDir + "\\TriBoard_TC27x_dap.cfg";    // target configuration file
var HexFile = WorkDir + "\\MulticoreDemo.hex";// target application file
var LogFile = WorkDir + "\\Memtool.log";               // Memtool log file
//WScript.echo("CfgFile: " + CfgFile);
//WScript.echo("HexFile: " + HexFile);
//WScript.echo("LogFile: " + LogFile);

WScript.Echo("Create new UDEMemtool object ...");
var Memtool = new ActiveXObject(ProgId);
WScript.echo("  Version: " + Memtool.Version);

// prepare logging
WScript.echo("Prepare logging...");
Memtool.CreateLogFile(LogFile, 50);

// initialize Memtool object using config file
WScript.echo("Init Memtool...");
Memtool.Init(CfgFile);

// connect target system
WScript.echo("Connect target...");
Memtool.ConnectTarget();

// load target application file
WScript.echo("Load target application file...");
Memtool.OpenFile(HexFile, true);

// program FLASH
WScript.echo("Program FLASH...");
Memtool.ProgramSections();

// verify FLASH
WScript.echo("Verify FLASH...");
var Diffs = Memtool.VerifySections("");
WScript.echo("  " + Diffs + " differences found");

// disconnect target and cleanup Memtool object
WScript.echo("Disconnect and cleanup...");
Memtool.DisconnectTarget();
Memtool.Cleanup();

WScript.echo("finished");

// Get current directory
function cwd() {
     var fso = new ActiveXObject("Scripting.FileSystemObject");
     var cwd = fso.GetAbsolutePathName(".");
     return cwd;
}
```

Script programming with Perl

```perl
#!/usr/bin/perl

# ------------------------------------------------------------------------------
# UDEMemtool automation basics - Perl demo script
#
# Command line run: perl MemtoolAutomationDemo.pl
# ------------------------------------------------------------------------------

use strict;
use Win32::OLE;
use Cwd;

print("UDEMemtool Perl Sample\n");

my $ProgId = "UDE.Memtool";
#my $ProgId = "UDE.Memtool.2025";     # use a special version of UDE

#my $WorkDir = "U:\\ude-dev\\Ude\\Automation\\MemtoolAutomationDemo";
my $WorkDir = cwd();
#print("WorkDir: $WorkDir\n");

my $CfgFile = $WorkDir . "\\TriBoard_TC27x_dap.cfg";  # target configuration file
my $HexFile = $WorkDir . "\\MulticoreDemo.hex";       # target application file
my $LogFile = $WorkDir . "\\Memtool.log";             # Memtool log file
#print("CfgFile: $CfgFile\n");
#print("HexFile: $HexFile\n");
#print("LogFile: $LogFile\n");

print("Create new UDEMemtool object ...\n");
my $Memtool = new Win32::OLE($ProgId) or die "Failed to create UDE.Memtool object !\n";
print("  Version: " . $Memtool->Version . "\n");

# prepare logging
print("Prepare logging...\n");
$Memtool->CreateLogFile($LogFile,50);

# initialize Memtool object using config file
print("Init Memtool...\n");
$Memtool->Init($CfgFile);

# connect target system
print("Connect target...\n");
$Memtool->ConnectTarget();

# load target application file
print("Load target application file...\n");
$Memtool->OpenFile($HexFile, 1);

# program FLASH
print("Program FLASH...\n");
$Memtool->ProgramSections();

# verify FLASH
print("Verify FLASH...\n");
my $Diffs = $Memtool->VerifySections("");
print("  " . $Diffs . " differences found\n");

# disconnect target and cleanup Memtool object
print("Disconnect and cleanup...\n");
$Memtool->DisconnectTarget();
$Memtool->Cleanup();

print("finished\n");
```

Script programming with Python

```
# ------------------------------------------------------------------------------
# UDEMemtool automation basics - Python demo script
#
# Command line run: python MemtoolAutomationDemo.py
# ------------------------------------------------------------------------------

import sys
import os
import win32com.client

print("UDEMemtool Python Sample")

ProgId = "UDE.Memtool"
#ProgId = "UDE.Memtool.2025"          # use a special version of UDE

# get current directory
WorkDir = os.path.abspath(".")
#print("WorkDir: " + WorkDir)

CfgFile = WorkDir + "\\TriBoard_TC27x_dap.cfg"# target configuration file
HexFile = WorkDir + "\\MulticoreDemo.hex"             # target application file
LogFile = WorkDir + "\\Memtool.log"                   # Memtool log file
#print("CfgFile: " + CfgFile)
#print("HexFile: " + HexFile)
#print("LogFile: " + LogFile)

print("Create new UDEMemtool object ...")
Memtool = win32com.client.Dispatch(ProgId)
print("  Version: " + Memtool.Version)

# prepare logging
print("Prepare logging...")
Memtool.CreateLogFile(LogFile,50)

# initialize Memtool object using config file
print("Init Memtool...")
Memtool.Init(CfgFile)

# connect target system
print("Connect target...")
Memtool.ConnectTarget()

# load target application file
print("Load target application file...")
Memtool.OpenFile(HexFile, True)

# program FLASH
print("Program FLASH...")
Memtool.ProgramSections()

# verify FLASH
print("Verify FLASH...")
Diffs = Memtool.VerifySections("")
print("  " + str(Diffs) + " differences found")

# disconnect target and cleanup Memtool object
print("Disconnect and cleanup...")
Memtool.DisconnectTarget()
Memtool.Cleanup()

print("finished")
```

Reference

The Reference of the UDE® Universal Debug Engine is available via the Window's Help system of UDE®. To use the help, please open UDE® and press **<F1>** or browse to the UDE® installation menu and select in UDE® menu **Help**.

Trouble Shooting with UDE®

Trouble Shooting Checklist

Please read the **UDE Memtool Manual** and the hints very conscientious. If the problem is not soluble, the fastest way is to download the **UDE Support Checklist Form** from **https://www.pls-mc.com/downloads/UDE_Support_Checklist_Form.pdf**, fill out and e-mail it to the PLS Support Line at **support@pls-mc.com**.

Our Support team will contact you as soon as possible.

Frequently Asked Questions (FAQ's)

For technical questions, please consult the UDE® FAQ list on our website **https://www.pls-mc.com/faqs.html** first.

Precautions when installing a new UDE® version

When starting a newly installed version of UDE® for the first time, a **firmware update** may be executed for the access device (UAD2^{pro}, UAD2^{+}, UAD2^{next}, and UAD3^{+}). This may take some more time than usual for the "target connect" operation. Please **DO NOT** power OFF or unplug the access device during this period!

Downloading the latest UDE® Version

Our **PLS Newsletter** keeps you informed about latest news and software versions of the UDE®. You can subscribe to the PLS Newsletter via your profile at **https://www.pls-mc.com/accounts/profile/**.

You may find the latest version of UDE® Universal Debug Engine and other components on our website **https://www.pls-mc.com/download.htm** for downloading. It is required that you are registered and logged in before.

Index