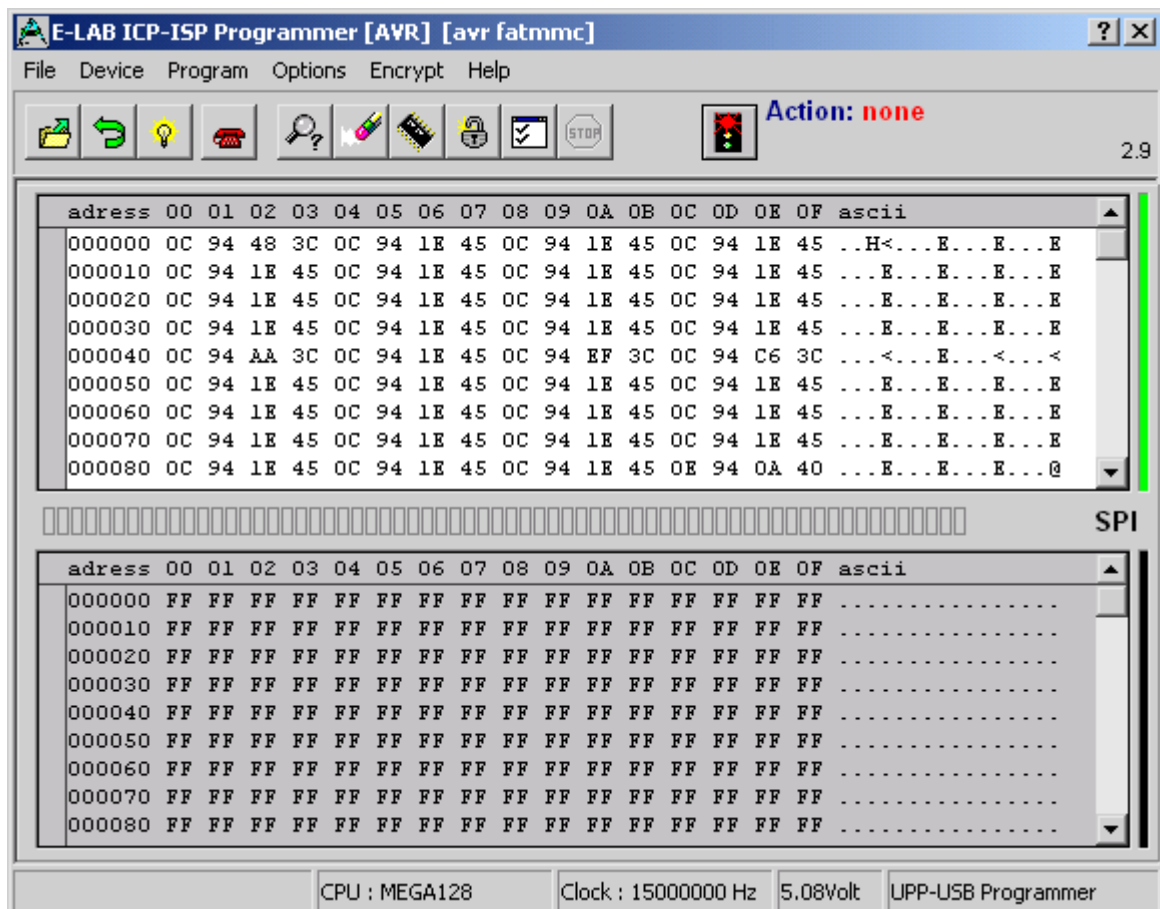




User Manual E-LAB UPP1-X (XS) Programmer

UPP1-X *with AES encryption* *Portable Programmer*

© Copyright 1998-2018 by E-LAB Computers



WEB: www.e-lab.de Tel: 07268/9124-0 Fax: 07268/9124-24



Table of Contents

| | |
|--|-----------|
| OVERVIEW..... | 3 |
| FEATURES..... | 3 |
| CONNECTIONS | 4 |
| SOFTWARE | 5 |
| AVRProg | 5 |
| Call options | 8 |
| Button-Bar | 10 |
| Function of Buttons and Menus..... | 11 |
| Encryption..... | 20 |
| AES PAC files..... | 24 |
| Programming directly..... | 25 |
| Project download | 26 |
| Programming using the SD card | 27 |
| States, Error Display and Problems | 28 |
| Building Project Files for UPP1 programmers..... | 29 |
| PackProg | 30 |
| Buttons and Menus..... | 30 |
| Command line parameters | 33 |
| Return Codes | 33 |
| Telnet Interface..... | 34 |
| Stand alone mode..... | 36 |
| Option XS | 37 |
| DLL REMOTE CONTROL | 38 |
| EXTERNAL HARDWARE | 39 |
| Battery Operation | 39 |
| Miscellaneous Adaptors | 39 |
| TARGET POWER SUPPLY | 40 |
| MULTIPLE PROGRAMMERS | 40 |
| USB DRIVERS | 41 |
| Installing unsigned drivers..... | 42 |
| FIRMWARE UPDATE | 45 |
| ADDENDUM | 47 |
| xMega..... | 47 |
| TINY4-5-9-10-20..... | 48 |
| UPDI Tiny and UPDI Mega | 49 |
| Chipcon | 50 |
| Chipcon Evaluation Boards | 51 |
| AT89LPxx | 53 |
| Serial Flash (SPI-Flash)..... | 54 |
| OTHER E-LAB PROGRAMMERS | 55 |

OVERVIEW

In-Circuit Programming (ISP or JTAG) is the technology of the future, at least for small and medium series electronic components with embedded processors. With SMD parts there is the problem of programming because many expensive and specialized adapters are required. An additional advantage of ISP/JTAG is the practically unlimited reprogrammability of the CPU's.

This UPP-Programmer is distinguished by minimum size, extensive software and ease of use. This manual is only valid for the **UPP1-X** versions of programmers.

FEATURES

- Connection to the PC through a USB port. USB2, USB1 and Hubs supported
- No power supply necessary. The unit is powered directly from USB, **internal battery** or the target.
- Adapts automatically to the target's voltage (1.8-5.5Volt)
- Minimum voltage 3.3V if powered from the target system
- Power consumption 40..60mA if powered from the target system
- Easy and extensive software. A project can be stored on the included micro-SD card.
- Software runs under Windows XP, Vista and WIN7,8 / 32 and 64bit
- Small, lightweight and handy unit - 110x55x20mm
- Supports all **SPI, JTAG, PDI, TPI** and **UPDI** programmable AVR's
- Supports the SPI programmable AT89Sxx types
- Supports the Atmel AT89LPxx family
- Supports the ChipCon CC1110, CC2510 und CC2430 family
- Supports the serial flash (SPI-Flash) families AT25DFxxx, S25FLxxx and SST25VFxxx
- Programmable supply voltage (source) for the target system 30mA..300mA 1.8..5.5Volt
- Extremely fast, programs a full Mega128 in **3sec** (JTAG) and a Tiny44 in **1sec** (16MHz)
- Self update feature via the PC
- Very well suited for production programming by using high secure **AES** encryption
- Processes AES encrypted project files (PAC) onboard
- Supports programming cycles limitation with AES-PAC files
- UPP1 devices with the **Option "S"** provide a project selection out of 10 stored projects on the micro-SD card by the help of up-down buttons and a single digit **LED display** This is very useful and has many advantages for portable (offline) operation.
- micro-SD card included. FAT16 up to 2GB and FAT32 up to 32GB and SDHC types are supported.
- external switching power supply with 5Volt/1000mA included
- Car adapter for cigarette lighter included
- Internal Battery Pack 3,7V/700mAh included. Option 3,7V/1500mA available
- Option adapter from Atmel 6pol SPI to Atmel 10pol programming connector (not included)
- Option adapter from E-LAB JTAG to Atmel JTAG programming connector (not included)
- Option adaptor from E-LAB UPP1-X to ChipCon CC2430 Evaluation Board (not included)

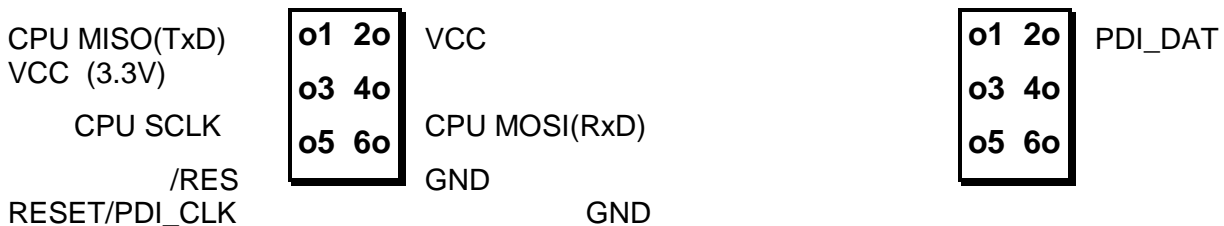
CONNECTIONS

XP, Vista or WIN7/8 is required. An USB-1 or USB-2 port is required. With an USB-1 port the programming time will be increased somewhat.

The included USB cable must be connected to a free USB connector on the PC. The USB drivers must be installed once in order to work properly. See the USB section at the end of this manual.

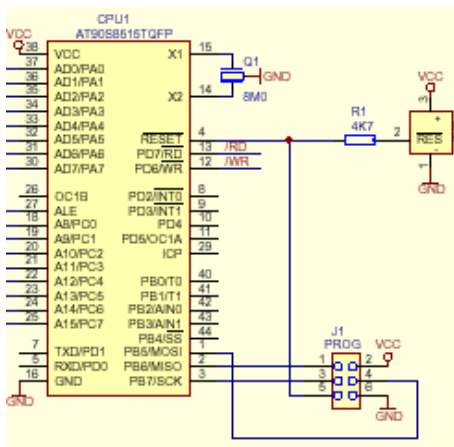
The internal voltage is 3.3Volt. Don't apply high loads at the target pins used for programming. No capacitors are allowed at these pins. Capacitors at the RESET pin must not exceed 100nF. With XMegas no capacitor is allowed at reset. Here the pull-up should not be lower than 100kOhm.

The definition of the 6-pin target plug (0.1 inch pitch male header, dual inline) conforms to a recommendation from Atmel. The **TOP VIEW** onto the connector of the Target is below:



Top View AVR SPI header

Top View XMEGA PDI header

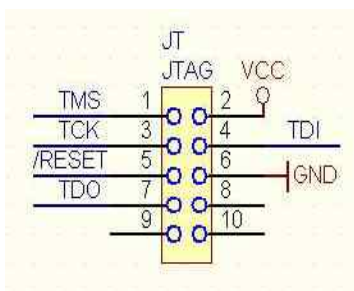


Pin1 of the 6pin plug of the programmer can be located by a small triangle on the front of it. A misalignment of the plug leads to malfunction and can possibly **DESTROY** the unit.

The working voltage of the Target CPU must be in the range of 1.8V..5.5V. XMegas must be supplied with max. 3.6V and the TINYs not below 5.0V.

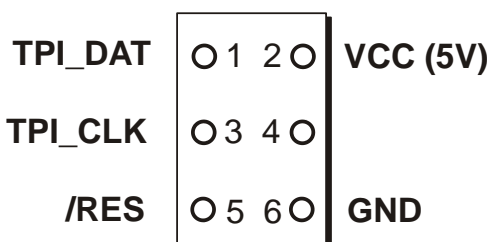
None of the 4 control lines of the device must be shorted. A **continuous short circuit** can destroy the programmer. Only electrically tested boards should be connected.

JTAG programming



For JTAG programming of the target CPU the 6-wire ribbon cable must be replaced by 10-wire type. Because the UPP2-X uses the same plug for ISP and also for JTAG programming, the JTAG plug on the target system differs from the original Atmel JTAG plug. See the schematic on the left.

On the left is the recommended E-LAB plug connection which must be used for the target system if the JTAG interface of the target AVR is to be used for programming. Please note that also the **/RESET** line must be connected to the target CPU.



Tiny 4, 5, 9, 10, 20 programming

These TINYs must be programmed with 5V through 3 pins.

The programming mode is called TPI. Because the same programming plug is also used by the SPI mode the plug on the target system differs from the Atmel plug. The load on the TPI_DAT pin must not be lower than 80kOhm.

SOFTWARE

There are two supplied programs that can be used to control the UPP1-X programmer. The first is **AVRProg.exe** which is the more extensive program that can be used for creating a project, editing/viewing the data and selecting fuse, lock and programmer setup options, such as power supply, programming and verifying the target and creating packed or encrypted programming files for use elsewhere.

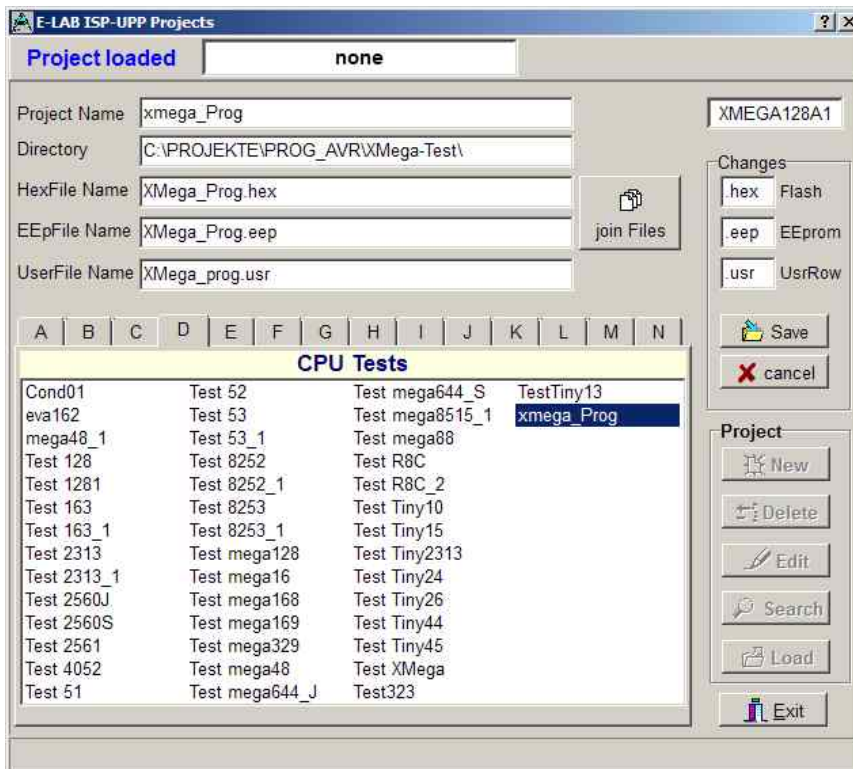
The second program is **PackProg.exe** and it can only be used for programming or verifying using a packed or encrypted file that has been previously created by AVRProg. It is the most suitable program for production use and is described later in this manual.

The UPP1 can also be used in **Stand Alone** mode (i.e. without a PC attached) and this is described later.

AVRProg

AVRProg can be executed from within the AVRCo IDE (PED32) and in this mode all the project settings are passed directly from the IDE. It can also be executed directly from a shortcut or the Start Menu. In this mode an existing project has to be opened or a new one created so the project select dialog is opened on startup.

Open Existing Project: A project can be opened and loaded by a double click on the desired entry or by a single click on the entry and an additional click on the Load button. All project related parameters and files are loaded.



Details of the highlighted project from the select window with CPU type.

Accompanying project path.

The Flash Hexfile with file extension.

EEProm file with extension

XMega UserRow with extension

Store changed or new project.

Commands

Build a new project

Delete a project

Edit an existing project

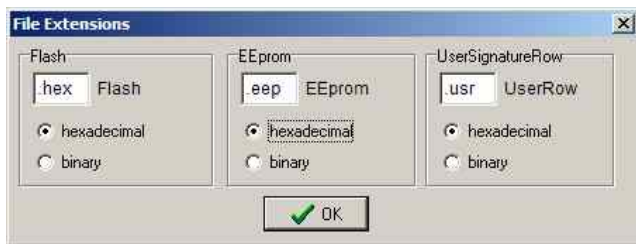
Search project on network

Load a project

Exit this dialog.

With each project it is possible to store a text comment. See 'Comment' dialog below.

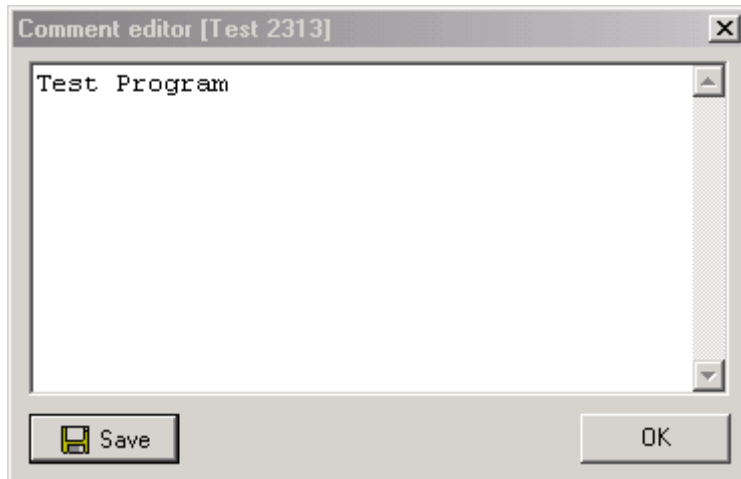
Build a new project: Select the desired target page of the tabbed notebook. Click button **New**. Type the desired name into the field **Project Name**. Click to the field **Directory**. From the appearing dialog select the desired directory. Now the dialog for selecting the file extensions and file types appears. Select/edit extensions and file types. The Flash file dialog appears. Select the file which contains the Flash contents. Finally the CPU type must be selected from the CPU-Select dialog. Up to here the selections are a must. The following dialog for selecting an EEPROM file can be ignored, if nothing exists. The new project must be stored by the **Save** button.



Dialog for the file types and extensions of a project. These parameters are project related and must be defined for each new project.

Editing an existing project: Click the button **Edit**. The program is now in edit mode. With a click onto the items and fields the accompanying dialog opens. After all changes are done, store them with a click onto the **Save** button.

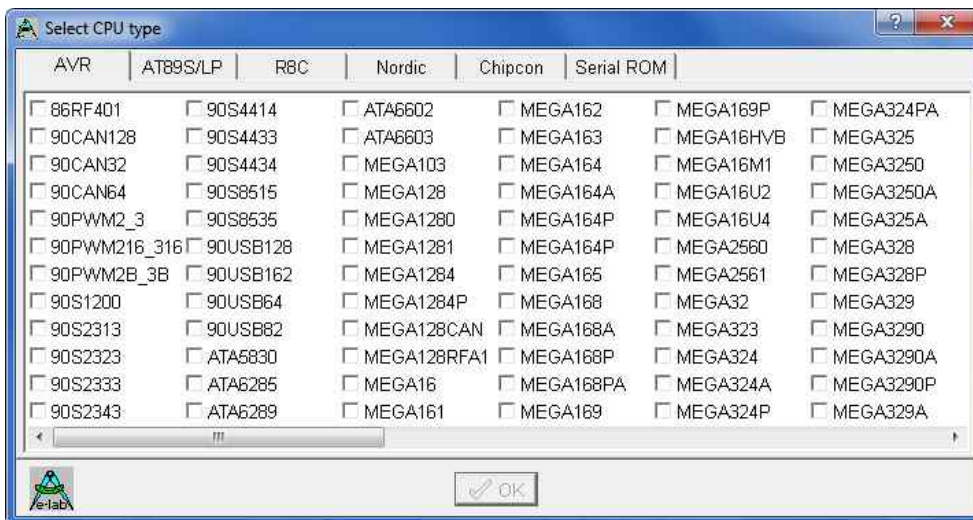
Moving a project from one page to another page of the notebook is very simple. Select the project, enter the edit mode, switch the notebook page to the desired page and then store the project with the save button.



Comment Editor

A right click to an entry in the project select/load dialog opens this Comment dialog.

With this dialog it's possible to add or view a comment for each project.



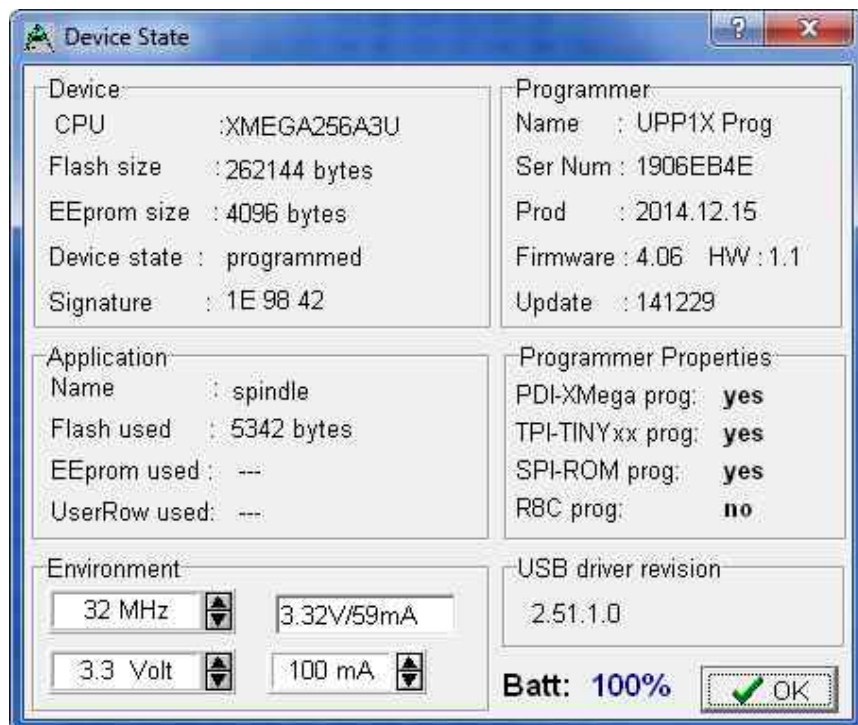
Dialog for CPU Selection

If a project is being created initially or an existing project must be changed, first the CPU type must be defined. This is the purpose of this selection dialog.

Device state dialog



After selecting the CPU the clock-frequency of the target must be set. To do this, the button opens the Device State Dialog. The dialog shows the parameters of the PC-loaded project and the parameters of the hex-files loaded.



Editable parameters are located in the **Environment** group.

The clock defines the SPI speed for **ISP** programming and can be changed every time. If this value is too high, the programming can fail.

For **JTAG, TPI, PDI** programming mode this value has no meaning. But this parameter should have always the correct value.

The voltage field on the right of it reflects the current voltage value which the ISP programmer measures on the target board, if connected.

The current field selects the maximum allowed current to supply. 0.0mA sets the internal supply to the off state. The other values enable the ISP-internal supply. If a current > 0mA is selected the desired supply voltage can be selected in the voltage field. If enabled the UPP1-X programmer supplies the selected voltage to the target system (only in programming state). The current will be limited to the selected value.

USB and Power Supply

The UPP has an internal Step-Up Converter. It creates an output voltage up to 5,2V from the USB or battery voltage (4..4.8V).

To supply the UPP by the target its voltage must be at least 3.3V.

| | | |
|----------------|------|------|
| Target voltage | 3.3V | 60mA |
| | 4.0V | 40mA |
| | 5.0V | 40mA |



The UPP1-S has no dedicated power supply connector. The USB connector is alternatively used to connect a mobile phone power supply. Such a switching supply 5V/800mA is included and also a car charger.

The **Application** group shows the actual loaded project in the PC.

The **Programmer** group shows the information about the connected programmer device: programmer type, serial number, date of production, firmware revision and the last firmware update. The last item is important because all update files start with their date-of-build **yy-mm-dd**. In the dialog above this is 2011, Feb, 21.

So you can easily find out whether a new downloaded file from the WEB is the same or newer than the one already downloaded in the ISP.

The download of an update into the ISP is described below in the section **Firmware Update** at the end of this manual.

Call options

Start with the Windows Explorer

If you make a link in the Windows Explorer from the file-extension ***.ispe** to **AVRprog.exe**, a double click to a project-infile xxx.ispe in the Explorer automatically invokes AVRprog.exe, which by itself reads the inifile and loads the complete project.

Start within the E-LAB IDE PED32

The program start is already implemented into the IDE.

Command line parameters

With all calls of AVRprog command line switches can be appended. These are:

- USB2** Only search for devices with USB2
- e** Automatic Device erase
- p** Automatic Programming Start
- r** Automatic Target Run
- c** Program exit
- s** No visual error messages are generated. Instead the errors are written into the file 'ICPISP.err'.
This file then can be found in the project path or in the program directory if no parameter specifies the project path.
- u0** A standard Pack File will be build
- u1** An encrypted Pack File will be build
- gProgSerNum** If more than one Programmer is found then the Programmer with the serial number **ProgSerNum** is used.

The order of the switches in the command line doesn't matter. The internal processing is always done in the above order. The switches must be separated by spaces. A switch must not contain spaces.

Example:

C:\pppp\AVRProg.exe ProjectName -USB2 -p -c

If the switch **-p** is active, a previous erase (**-e**) is not necessary because a programming process always first erases the target.

If the switch **-c** is active, a previous Target RUN is not necessary because a program exit also releases the target.

With the parameter **ProjectName** there are 2 possibilities. You can pass the complete path and name of the desired control file like: 'C:\files\hex\myprog.ispe'. Or only the name of the project is supplied like: 'myprog'

Return Codes

| | | |
|----|--------------|---|
| 0 | dsOk | Operation successful finished |
| 1 | dsPwrDown | No Target voltage |
| 2 | dsPwrErr | Target too high or too low |
| 3 | dsFalseTyp | Wrong CPU ID found |
| 4 | dsProtect | Target is protected by fuses |
| 5 | dsNotEmpty | Target is not empty after an erase |
| 6 | dsVerifyErr | Target or Programmer found a Verify error while programming |
| 7 | dsFileError | N.A. |
| 8 | dsTimeOutErr | USB driver returns a timeout |
| 9 | dsCommError | Communication problem with the Programmer |
| 10 | dsNoProg | Programmer not found |
| 11 | dsNoProj | Project not found |
| 12 | dsFwLost | Programmer returns an invalid firmware |
| 13 | dsNotfound | File, erg. Hexfile, was not found |
| 14 | dsCalReq | Programmer returns a lost or illegal calibration |

Networks

Some networks, e.g. Novell, use DOS 8.1 style filenames and cannot handle the Project File Extension **.ispe**. This can be solved by changing the corresponding entry in the INI-File of the programmer **ISPISP-3.ini**

Example:

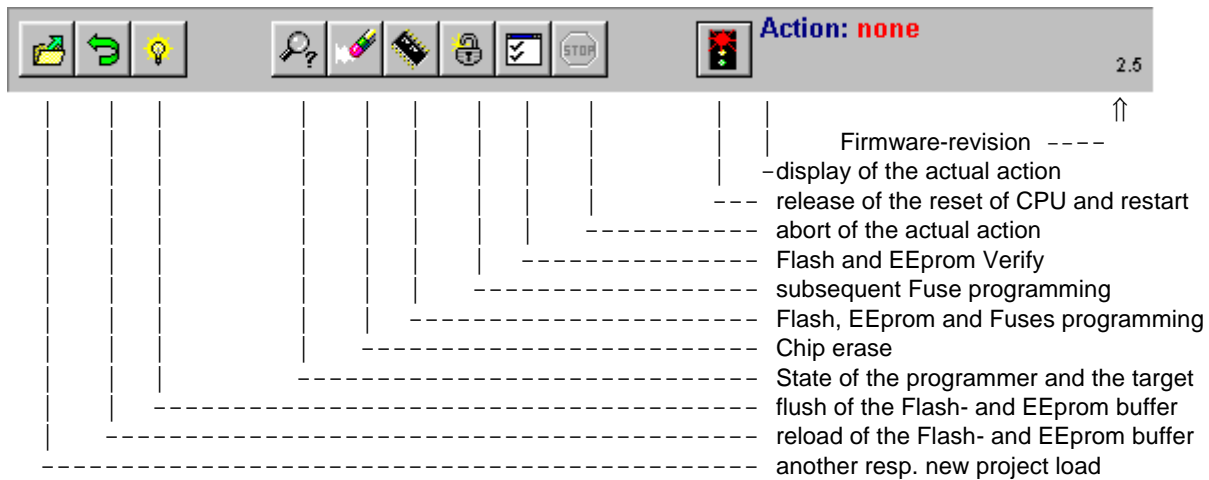
```
[Settings]
ProjExt=.isp
```

Alternative programming

A Pack file can be created that contains all the project information. This can then be used with the PackProg program or transferred to the UPP SD card for Stand Alone Mode. PackProg is included in the UPP1-X programmer package and described later in this manual. This option is also useful when creating a file to send to another location for programming.

Button-Bar

For fast access of the functions with the mouse the program has a bar with speed-buttons. These allow fast working without the use of the menus.

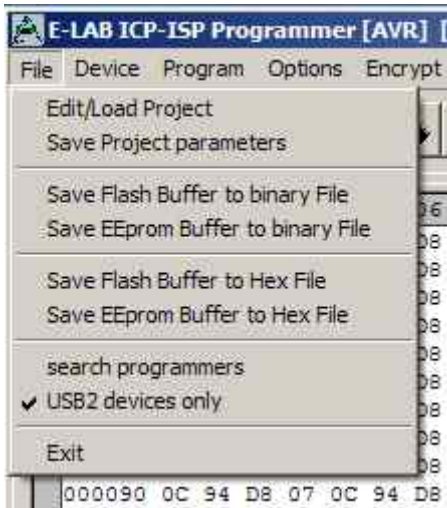


Download button (not shown above) is used to transfer projects to the SD card – see **Project Download** section below.

Function of Buttons and Menus

Normally the use of the menus is not necessary. All standard operations can be started with the speedbuttons by a mouse click. Specialized operations can be found only in the menus.

File Menu



Edit/Load Project opens the project dialog. A new project can be build or an existing one can be opened and loaded.

Save Project parameters stores the actual parameters to the isp-file

Save Flash Buffer to binary File stores the Flash-Buffer into a binary file. A File-Dialog is opened..

Save EEPROM Buffer to binary File stores the Flash-Buffer into a binary file. A File Dialog is opened..

Save Flash Buffer to binary File stores the Flash-Buffer into a hex file. A File-Dialog is opened..

Save EEPROM Buffer to binary File stores the Flash-Buffer into a hex file. A File-Dialog is opened.

Search Programmers is a support function which closes the currently opened programmer connections and then tries to find all connected programmers. See separate section below regarding Multiple Programmers.

USB2 devices only disables the global programmer searching and enumerates USB2 types only. This avoids long timeouts with the COMport searching which can take several minutes if a Bluetooth virtual Comport is installed on the PC.



The **Project-Open** button opens the project dialog. A new project can be built or an existing one can be opened and loaded.



The **Reload** button loads the previously loaded Hex-Files again.

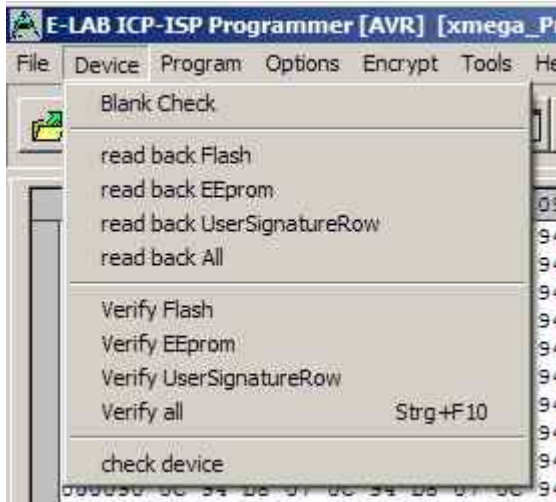


The **Flush** button clears the Flash-Buffer and also the EEPROM-Buffer completely to \$FF



The **Device State** button opens the Device State dialog. Description see above.

Device Menu



Blank Check tests the target (CPU) for unprogrammed i.e. empty. If the target is protected, a message is raised.

Read back Flash if the target is not protected, the contents of the Flash is read back into the Flash buffer.

Read back EEprom if the target is not protected, the contents of the EEprom is read back into the EEprom buffer.

Read back UserSignatureRow if the target is not protected, the contents of the UserRow memory is read back into the UserRow-buffer. Only XMega.

read back All if the target is not protected, the contents of the Flash and the EEprom is read back into the related buffer. With XMegas also the UserSignatureRow is read back.

Verify Flash if the target is not protected, the contents of the target's Flash is compared to the Flash buffer. If there is any difference an error message is raised.

Verify EEprom if the target is not protected, the contents of the target's EEprom is compared to the EEprom buffer. If there is any difference an error message is raised.

Verify UserSignatureRow if the target is not protected, the contents of the target's UserRow is compared to the UserRow-Buffer. If there is any difference an error message is raised. Only XMega.

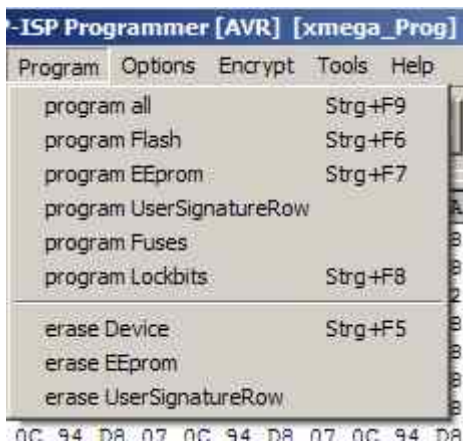
Verify all if the target is not protected, the contents of the target's Flash and EEprom are compared to the related buffer. If there is any difference an error message is raised. With XMegas also the UserSignatureRow is verified.

check device checks the programmer and also the target. If any there are any problems they will be displayed. If the CPU is protected only the device ID can be displayed (ID = 00 01 02).



The **check** button has the same result as the item 'check Device' in the Device Menu above.

Program Menu



program all programs the Flash, the EEprom and also the fuse and lock bits in the manner defined at [Options](#).

program flash programs the Flash only

program EEprom programs the EEprom only (not in JTAG mode)

program UserSignatureRow programs the UserRow (XMega)

program Fuses programs the fuse bits as defined at [Options](#).

program Lockbits programs the lockbits only

erase Device erases the entire device, but not the fuse bits

erase EEprom erases the entire EEprom (not in JTAG mode)

erase UserSignatureRow erases the UserRow to \$FF. (XMega)



A click on the **Erase** button erases the entire chip inclusive the lock bits. Please note that the fuse bits are not erased or changed.



The **Program** button erases the chip completely including the lock bits, then the chip is reprogrammed. The fuse and lock bits are treated as set in [Options](#).



The **Security** button writes the lock bits which are defined in [Options](#). It is required that the chip is not protected until this time



The **Verify** button starts a verify of the target with the buffers. Only possible on an unprotected device.



The **Stop** button aborts the current action.



After programming a device the reset stays active. The target can be released by a click on the **Run** button or again reset without disconnecting the programmer from the target.

If the option **autorelease target** is enabled in the option dialog, the reset is always removed after a programming cycle.

Options Menu

All of the fuse bits and lock bits, Reset polarity etc. and also the whole behaviour of the programmer and it's additional options must be setup at least once for a project. To do this there are two dialogs: the Options Dialog and the Target Options Dialog. These are called with the menu below.



Programmer options starts the Options Dialog where the Fuse and Lock bits can/must be defined and also some other functions.

Target options starts the Target Options dialog where powerful extra functions can be enabled and setup.

Download new Firmware starts a firmware update of the programmer. For more information take a look into the chapter **Firmware Update** at the end of this manual.

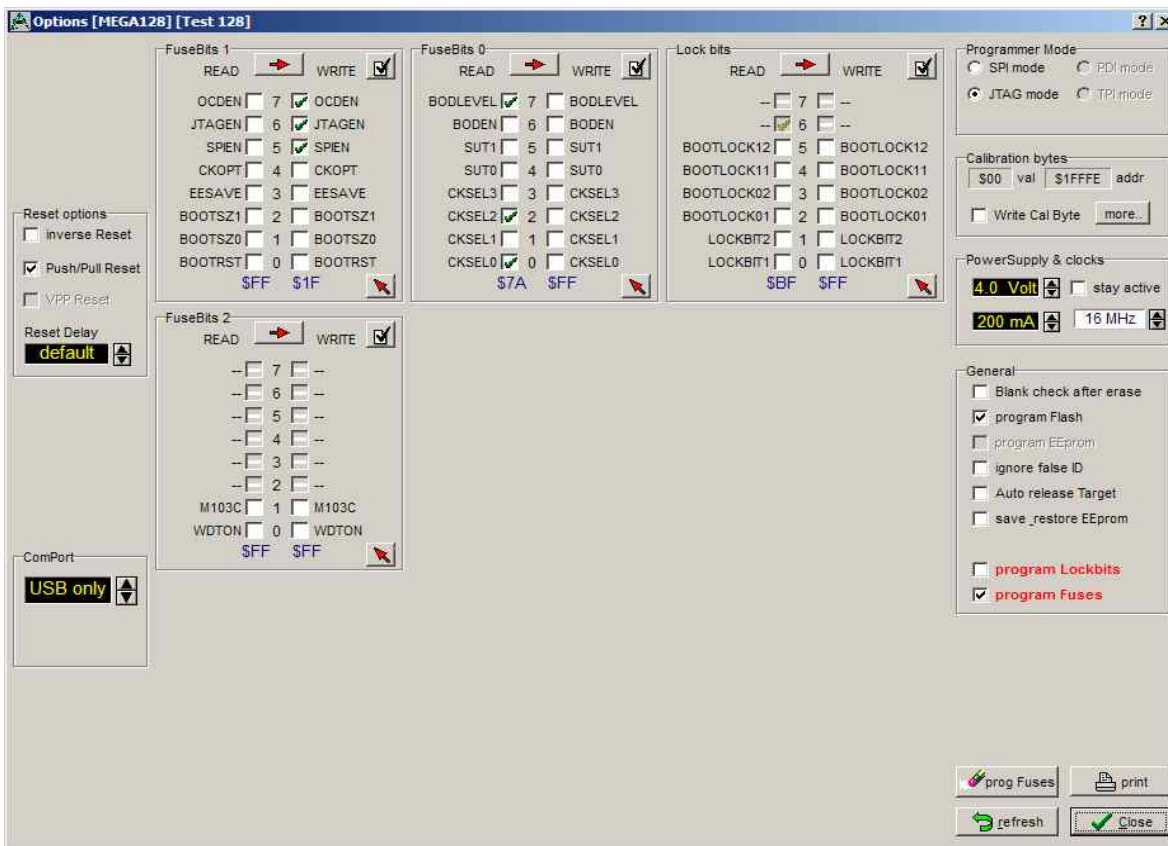
SD-cards size 2..32 are supported. Also SDHC types. Please use only speed grade 4 and above.

For highest speed the cluster size is important. Good values are

up to 4GByte size -> 4kB, 8GByte size -> 8kB, 16GByte size -> 16kB, 32GByte size -> 32kB

Programmer Options Dialog

The Options-Dialog controls the behaviour of the programmer at erase resp. programming of the target. The types of the options depend of the Hex-files and the selected CPU-type..



If, for example no EEprom-file is loaded (.EEP), the item **program EEprom** is disabled.

If the CPU does not support 'Fuse Bits' the **Fuse bits** groups are not visible, otherwise the accessible fuses are enabled for access. The meaning of the Fuse Bits can be found in the Atmel CPU datasheets.

The **General** group defines the common behaviour of the unit. The item **blank check after erase** is only necessary for testing purpose. Normally it should be disabled.

Program Flash and **program EEprom** normally both should be enabled

Ignore false ID disables the error popup in case of a false Device-ID. It is generally unadvisable to check this. Investigate why the wrong ID is being returned.

Auto release Target releases the RESET of the CPU automatically after programming.

Program Fuses should always be checked. Fuses are essential for the AVR.

Program Lockbits also should be checked. But it has no meaning if no Lockbit is activated.

The item **Security** defines the lockbits for the protection modes. **Lockbit0** or **Lockbit1** by themselves make little sense. A complete protection of the device can only be achieved if both bits are active/checked. The BootLock bits should only be activated if the boot section in the AVR is used for booting (self program). The exact function and meaning of the fuses should be observed in the CPU datasheets.

The **Fusebits...** group define various functions as defined by particular CPUs. e.g. Mega128, the fuse programmable internal Reset time. With other CPU types the internal oscillator or the Brown-Out can be defined. Because each CPU interprets these bits in its own way it's impossible to make a general statement here. Some fields can be hidden if the CPU doesn't support some fuses, otherwise the supported fuse bits can be changed by the user. Unsupported fuse bits are disabled. An empty field means that this option is disabled = 1. A green ok means that this option is set = 0 i.e. enabled.

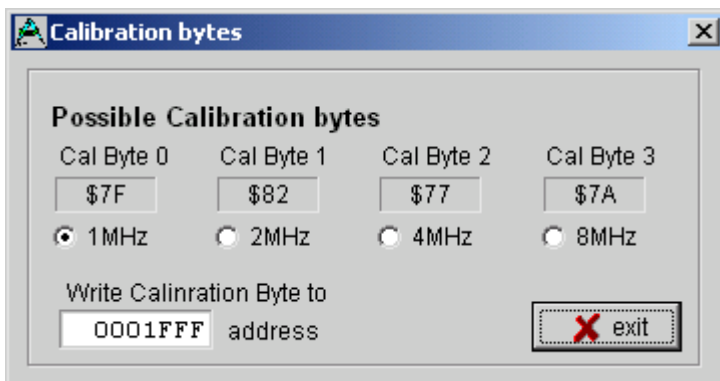
Attention: within the fusebit groups some CPUs have a **SPIEN** fuse. This fuse normally is 'don't care' with SPI-Low-Voltage programming. But there are exceptions, the Tiny12 for example. In this case the SPIEN fuse must be activated. Otherwise the chip is not accessible any more.

The **SPIEN** fuse is always programmable in the **JTAG Mode**. If **SPIEN** and **JTAGEN** are disabled this CPU is also not accessible any more. Note that JTAG mode commandeers some I/O pins from normal use.

- ☒ Sometimes it is more readable for the designer if the binary values of the fuses are shown instead of the boolean values. This can be accomplished with this button.

Calibration Byte Dialogue

Some CPUs which feature an internal RC-oscillator often have also up to four special read-only fuses called 'Calibration bytes', which are shown in the field **Calibration Bytes**. These bytes can be used by the program/application to fine tune the internal RC-oscillator. Because this byte is unique in each CPU it must be individually passed to the application. A checked checkbox **WriteCal Byte** forces the programmer to read this fuse byte and store it into the Flash. The target location of this calibration byte must be supplied by the user with the help of the dialog 'Calibration bytes' shown below.



Dependent of the CPU type there are up to 4 Calibration Bytes which must be read out of the CPU. Each byte corresponds with a unique RC-oscillator. The choice of this byte (radio buttons) is dependent on the settings of the CLKSEL-fuses.

With the edit field 'address' the target address in the Flash for this byte must be set. With CPUs up to 64kByte Flash size this can be each value > \$0000. With CPUs > 64kB Flash (mega128) the selected address must be even.

With newer AVR's and the use of the standard RC-oscillator a Calibration Byte handling is not necessary. These devices automatically read this byte from its EEPROM memory into the OSCAL register.

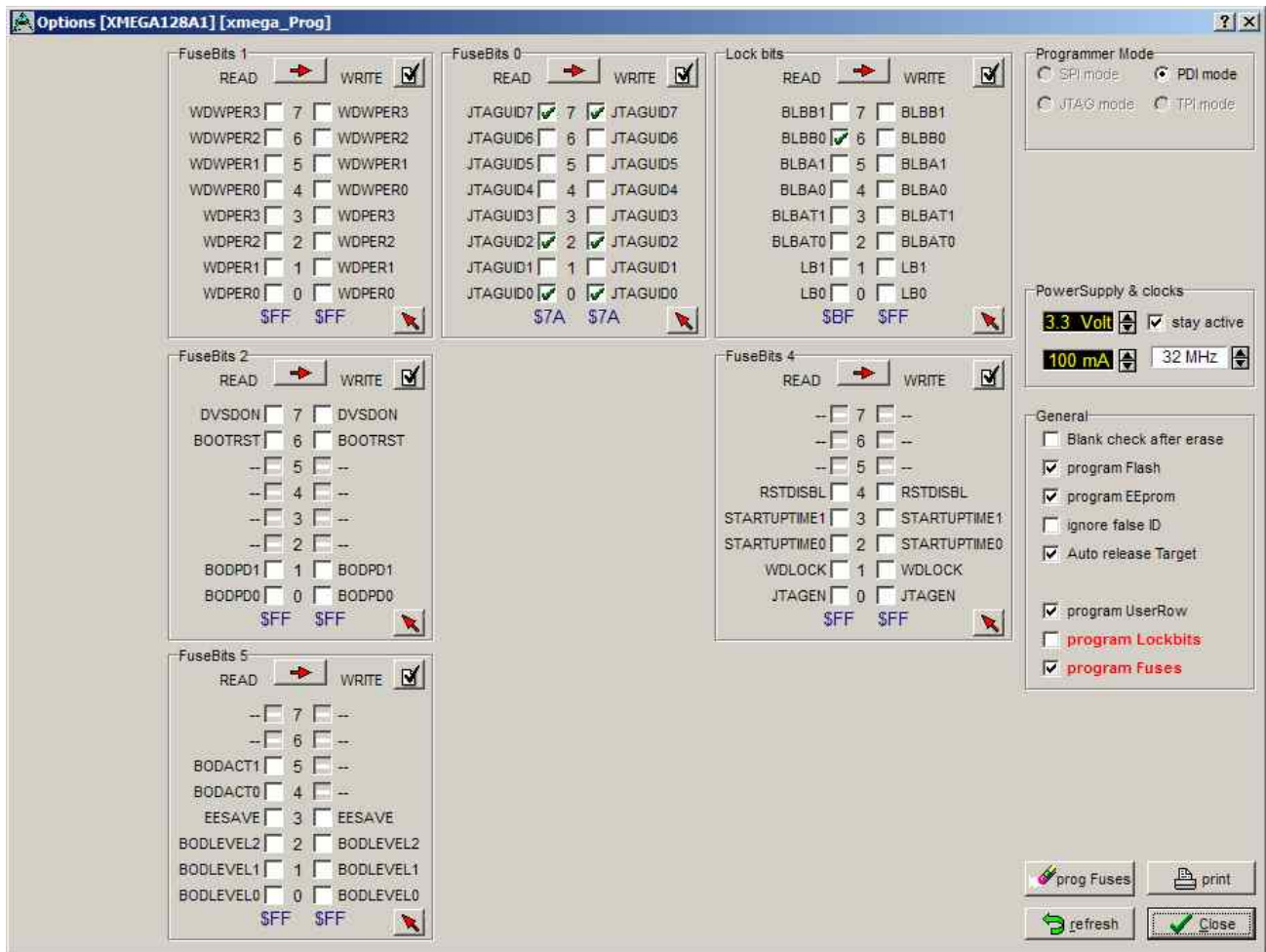
Attention: All Fuse and Lockbits are low active. This means if the data sheet shows a zero '0' for a specific bit the corresponding field in this dialog must show a green 'ok'. Then a '1' bit must show an empty field. Atmel always uses negative logic for Fuse and Lockbits!

With the field **ComPort** the interface to use can be selected. With an UPP1-X the setting 'USB only' should be preferred. If V24 (serial) programmer types can also be connected to this PC 'automatic' should be selected.

The field **Reset options** defines the controlling behaviour of ISP to the target CPU. Normally all 3 items are inactive. For special target hardware the reset level can be inverted. Push/Pull reset makes sense if the reset input of the target CPU is burdened by other electronics e.g. R/C combination. But here the loadings must be reduced by a series resistor of a few kilo ohms. The reset-delay (the time a reset stays inactive before the next activation can take place) can be extended. This is only for special cases.

The field **Programmer Mode** is only visible if the selected CPU supports SPI, JTAG, TPI or PDI programming. In this case one of these modes can be selected.

XMega Options Dialog



With the XMegas some options are impossible or make no sense. The dialog above is typical for an XMEga. Please note that there is the checkbox **program UserRow**. It can only be checked when a hex file for the UserRow is present. (xxx.usr)

The fuse **JTAGEN** can always be unprogrammed because the JTAG interface is never used here. Instead the PDI interface is always used.

Some combinations in the Fusebits1, Fusebits2 and Fusebits5 can be illegal and should be avoided. Otherwise an unexpected behaviour of the CPU can result.

Please note that the voltage supply is limited to 3.6V for XMegas.

Power Supply

The **Power Supply** group selects the maximum allowed current to supply. 0.0mA sets the internal supply to the off state. The other values enable the ISP-internal supply. If a current > 0mA is selected the desired supply voltage can be selected in the voltage field. If enabled the ISP programmer supplies the selected voltage to the target system. The current limiter can be set between 30mA and 300mA. The current will be limited to the selected value.

Basically this power supply is switched off after the programming cycle. If the supply shall continuously supply the target so the checkbox **stay active** must be checked. If in addition the *auto release target* is activated the target system starts up and can be tested at runtime.

The editable fuse- and lock bits are displayed on the right side in the **Write column** and here they can be edited. The **Read columns** can always be updated with the Refresh button. To do this the actual fuse and lock bits are read out of the target as far as possible.

The button **program Fuses** is very useful for erasing of illegal FuseBits, which may be activated by an accidental programming.

One can try with 'program Fuses' to set all fuses to the desired value. Some possible error messages can be ignored in this case. In most cases the CPU then shows a 'normal' behaviour as expected.

Attention (SPI mode):

Some CPU types have an internal RC-Oscillator or the feature to connect an external RC-Oscillator. These options must be selected by some fuse-bits. Sometimes it's also possible to select an external low-frequency quartz crystal. With selecting such an oscillator one must be very careful:

1. Internal RC-oscillator.

With this option selected the standard frequency is typical 1MHz. Because of this the programmer's frequency selector must also be set to 1MHz, otherwise there will be errors with accessing the target CPU. The nominal frequency is 1MHz. With a CPU-voltage of 3Volt the frequency drops to 500kHz.

2. External RC-oscillator.

If this mode is activated, there must be a proper external circuit connected. Otherwise the target CPU will be never accessible.

3. Low Frequency Crystal.

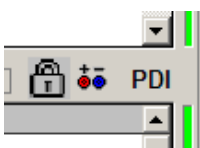
If this mode is activated, a 32kHz watch quartz must be connected to the target. Otherwise the target CPU will be never accessible.

Please note in addition, that while programming, the actual fuses in the CPU are relevant. The new programmed fuses become valid the first time after a reset. Some fuses become valid the first time after a power down.

With accidentally wrongly programmed oscillator fuses it's possible that after that the external oscillator circuit must be changed to again get access to the target CPU.

The above restrictions and warnings are not relevant for JTAG, TPI and PDI programming. Here the CPU must simply be supplied with voltage/current. A working oscillator is not necessary and is ignored. But then never disable the JTAGEN fuse in JTAG mode.

The settings of the Lockbits (protected/unprotected) and the voltage supply of the target by the ISP are also displayed in the main program by two symbols.

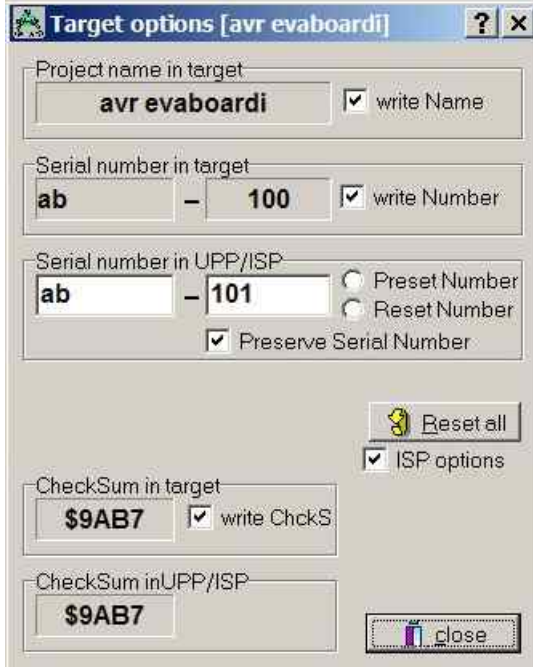


The padlock means that the target will be locked/protected by the lockbits. The mains or battery symbol means that ISP must supply the target system. Both symbols belong to the actual loaded project in the PC program.

Target Options Dialog

This dialog has two jobs:

1. Store/program of additional data into the Flash of the target.
2. Serial number administration.



The screenshot shows the 'Target options [avr evaboardi]' dialog box. It contains several sections:

- Project name in target:** A text field with 'avr evaboardi' and a checked 'write Name' checkbox.
- Serial number in target:** A text field with 'ab', a minus sign, a text field with '100', and a checked 'write Number' checkbox.
- Serial number in UPP/ISP:** A text field with 'ab', a minus sign, a text field with '101', radio buttons for 'Preset Number' and 'Reset Number' (both unchecked), and a checked 'Preserve Serial Number' checkbox.
- CheckSum in target:** A text field with '\$9AB7' and a checked 'write ChkS' checkbox.
- CheckSum in UPP/ISP:** A text field with '\$9AB7'.
- Buttons:** 'Reset all' (with a question mark icon), 'ISP options' (checked), and 'close'.

The caption shows the name of the project present in the PC.

If a project name is programmed into the target and it is readable, it is displayed in this panel. The check button **write Name** shows the state of this option.

Serial number in target shows the read back serial number from the target, if any. The check-button **write Number** shows the state of this option.

Serial number in UPP/ISP shows the serial number in the ISP and PC. It's incremented after every programming. With **Preserve Serial Number** the number in the Target is read back and used for the next programming cycle.

Reset all resets all options, also the serial number.

ISP options enables or disables all options.

CheckSum in target shows the flash check sum in the target, if present. The check-button **write ChkSum** shows the state of this option.

CheckSum in UPP/ISP shows the flash check sum present in the ISP and PC.

Storing Parameters into the target

If there is enough space in the flash memory of the target CPU additional parameters and information can be placed at the end of the flash memory. With the option switches (checkboxes) of the dialog above each option can be individually enabled or disabled.

A change gets saved immediately. Also these options are stored into the project's INI-file as usual. The options, if any, are programmed at the end of a programming cycle into the last bytes of the target's flash.

At each invocation of this dialog there is a try to read the actual-parameters from the target. This operation only works, if the target is present, powered and not protected, of course. The firmware in the target CPU always has access to this data.

Project Name into Flash

If the checkbox **write Name** is activated, the ISP is enforced to write the project's name into the flash. This is done at the end of a programming cycle.

Serial Number into Flash

Activating the checkbox **write Number** instructs the ISP to burn a serial number into the flash. The integer part of this number is then incremented. The serial number consists of 2 parts:

1. Two arbitrary characters from the field **Serial Number in ICP/ISP**. This part stays unchanged.
2. A number in the right field. This number can be zeroed with the button **Reset Number** or preset with a number and the **Preset** button.

The count of the programmed targets up to now can be found here.

Preserve Serial Number

If the serial number is enabled then this option prevents writing a new/different number into the flash. If the target is not protected the internal number in the target is read out and is used for reprogramming the flash. This ensures that the number stays unchanged and the old one is re-used, despite of a new programming of the chip.

Checksum into Flash

While downloading the flash file, a 16bit checksum is generated over this data. This number is displayed in the field **Checksum in ICP/ISP**. This value can be written into the flash by checking the checkbox **write CheckSum**. This is done after every programming cycle. Note that the checksum contains only values from the original Flash-Hexfile. Additional parameters programmed at the end of the flash by the ISP itself are not recognized. Also empty (not addressed) parts in the hexfile are discarded.

Reset All

This button resets all values and options.

Note: all changes are immediately stored. But they are only recognized at the next programming cycle.

The application/firmware always has access to this parameters. The parameters are stored into the last 16 resp. 32 bytes of the flash. The project name has a lead-in of 'proj'. The serial number has a preamble of 'ser#'. The number is always the fourth and third last byte. The checksum, if present, always can be found in the last 2 Bytes of the flash. The order of the serial number and the checksum is loByte/hiByte. Sample:

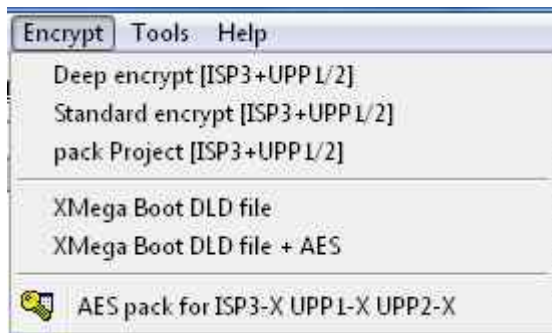
```
01FFFE0 FF FF 70 72 6F 6A 0D 61 76 72 20 65 76 61 62 6F ..proj.avr evabo
01FFFE0 61 72 64 69 BA 00 73 65 72 23 61 62 65 00 B7 9A ardi...ser#abe...
               ↑   ↑           ↑   ↑
             LSB  MSB       LSB  MSB
             ser number   checksum
```

Attention:

programming from the flash card writes the serial number on each successful programming cycle to the card. This means stress to the card and may damage it early. To minimize these problems one should use cards with the attribute **x80**.

These are very fast and promise a 10 times longer lifetime for write cycles.

Encryption



The first four items of the menu are not valid for the ISP3-X programmer types.

The last 3 items are relevant for the ISP3-X.

These menu items open an encrypt or pack dialog. With the help of one of this dialogs **encrypted** or **packed** binary files can be created which can only be read and processed with the included program **PackProg.exe**. All three file types can be used with an ISP3-X as the programmer expects the program PackProg.exe

With **packed** files or projects there is a medium protection. All Hex files and also the programming control, fuses etc. are written into a binary file so the recipient does not need to have great programming knowledge and it is also impossible to change any setups, fuses or file contents.

The **encryption** enhances a packed file in such a way that programming files can be sent to every place in the world and the recipient or others are unable to disassemble them or do any 'reverse engineering'.

Furthermore there is the additional feature to include a **password** into the encryption so only the right ISP3-X programmer can use this file. This is an additional protection against illegal copies.

Deep encrypted files can only be used with the program PackProg.exe

Standard encrypted files can be loaded into the UPP programmer types. But for use with the ISP3-X the utility program PackProg is absolutely necessary.

Packed files can be loaded into the UPP programmer types. But for use with the ISP3-X the utility program PackProg is absolutely necessary.

Deep encrypt Mode

This mode should not be used with the new programmers (ISP3-X, UPP1-X, UPP2-X). These use the much better **AES** encryption. AES creates an **extremely encrypted** project that can only be read and processed with **Prog.exe** utility. As an option an additional password to make sure that only a specific programmer is able to process these projects. This menu shows the following dialogue:

This encryption allows you to ship program data around the world without enabling the receiver or a third party to disassemble or "Re-Engineer" the contents.

The encrypted file (*.enu or *.en#) contains the serial number of the target programmer. This ensures that only this specific UPP is able to process the files.

To generate such a file the creator of the file must know the serial number of the target UPP.

Read Key reads the serial number if the target UPP is connected to the PC:

Press **Read Key** and the number appears in **edit Password**.

Add Key allows adding the number to the list.

If the serial number of the target programmer is unknown, its owner must use **PackProg** to read an exchange it with the manufacturer of the project.

The manufacturer puts this number in the **edit Password** area.

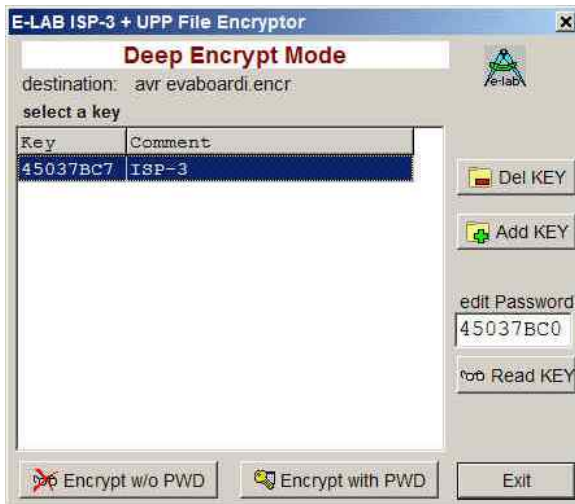
After entering a new password **Add Key** adds it to the list.

To create the encrypted file, select the UPP number from the list (blue bar).

Encrypt with PWD creates and stores the file.

Encrypt w/o PWD creates a file that is not dedicated to a specific programmer. This file is encrypted but not password protected and can be processed by any UPP. Depending on the file extension the created file shows the extension ***.enu** or ***.en0** or ***.en1** etc

The menu item opens the following dialogue:



New passwords are added with **Add Key** button and deleted with **Del Key** button.

Encrypt with PWD button creates a protection password. This ties the generated file to the programmer that generated this password.

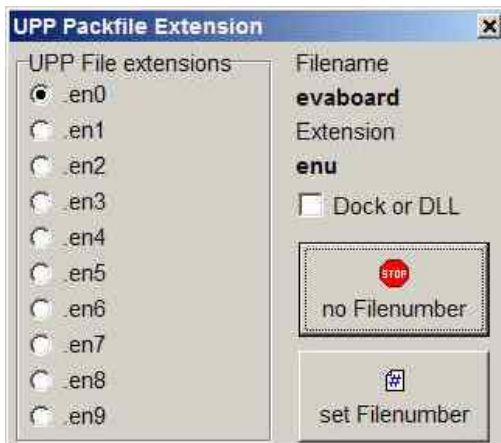
Encrypt w/o PWD button creates and stores an encrypted file without password protection

Generation of an ISP3-X password is also explained in chapter PackProg.

If the target programmer is already connected to this PC button **Read KEY** reads its password.

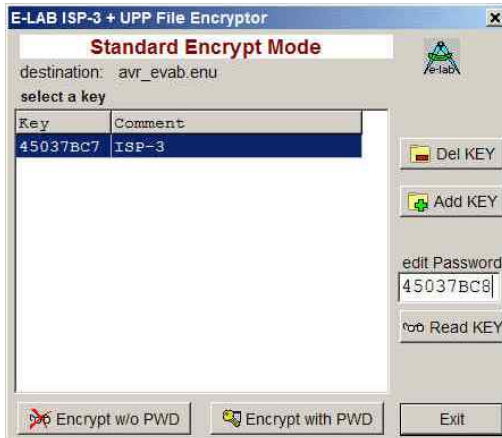
Standard encrypt Mode

This option builds a packed and **well encrypted** project which only can be processed with the **PackProg** program in conjunction with an ISP-3 programmer type. But also a UPP programmer can directly load (using its memory card) and process such a file. As an option, a password can be included so that processing this file is only possibly by the programmer which generated the password. The menu item opens the dialog shown below:



The dialog serves to set the file extension. The extension can include a number, en0..en9 for a better handling with the UPP programmer or a simple *.enu. The choice must be done with one of the file number buttons.

The checkbox **Dock or DLL** is not relevant for the UPP1-X.



If the extension was selected the dialog below opens:

This dialog mainly serves to decide whether a password must be included or not. A password exclusively binds this file to this specific programmer which supplied it. If a password is necessary then it must be selected from the list field.

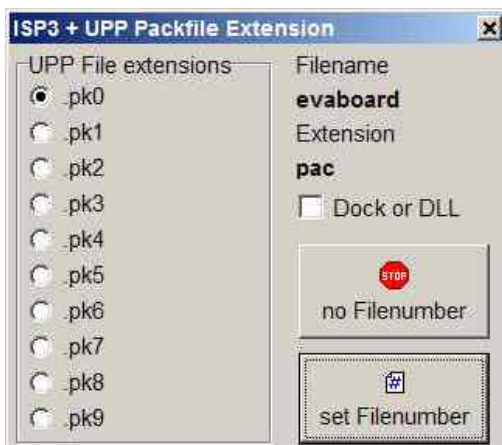
This option must be selected by the button *Encrypt with PWD* or disabled with the button *Encrypt w/o PWD*.

If the target programmer is already connected to this PC the password can be requested. With the button *Read KEY* the password is read out of the programmer and displayed. With the button *Add Key* the new password can be appended to the password pool.

This file type can be processed with the program PackProg.exe and also with all UPP programmer types.

pack Project Mode

This option builds a **packed** but **not encrypted** project which only can be processed with the **PackProg** program. A UPP programmer also can directly load and process this file type. There is no password possible and so every ISP-3 or UPP programmer can use this file.



The menu item opens this dialog.

The dialog serves to set the file extension. The extension can include a number, pk0..pk9 for a better handling with the UPP programmer or a simple *.pac. The choice must be done with one of the file number buttons.

The checkbox *Dock or DLL* is not relevant for the UPP1-X.

This file type can be processed with the program PackProg.exe and also with all UPP programmer types.

Procedure

1. Create a project as usual or load an existing project.
2. Select the proper setups and properties for this project as usual.
3. Select one of the three file modes, heavy encrypted, encrypted or packed.
4. The two encrypted types provide a password protection so that only this programmer which supplied the password can process it.
5. Click the button 'Encrypt' or one of the file number buttons.
6. Copy the generated file to a storage media.
7. If the receiver doesn't own the program '*PackProg.exe*', also copy this tool and this manual onto the media.
8. Ship the media to the receiver.

Shipping this file(s) via email is also possible. In this case it's a good idea to 'zip' all files for secure internet transport.

Encryption

The encryption feature uses a secure algorithm. Unfortunately up to now there is no absolutely secure (uncrackable) encryption. But the time investment to decrypt the file is so high and expensive, it's cheaper and faster to completely start an new program from scratch ☺

Password

The program '*PackProg.exe*' on demand reads out the internal password of a connected ISP-3 programmer. This password is only valid for this programmer and is not portable to other programmers. Because of this it's assured that programmer files containing this password can only be programmed with this programmer.

Procedure

1. The chip or board programmer must install the program *PackProg* on the computer which will be used for the chip programming. To do so the file '*PackProg.exe*' must be copied into the desired directory.
2. The program '*PackProg.exe*' must be started.
3. The UPP1-X programmer must be connected to this PC.
4. The password (displayed with the menu item 'Setup/request Password') must be passed to the creator of the programming files (packed or encrypted projects).
5. The creator then inserts the password in the program '*AVRprog.exe*' into his system:
 - a. Start of '*AVRprog.exe*'
 - b. Open the Encrypt/Pack dialog with the menu item 'Encrypt/PackProg Encrypt'.
 - c. Click the button 'Add'.
 - d. Add any comment or name into the comment field.
 - e. Insert the password into the password field.
 - f. Store all with the button 'Add'.
6. With files which are sent to this recipient with a password it must be clear that only the correct password must be used for the file generation. An alternative is item 7
7. With the button 'Public' there is no password included and all recipients who have the program '*PackProg.exe*' can process this file.

Programmer

The program '*PackProg.exe*' supports all programmer types *ISP3*, *UPP1* and *UPP1*. Find additional information below.

AES PAC files

If absolutely secure PAC files are needed for the types ISP3-X, UPP1-X or UPP2-X ('X' programmer) they must be created with **AES encryption**. AES is an absolutely secure encryption which can't be 'hacked'. Because the encryption can only be done **in** the 'X' programmer types, listening (sniffing) on the USB lines results in unusable data.

Furthermore the AES mode has the advantage to create a PAC file for a **specific programmer** using its serial number so this file can't be used on any other programmer. As an additional feature the **Quantity limitation** can be used. So hidden 'black' productions are absolute impossible.

So it makes sense to use the AES encryption at least for external production. With in-house production the 'standard' PAC file is sufficient.

| | | |
|---|---------------------------------------|-------------------|
| 1 | pack Project [ISP3+UPP1/2] | Standard PAC file |
| 2 | pack Project for ISP3-X UPP1-X UPP2-X | AES PAC file |



To create an AES PAC it is mandatory to have an 'X' programmer connected to the PC.

The dialog for creating of an AES PAC file provides three different modes.

- Standard Encryption (only AES used)
- Encryption with password (Sernum used). Here the serial number of the target programmer is needed. It will then be integrated into the encryption and checked by the target programmer. The target programmer is not needed here, only its serial number.
- Quantity limitation (Sernum+Quantity used). Also here the serial number of the target programmer is necessary. Furthermore the maximum allowed programming cycles are preset. If this number is reached/exceeded the programmer ignores further programming attempts.



The AES properties of a PAC file can be requested with the tool PackProg by 'File Info' from the target programmer.

Attention:

AES encrypted PAC files can be processed by every 'old' programmer type (ISP3-USB, UPP1-USB, UPP2-USB) but because they are unable to decrypt such files only nonsense will be programmed.

Programming directly



Interactive programming must be started with this button of the PC application.

The entire chip is erased including the lock bits and then totally reprogrammed. The fuse and lock bits are treated as described in Options.

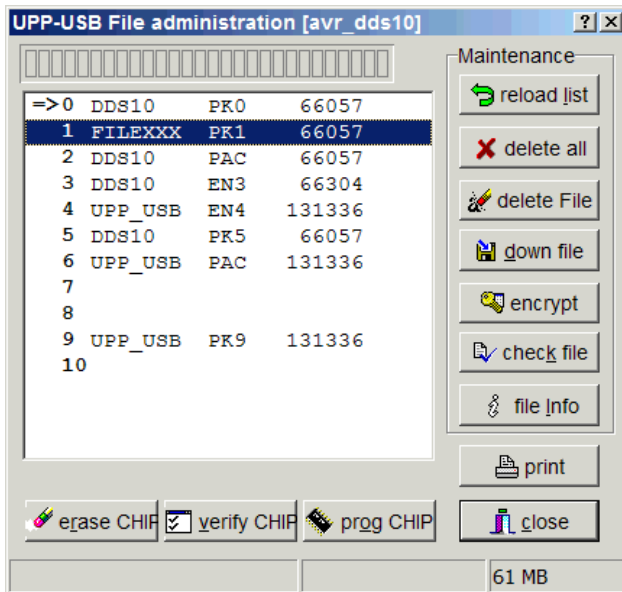
The MMC card is not necessary as the PC and the program has all necessary data.

It is further possible to program via PC but to use a project from the MMC card.

The download dialogue is opened by the



button.



Select the project by a click in the list box.

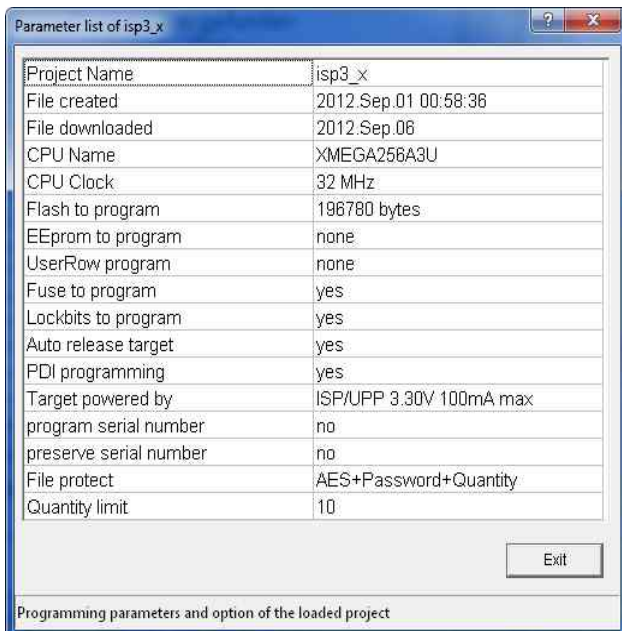
This enables the erase, verify and pro buttons.

prog Chip programs the CPU

If the chip is not read protected **verify Chip** reads and compares the CPU with the PC memory.

Print outputs a hardcopy of the list to the printer

Please note that with portable use (offline) only project number **0** is available for programming



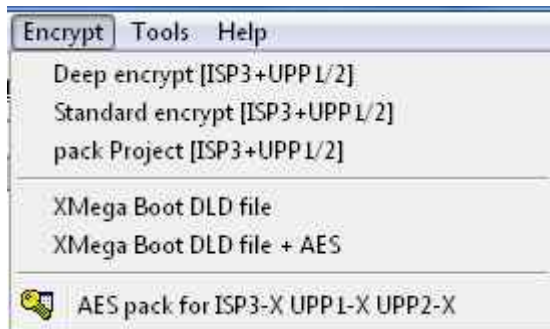
File Info shows the relevant parameters of the selected project.

Project download

The UPP programmer has several working modes. The PC-direct (Transparent Mode) mode doesn't need an SD flash card, and all other modes use an SD card with a stored project.

There are several possible ways to store a project into an SD card.

Flash card Reader/Writer



The PC recognizes this device as an additional removable drive.

With the E-LAB control program a packed file must be compiled from all parts of the project. To do this the desired project must be loaded. Then the Encrypt menu must be opened. A click onto the item

pack Project or **Standard encrypt UPP** opens a file dialog for storing the project in the packed format.

If the SD drive is chosen (reader/writer) the packed file is stored on the SD card.

As an alternative this file can also be stored onto another media, a floppy (remember them?) or USB stick for example. Or it can be sent as an attachment of an email.

Please note that with a UPP1 programmer as the target device only the options with **[ISP3+UPP ...]** should be used. Packed files built with the option **Deep encrypt** never can be directly used with the UPP. The program **PackProg.exe** must always be used.

With **packed** files or projects there is a medium protection. All Hex files and also the programming control, fuses etc. are written into a binary file so the recipient must not have great programming knowledge and it is not necessary but also impossible to change any setups, fuses or file contents.

The **encryption** enhances a packed file in a way so that programming files can be send to every place in the world and the recipient or others are unable to disassemble them or do a so called „re-engineering“.

Furthermore there is the additional feature to include a **password** into the encryption so that only the right programmer can use this file. This is an additional protection against illegal copies.

Deep encrypted files can only be used with the program PackProg.exe

Standard encrypted files can be loaded into the UPP programmer types

Packed files can be loaded into the UPP programmer types

pack Project for ISP3-X UPP1-X UPP2-X

In this mode always the absolutely secure AES 128bit encryption will be used.

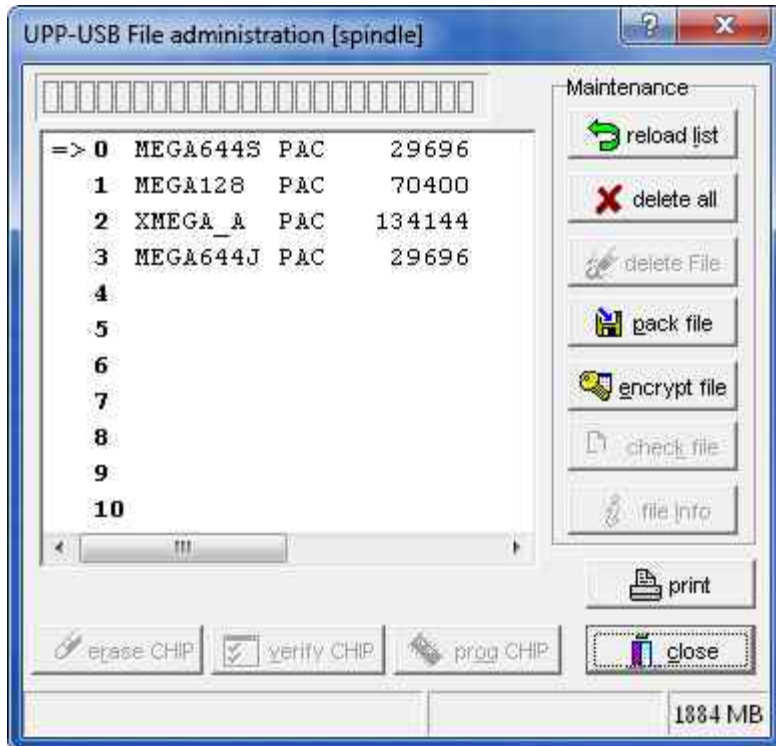
Note:

New Flash Cards must be formatted in a PC Flash drive before the first usage in the UPP. This always ensures that this card is readable in both devices, the PC and the UPP

Programming using the SD card

There is the additional possibility to use the PC for programming by using one of the projects which are stored on the SD card of the UPP.

To do this the download dialog below must be opened with the download button.

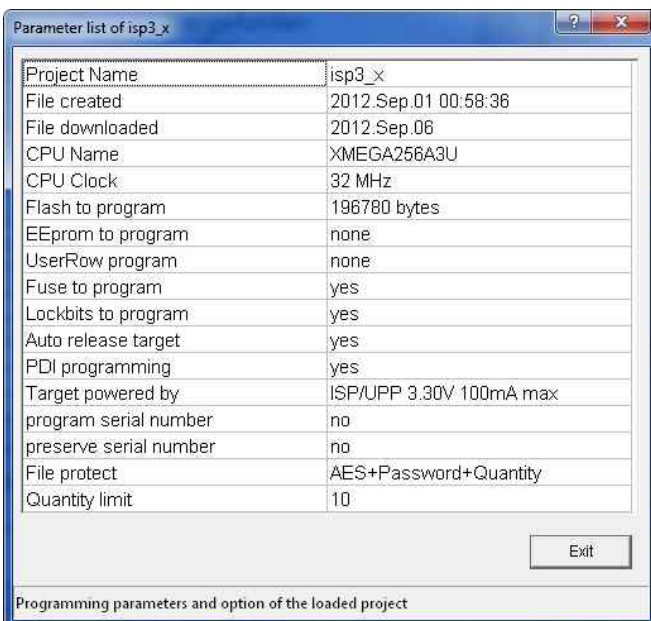


The project stored in the SD card must be selected by a click onto the entry in the list box. Now the erase, verify and prog buttons are enabled.

The programming can be started with the **prog Chip** button.

If the chip is not protected it can optionally be verified with the **verify Chip** button.

The current file list in the SD can be printed out as a hardcopy with the **Print** button.



The **file info** button shows some relevant programming parameters of the loaded project.

States, Error Display and Problems

Possible error messages of the programming system (AVRprog - PC):

Programmer not found

- a. The programmer is not plugged into the PC's COM port (or USB port).
- b. The COM ports of the PC are all used by other devices (N/A for USB).
- c. The connection is missing some lines in the cable (N/A for USB)

Please observe the paragraph **Connection**

Target Power down

- a. The 6 pin plug is not connected to the target
- b. The target is without power or the voltage is too low (< 3Volt)
- c. The project setup expects that the ISP powers the target, but the power supply of the ISP is not connected or the current consumption of the target is too high.

Device not responding

- a. The voltage of the target is too high or too low (see below)
- b. Target has no clock (SPI Mode).
- c. Chip defective
- d. Reset is not connected to the target CPU

Wrong Device ID

- a. The voltage of the target is too high or too low (see below)
- b. Wrong device selected
- c. Chip defective

It is possible that a device ID in the target is permanently destroyed, but the device works correct
If it is the correct CPU-type the programming can be continued.

All the above problems can also be a problem of a defective programmer. In many cases the Reset Pin is loaded with Rs and/or Cs. In this case activating Push-Pull Reset can help.

If a CPU shows a wrong device ID at programming, nevertheless one can continue with programming.

Bluetooth Interfaces

Many of these interfaces emulate a virtual COMport in the PC. If the programmer software in *Programmer Options* sets the interface to automatic then in addition to the USB drivers also all COMports are scanned. This can lead to unexpected long wait times until the Bluetooth returns with a timeout regardless whether any programmer is connected or not. In this case the programmer port selection must be set to **USB only**.

Messages in Stand Alone Mode

Without a connected PC the UPP Programmer works with his MMC flash card only.
The states are displayed optically and acoustically. See also *Stand-Alone Mode* below.

Hard Reset

A hardware reset of the programmer is in both modes (PC controlled and stand-alone) possible by simultaneously pressing the Prog and Era switch.
The UPP1-X has also a RESET pushbutton on the lower case cap.

Building Project Files for UPP1 programmers

The previous pages introduced two ways that UPP Project files can be created. Either by direct download into the SD card in the UPP or through a Flash drive of the PC.

There was a notice that the standard UPP version can only use one project in the portable mode because there is no way to select a project with the UPP itself.

Because of this there are the **Version S** and **Version D** of the UPP1, where the version „S“ has a rotary switch on its back side which supports the selection of one project out of 10 stored projects. The version „D“ in combination with its Docking Station also supports selecting one out of 10 projects.

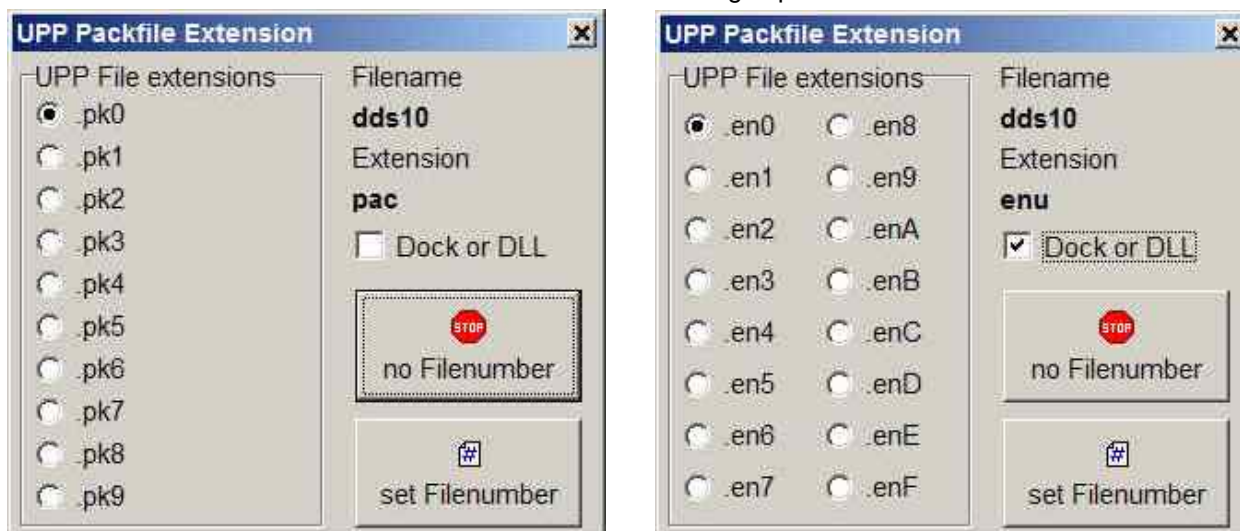
With selecting a project in these ways there is the problem that the relation of projects on the SD card to the position of the switch is not static. It absolutely depends on the order of the FAT16 directory entries on the SD card. If a card becomes completely erased and then projects are stored sequential onto the card the order of the storing absolutely corresponds to the switch positions.

But if then files are deleted, rewritten or updated this relation can change dramatically. As a consequence of this after each SD content alteration the resulting new file order must be copied from the UPP File-Dialog of the PC program. Without taking care of this there can be strange problems with portable programming.

In order to avoid all these hazzles both download functions provide the option to set an absolute relation between a file/project and the switch position of the UPP1.

Basically all UPP Pack Files have the file extension ***.pac** and the encrypted types have the extension ***.enu**. To set a fixed relation between such a project and the selection switch a number between 0 and 9 can be appended to the extension which forces the UPP to fix this project to a switch position.

When a **PAC** or **ENU** file must be created one of these dialogs opens:



If the button **no Filename** is pressed a standard *.pac or *.enu project will be build. If the button **set Filename** is pressed an encrypted file gets the extension ***.en0** and a packed file gets the extension ***.pk0**. The activated radio check defines the last character of the extension and this character now absolutely defines the position of this file in the internal file list of the UPP which furthermore defines its relation to the rotary switch.

The UPP rejects files with **extension numbers** when there is already a file on the SD which has the same extension number. These file types can co-exist:

DDS10.pac
 DDS10.enu
 DDS10.pk0
 DDS10.en1 etc.

PackProg

The main program of the E-LAB programming system, the **AVRprog.exe**, described in the preceding sections, can be used for all kinds of work:

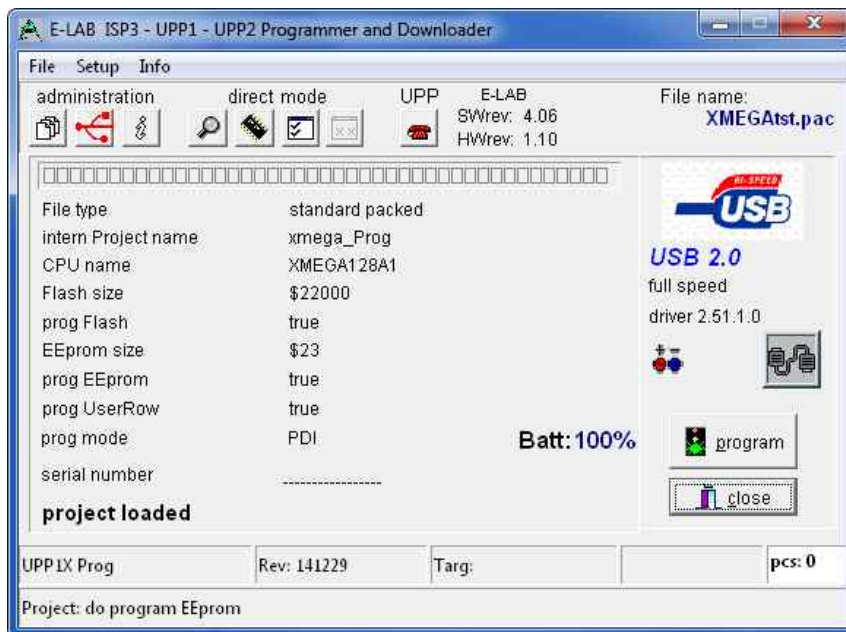
1. for creating projects with Fuse and Lockbits, defining/editing of the Flash and EEPROM files etc
2. for direct In Circuit programming of the Chips with all types of programmers
3. for indirect In Circuit programming (via Flash Card) with the UPP programmer types
4. for creating of packed or encrypted projects
5. for the download of packed projects into the flash card of the UPP programmer types
6. for storing of packed projects onto the flash card in the Flash Drive of the PC.

Most of these functions are not necessarily desired in the production and service area. And furthermore, they distract and can be possible sources for handling errors.

To avoid this there is the pure programming tool **PackProg.exe** for the programmer types ISP3, UPP1 and UPP2. This tool only supports programming of the Chips and with the UPP types the download of packed projects into the programmer. In addition this is the only tool which can process the 'deep encrypted' projects.

Buttons and Menus

Project Import



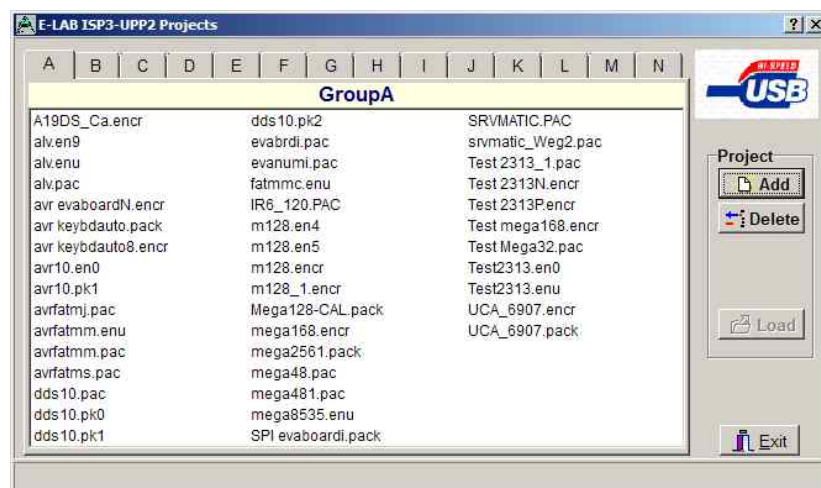
PackProg works with a Project Pool similar to the main program AVRprog. But only packed or encrypted projects that have been previously created with AVRProg can be imported.

This means that a new project must be registered first in order to work with it.

New projects must be registered by the project administration dialog opened by the menu item **open**



or the File button



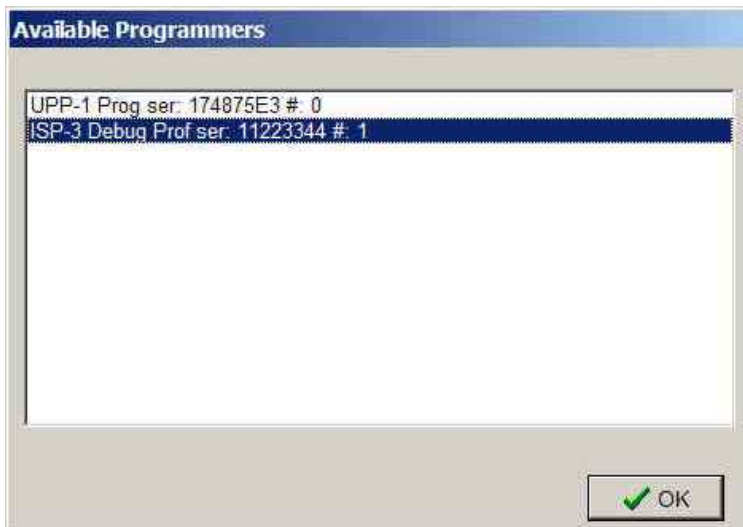
This opens the Project Dialog

With the Add button a new project can be included into the pool.

With the Delete button an existing project can be removed from the pool.

With the Load button or a double click onto a list entry the selected project is loaded.

Searching for Programmer



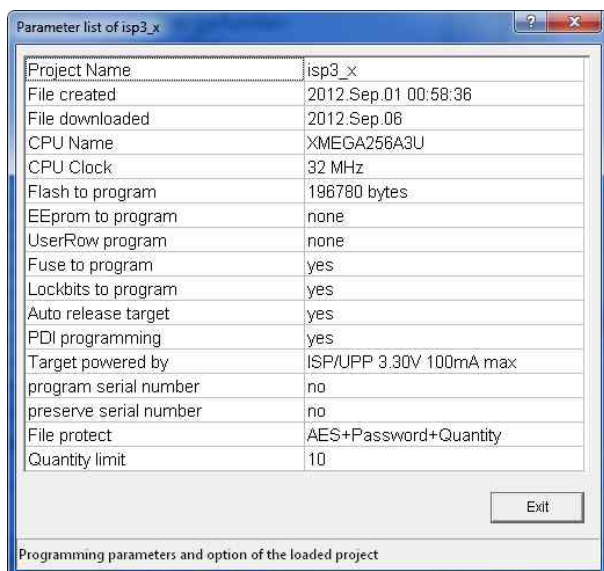
Before one can work with the selected and loaded project the programmer must be searched for. This must be done with the Check-USB button.

If a programmer was found this is displayed in the main window.

If more than one programmer is connected and found then this dialog is raised.

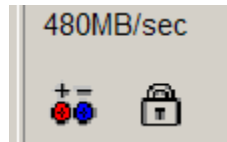
Because only one programmer at a time can be used the desired one must be selected from the list.

Project Check



With the Info button the most important parameters and properties of the actual loaded project can be displayed.

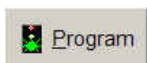
Two items of project information are also displayed in the main window:



The Battery symbol is visible if the programmer must supply the target with a voltage/current.

The lock symbol is displayed if the programmer must lock/protect the target.

Program



Program Chip. These buttons start a complete programming cycle of the chip. This includes the Flash, EEPROM, Fusebits, Lockbits and eventually a serial number.

Which of these operations are executed is controlled by the content of the packed or encrypted file (Project).

Because the control program knows and has access to all relevant data, the SD card is not necessary and is not used.

Device Check



Device Check. With this button a check is done of the programmer and the target device. A check of the target's supply voltage and reading and checking the device ID of target is included, if possible.

Verify



Verify Chip. Verify the contents of the Flash and EEPROM. Of course this is only possible if the chip is not protected.

If the programming of the serial number was enabled at the project creation time (packed or encrypted file) then the actual number is displayed. With a double click onto this field it becomes editable. At programming time this number is stored into the Flash and after that it is incremented.


Exception: if 'preserve serial number' is activated then the original number out of the target is used.

| | |
|---------------------------------------|-----------------------------|
| prog EEprom | false |
| prog mode | SPI |
| serial number | 8899AAB2CCDDEE3C |
| Target protected | |
| UPP-1 Prog | Rev: 061211 Targ: 3.28V/0mA |
| Infos about the actual loaded project | |

| | |
|---------------------------------------|-----------------------------|
| prog EEprom | false |
| prog mode | SPI |
| serial number | 88 99 AA B2 CC DD EE 3C |
| Target protected | |
| UPP-1 Prog | Rev: 061211 Targ: 3.28V/0mA |
| Infos about the actual loaded project | |

Setup Menu

If encrypted or deep encrypted projects are to be built using AVRProg then the password of the target programmer must be supplied to the file creator. To find the password of the connected programmer

| | |
|---|------|
| Setup | Info |
| request Password | |
| Calibrate | |
| download new Firmware | |
|  Telnet active Telnet config Telnet info | |
| ✓ auto search programmer | |

request Password must be used in the Setup Menu.

| | |
|--|--|
| Encryption Password | |
| 174875E3 |  copy to clipboard |
| close | |
| Get the programmers internal password. Needed to enc | |

The download of a new **Firmware** for the programmer is started and executed as described below.

If the option **auto search programmer** is enabled a programmer search is automatically started the program PackProg is run. The system is then scanned for any connected and active ISP3 or UPP programmer. If a programmer is found it is displayed here:

| | |
|----------------------------|-------------|
| serial number | ----- |
| UPP-1 Prog found | |
| UPP-1 Prog | Rev: 061211 |
| Project: do program EEprom | |

Project Download



Projects can be downloaded to the UPP1's SD card as described above in the AVRProg section above.

Programming using the SD card



PackProg is also able to program the AVR using one of the projects which are stored on the SD card of the UPP. This is described in detail in the AVRProg section above.

Command line parameters

In principle with all calls of PackProg switches can be appended. These are:

- p Automatic programming start
- v Verify target
- f Verify flash only
- u Check programmer
- t Enable Telnet server
- c Program exit
- s No visual error messages are generated

Instead the errors are written into the file 'ISP_UPP.err' in the program directory.

'**Filename**' automatically opens and loads the PackFile defined with 'Filename'.

The order of the switches in the command line doesn't matter. The switches must be separated by spaces. A switch must not contain spaces.

Example: `C:\pppp\PackProg.exe 'abc.pac' -d12345678 -p -c`

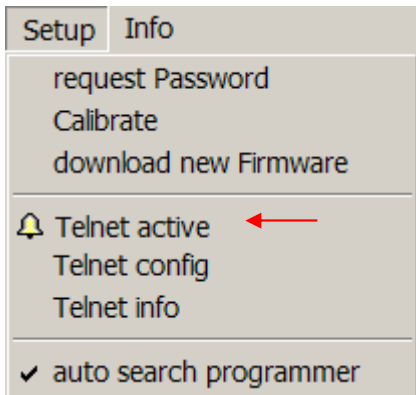
Return Codes

These return codes can be used by a batch file to control its flow.

| | | |
|----|--------------|---|
| 0 | dsOk | Operation successful finished |
| 1 | dsPwrDown | No Target voltage |
| 2 | dsPwrErr | Target too high or too low |
| 3 | dsFalseTyp | Wrong CPU ID found |
| 4 | dsProtect | Target is protected by fuses |
| 5 | dsNotEmpty | Target is not empty after an erase |
| 6 | dsVerifyErr | Target or Programmer found a Verify error while programming |
| 7 | dsFileError | N.A. |
| 8 | dsTimeOutErr | USB driver returns a timeout |
| 9 | dsCommError | Communication problem with the Programmer |
| 10 | dsNoProg | Programmer not found |
| 11 | dsNoProj | Project not found |
| 12 | dsFwLost | Programmer returns an invalid firmware |
| 13 | dsNotfound | File, erg. Hexfile, was not found |
| 14 | dsCalReq | Programmer returns a lost or illegal calibration |

Telnet Interface

There is an easy to use remote control interface in **PackProg**. To simplify this remote control the Telnet protocol is used. So other applications and also other PCs can remotely control a programmer via Telnet command strings.



This menu item enables and disables the PackProg Telnet Server.

The active Telnet Server is displayed in the main window:

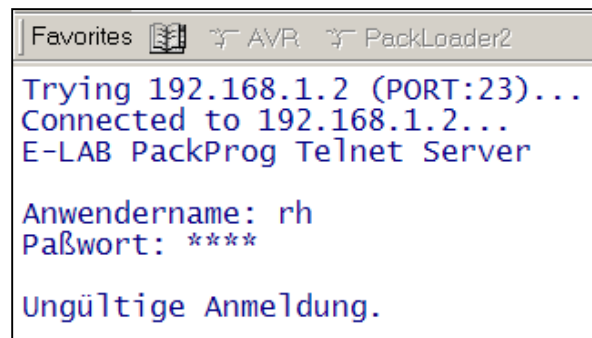


Telnet config

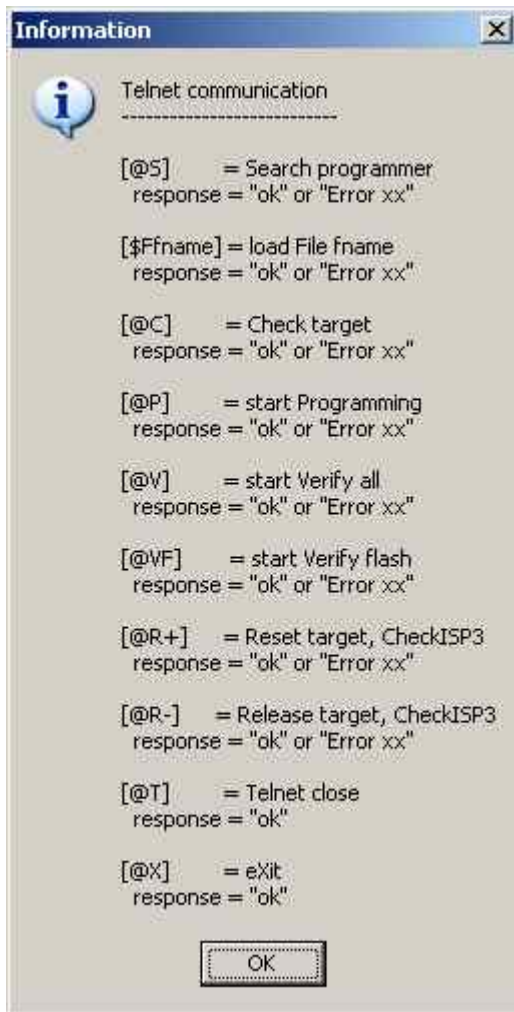
The Telnet Server needs an USER ID and a password. If a Telnet Client (another application) connects to the Server this Client must provide the correct ID and password. This disables unauthorized Telnet Clients so they can't take control of the Server or influence it.



An invalid request made with a Telnet tool looks like this:



The menu item **Telnet info** shows the Telnet command list.



Possible error messages are

1. PowerDown Target has no power
2. Voltage Target voltage too high or too low
3. 'FalseTyp Target has the wrong ID or ID not readable
4. Protected With a Verify operation
5. notEmpty With a programming cycle
6. Verify Verify Error while programming
7. File File not found, command \$F
8. TimeOut Communication timeout
9. Comm Communication problem
10. noProg Programmer not found
11. noProj No project selected
12. Firmware Programmer lost Firmware

Stand alone mode

The UPP1-X types have a buildin Li-Po battery with 700mAh (option 1500mAh).



The battery must be charged with a powersupply connected to the USB connector.

In stand-alone the device must be started with a short click onto the blue RESET button **3**. Switching off (powerdown) must be done with pressing the ERA and PROG keys at the same time. Also the device automatically shuts down after 10min inactivity.

The UPP1 has two buttons, *era/clear* and *prog/continue*. If any key is pressed an SD file check is done before any other operation. If an error occurs e.g. missing the flash card, an error peep can be heard and the LED flashes the error message 7 (SD error).

If there are **more than one project** stored in the SD the first (oldest) is always used. There is no way to select another one in stand-alone mode except if the programmer has the option **XS** built in.

With this option the up-down keys of the unit must be used to select one out of maximum of 10 stored projects. If the selected file number doesn't exist a warning signal sounds.

prog button



Pressing *prog/continue* now starts a programming cycle. If the target CPU is connected and is powered as expected, the LED flashes until the process has finished or an error occurred.

If the programming was successful the device enters its basic state, which is shown by a continuously lit LED.

If an error occurred (error code on the LED) this error must be cleared by the *era/clear* button and the device again enters its idle state.

In case of an error there are two special cases:

1. A wrong device-ID was read out of the target CPU (3 x flash). Because this ID is stored in EEprom cells in the AVR CPU and this content can be lost under some circumstances, this error (wrong ID) can be ignored and the programming can be continued by an additional click to the *prog/continue* button.
2. If the project setup expects that the target system will be powered by the programmer itself and the error "Target Power Down" (1 x flash) appears, this error can only be removed if the power supply is connected to the programmer and the target is then supplied by the programmer.
3. Furthermore (in this case) the target voltage is checked for the expected value. A voltage deviation of more then +/-0.2Volt also results in an error (2 x flash).

era button

This button mainly serves to reset an error condition as described above. Secondly a complete chip erase can be done with the target CPU.

Error blink codes

While erasing or programming there can be several errors, which are displayed by a count of flashes of the LED. Each error is preceded by an error beep:

- 1x flash = Device Power down
- 2x flash = Voltage error
- 3x flash = Wrong device/signature
- 4x flash = Device protected
- 5x flash = Device not empty
- 6x flash = Verify error
- 7x Blink = SD error, caused by missing SD card, faulty file, file not present etc.

Option XS

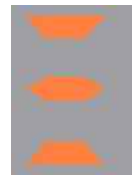
The UPP1 with option XS provides two additional keys and a 14seg LED display. After power-on a LED test appears on the display. Then the actual battery state is shown.



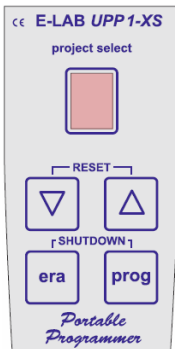
Batt empty
or nearly empty



Batt half full



Batt full



The up/down keys serve for select of one out of 10 projects on the SD card. After the selection of the desired project the programming can be started. The handling of the ERA and PROG keys is identical like the description above.

Errors are raised by a sound and also optical.

Disk and file-errors are displayed with an **E** and **D**, where the E and D are alternatingly displayed.

Programming errors are displayed with an **E** and an **errornumber**, where the E and the number are alternatingly displayed.

Programming errors can be:

- 1 = Device Power down
- 2 = Voltage error
- 3 = Wrong device/signature
- 4 = Device protected
- 5 = Device not empty
- 6 = Verify error
- 7 = MMC error

Additional display of battery state

it is possible to show the battery state in greater detail during normal operation by wrapping around below 0 pressing the down arrow key. It appears an alternating **B** (battery) and a number **0...9** or **F**. 0...9 corresponds to 0...90% full and F means full (100%). With the ERA button this info becomes removed.

File name display

The name of the selected project (file name) can be displayed by constantly pressing the up-button in a marquee mode.

Error text display

In case of an error the error text can be displayed by constantly pressing the up-button in a marquee mode.

Shutdown

By pressing the ERA and the PROG button at the same time the device becomes completely shut down.

DLL REMOTE CONTROL

DLL for use with the E-LAB UPP1-X and UPP1-P programmers Production Programming System for Atmel AVR CPUs

Up to 16 simultaneous programmings

Controlled through a WIN32 DLL

Master control program supplied by the user

One USB port used for each programmer

By the use of E-LAB UPP1-X or UPP1-P Programmer a flexible and fast InCircuit programming can be managed

Projects can be packed or encrypted on a PC.

The generated projects, up to 16, must be stored onto a SD flashcard.

Simple program change in the production by exchanging the flashcard or by selecting another project stored on the card. (up to 16)

File/project administration also through the DLL with the support functions:

List all Files, Delete File, Download File, Check File.

Support for serial number or MAC address patching

Support for direct patching of files or targets

EXTERNAL HARDWARE

Protection of the programmer against external short-circuits and over-voltage

All newer programmer types are protected against continuous short circuit. The overvoltage protection must be somewhat limited by such a device and is only allowable for a very short time and low voltages.

The protection consists of a resistor-zener diode combination for each control line. The resistor is **220 Ohm** and is connected between the 10 pin ribbon cable header and the internal AVR CPU. The diodes are a high speed protection diode array of 6 Volt types. Each diode is connected to the junction between the resistor and the CPU pin, the other side is connected to ground. Because of this wiring there can be problems with programming parts if the target system loads the control lines with low ohm resistance. Also dynamic loads caused by capacitors can lead to problems.

So it's a good idea to design system with less or no loads on the programming lines. Resistance below 2k Ohm and capacitors larger than 100pF should be avoided in conjunction with the programming lines. This is true for all programming modes.

In the PDI mode capacitors and pull-up resistors at the two programming lines should be avoided.

Battery Operation

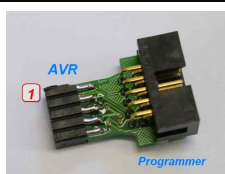
The programmer contains a rechargeable battery pack with 3,7V/700mAh. When the device is portable operated and programming a target without an own power supply or the target's power supply is too weak to also supply the programmer this battery is used. An internal battery with 3,7V/1500mAh is available as an option.

An internal step-up regulator provides the selected target voltage.

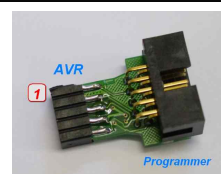
The internal battery is recharged whenever the programmer is connected to a powered USB port. Contained in the USB1-X package is also a mini AC to USB power supply as well as a car converter (for cigarette lighter 12V to 5V USB).

Miscellaneous Adaptors

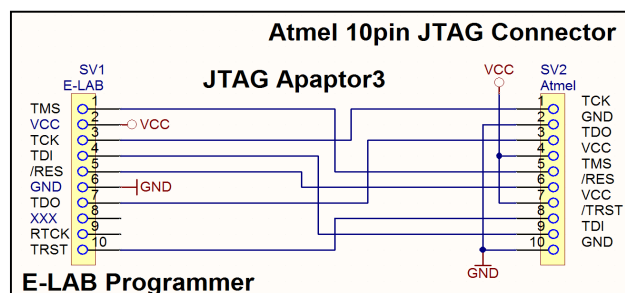
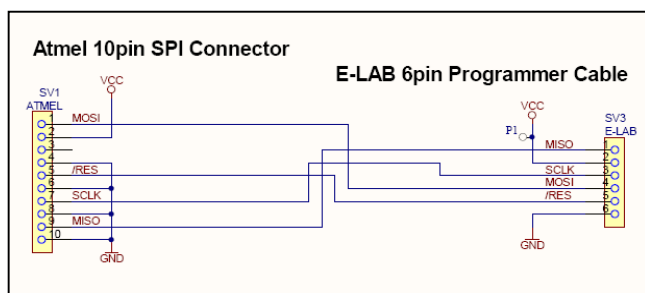
These adapters are useful if your target board has connectors with Atmel standard pinouts.



As an option an adaptor is available from E-LAB
6 pin SPI to Atmel 10 pin SPI #2125



As an option an adaptor is available from E-LAB
10pin JTAG to Atmel 10pin JTAG



TARGET POWER SUPPLY

All UPP programmers have the feature to supply the target CPU with a selectable voltage/current.

Because a USB-port of the PC or HUB also can supply at least 100mA/5Volts this can be used as an alternative to a separate power supply. Please note that the official (nominal) 5V in many cases is not 5V but maybe 4.8V or less for example. Furthermore the device internal voltage regulator also has a drop out voltage of approximately 0.2Volt. To reach a 5V output the programmer has a built-in step-up regulator so the $\geq 5V$ are always achievable.

In addition one must know that most USB Hubs can supply max. 100mA. The programmer internal current consumption can be up to 20mA. So maximum achievable current to the target can be approximately 80mA. Here an external power supply maybe necessary.



As an option an external power supply with 5 Volt can be connected to the USB jack. This supply is included. In stand-alone the internal battery (4V/700mAh) is used to unburden the target from supplying the programmer. If the programmer is directly connected to the PC the internal power supply can deliver up to 300mA. As an option there is an internal battery with 1500mAh.

01 20

03 40

05 60

07 80

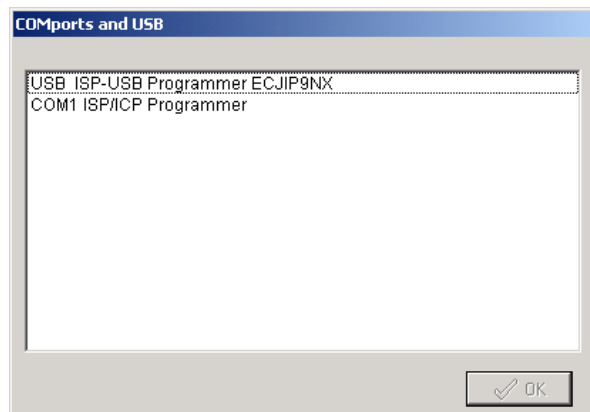
09100

The programming connector of the UPP1-X device also provides a pin for an external power supply. It can be connected to a power supply which can supply 5..9V/DC up to 500mA.

Pin6 Ground

Pin10 can be used as the external positive supply input.

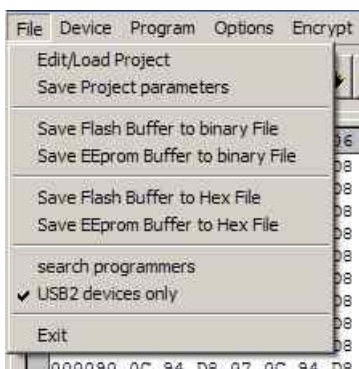
MULTIPLE PROGRAMMERS



If the PC control program *AVRprog.exe* or *PackProg.exe* finds several USB programmers or at least one serial and one USB type are present, the programmer must be selected from this dialog.

Basically only one serial type is searched. If one is found, the search for these types is aborted.

Regardless of how many programmers have been found only one can be used for working at a time. Gang programming is not possible.



If additional programmers are connected or disconnected it is possible to detect the currently connected programmers with the menu item *File/search programmers*.

USB DRIVERS

Windows compatibility

For a reliable working with a USB Programmer and a Windows system at least Win98SE is necessary. Win95, NT3/4 and standard Win98 don't support USB. Also with Win98SE one must be careful. Not all versions can handle USB in a reliable way. Because of this E-LAB can not guarantee the functionality of the USB programmer with Win98SE. So for the USB versions only XP, Vista or Windows7 should be used. Basically hard and/or software development systems should only be operated with XP or newer. These systems are very stable now and don't show the previous problems of restricted resources and handle count like their predecessors.

Driver locations

E-LAB programmers with USB-interface (like all other non-standard USB devices) need a special USB driver. This driver is included in the installation.

All USB-2 types have a common special driver set which resides in a separate sub directory of the installation directory:

`..\\USB2_Driver*.*` driver set for all **ISP3-USB**, **UPP-1** and **UPP-2** types

In the AVRco compiler installation these drivers can be found in their subdirectories below the directory
 `..\\AVRco\\.`

Driver installation

The necessary installation of the driver onto a PC is relative simple and without any problems. With the installation of the programmer package the necessary USB drivers are automatically installed.

But one must proceed in fixed procedure.

1. Disconnect any programmer devices from the PC
2. Startup the computer and wait until the system is ready for working.
3. The execute the included programmer install program "*ISP_ICPinst.exe*". (**not** for AVRco)
4. The following Windows dialogs concerning the USB driver must be answered with yes
5. Plug the USB programmer into a free USB-port of the PC.
6. Windows now recognizes a new unknown USB device and registers it.
7. The programmer software is now ready to work.

Installing unsigned drivers

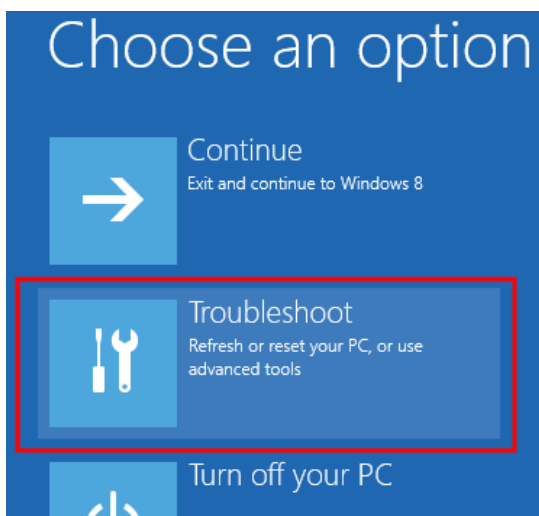
In order to force the installation of unsigned (not Microsoft certified) drivers under Windows 8/64 this must be enabled before the start-up of the Windows system with “cmd shutdown /r /o”. Windows now executes a restart.

This concerns the E-LAB AVRprog USB driver and also the AVRco USB driver.

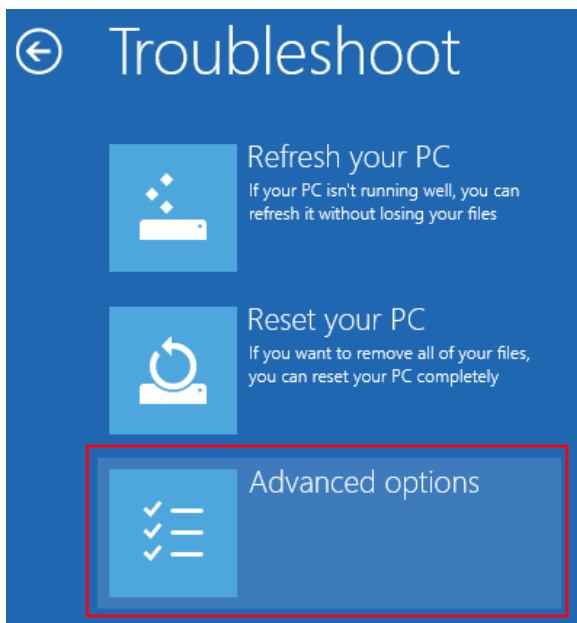
Step 1:

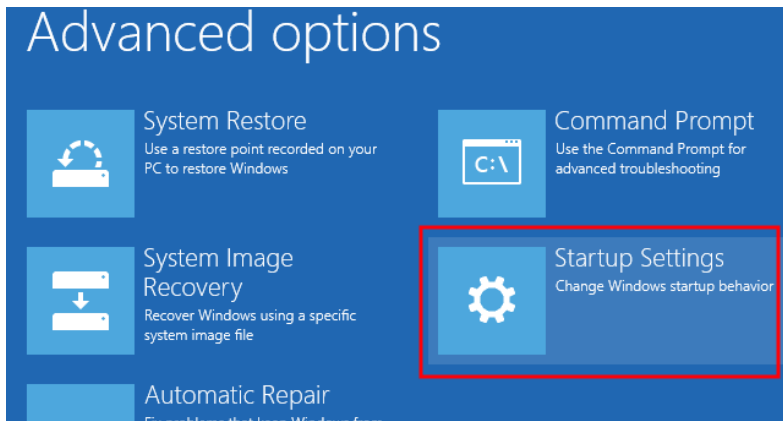
If Windows 8 has started then the Boot-Options-Menu must be started. To do this, on the Windows 8 desktop the button combination of **Win+R** must be pressed to open the “Execute”-dialog. In this dialog the following command must be typed in to start the “Options”-menu at boot time:

Step 2: In the „Options“-menu click onto “Troubleshoot”.

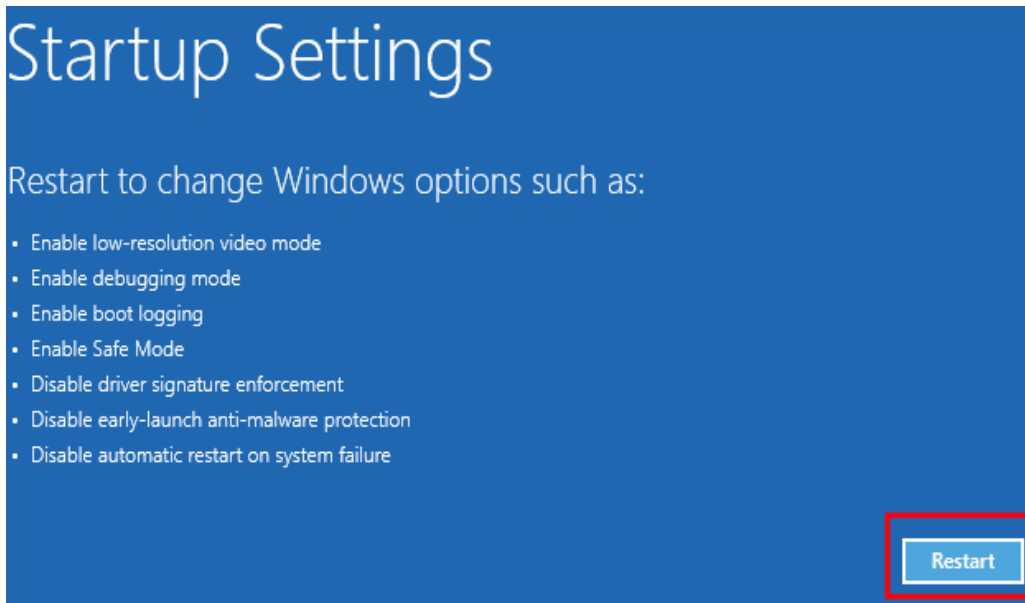


Step3: click onto “advanced options”.

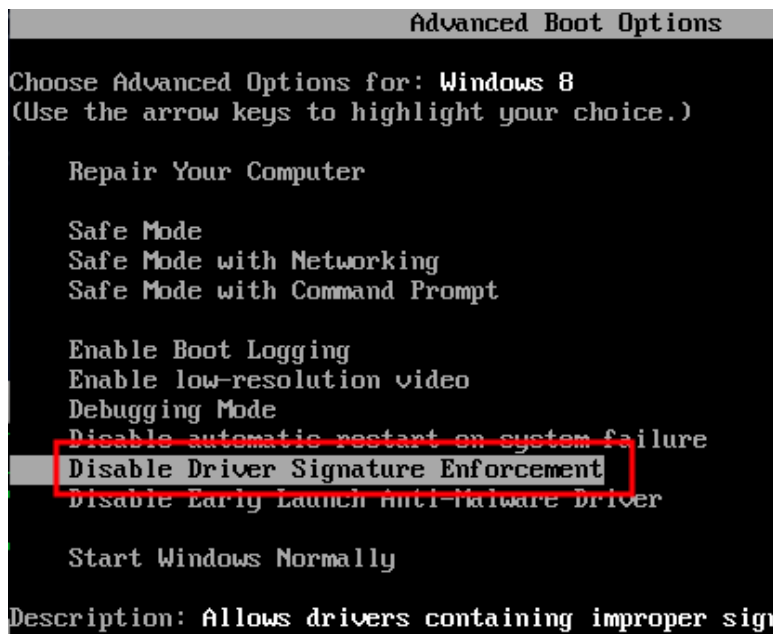




Step4: click onto “Startup Settings”.



Step5: click onto “Restart”.



Step6: Choose **Disable Driver Signature Enforcement** and Enter key to start Windows

Now the AVRco or AVRprog driver can be installed without any error message.

Please note that the included EXEs must be started in WIN7 mode.

Installing multiple programmers

The E-LAB programmers are individually registered by Windows. If the very first USB programmer becomes registered now Windows at least knows this version/type of (ISP-3 or UPP) of programmer.



Further new devices normally show this dialog at the first plug-in and then automatically become registered. But with some system environment it's also possible that Windows requests the location/path where the driver is located.

De-installing programmer drivers

Basically it makes no sense to de-install drivers for the E-LAB USB programmers because these drivers are only loaded temporarily if a device gets connected. So the system is not loaded if no device is connected. But if it becomes necessary to de-install USB drivers this must not be done manually.

Only this driver where the related hardware is connected, can be de-installed. To start de-installation at first open the device manager of Windows. With the help of this tool the driver can be de-installed. Other Windows utilities offer a so called deactivating of drivers which has nothing to do with de-installation.

FIRMWARE UPDATE

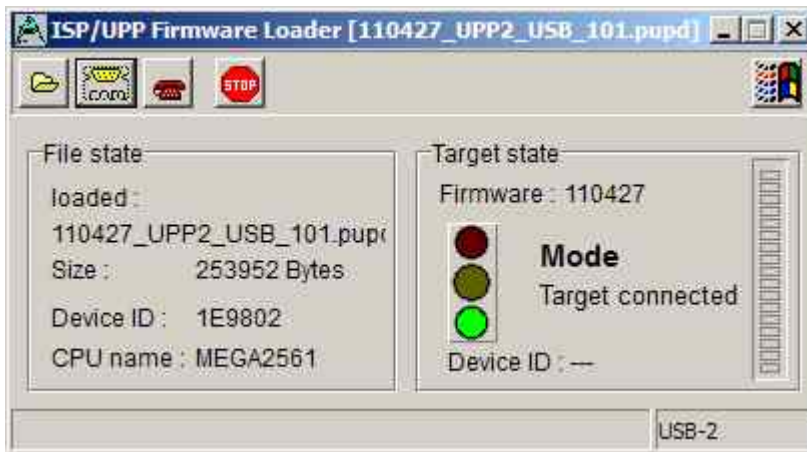
The latest E-LAB programmers support downloading of new firmware into the programmer. You can determine whether your device has this feature if you open the **Device Status** dialog. If this device is updateable then in the line Update a number appears which shows the current firmware date, which is also a part of the update file name.

With a firmware update of a programmer the entire program (Flash) is erased except a small partition which is called "boot sector". New firmware for the programmers can be downloaded from www.e-lab.de. These programs cannot be executed on the PC, but must be loaded into the programmer.

USB version file: **yymmdd_UPP1_X.pupd** or from WEB in file: **UPP1_Xupdate.zip**

This must be done with the menu item **Options/Download new Firmware**. Please note that all update files must be placed into the folder **ISP_Updates** below the programmer's home directory. Otherwise they will not found. The menu item opens the dialog described below.

ISP/UPP Firmware Loader Dialog



The group "File state" contains the state of the loaded file: filename, file size and info about the expected target CPU.

The group "Target state" contains the state of the programmer.

If the traffic light shows a green the downloading can be started.

The vertical bar shows the download and programming progress.



The "Open" button opens a dialog which shows all possible firmware update files. The filenames start with the date of creation and can be easily located. A double click loads the selected file into the download buffer.



A click to the Com port button then connects the programmer's download section to the PC's downloader. The downloader checks the file against the info in the programmer. If the file can't be accepted an error messages is raised.



If the file matches the programmer and a connection is established the download can be started with a click to the phone button. This process can be observed in the progress bar at the dialog's right side.



If the downloads "hangs" a click to the stop button aborts the operation. On problems an increase or decrease of the external UPP voltage can help.

An incomplete download/programming normally is detected at reset or power up of the programmer. If so, the programmer immediately enters the download state and awaits a new download.

It is also possible to **force a firmware download**.

The UPP1-X has a RESET button on the rear front of the case.



1. USB and power supply jack
2. Micro-SD card slot
3. RESET button

This RESET button has two functions:

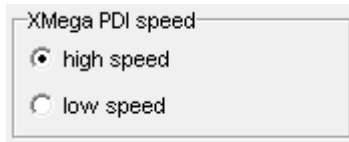
1. a short click to this button executes a **Hardware Reset**.
2. a longer click to this button (>1sec) also executes a **Hardware Reset** and in addition it forces the device to enter the **forced Download state** like described above.

ADDENDUM

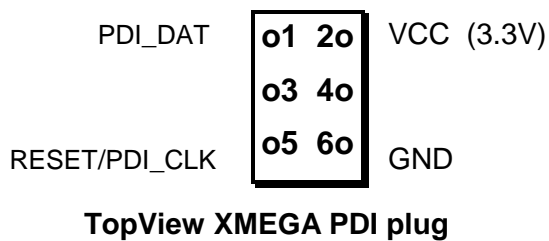
xMega

Attention:

The RESET line on the target must never be loaded with capacitors because the PDI-CLK is fed here with > 1MHz. A jumper which disconnects RESET condenser at programming time can help.



With older XMegas, for example Xmega128A1, or long programming cables some errors maybe raised. Here Atmel recommends that the clock rate of the PDI interface should be reduced. This can be done in “Options/Programmer options”.



TINY4-5-9-10-20

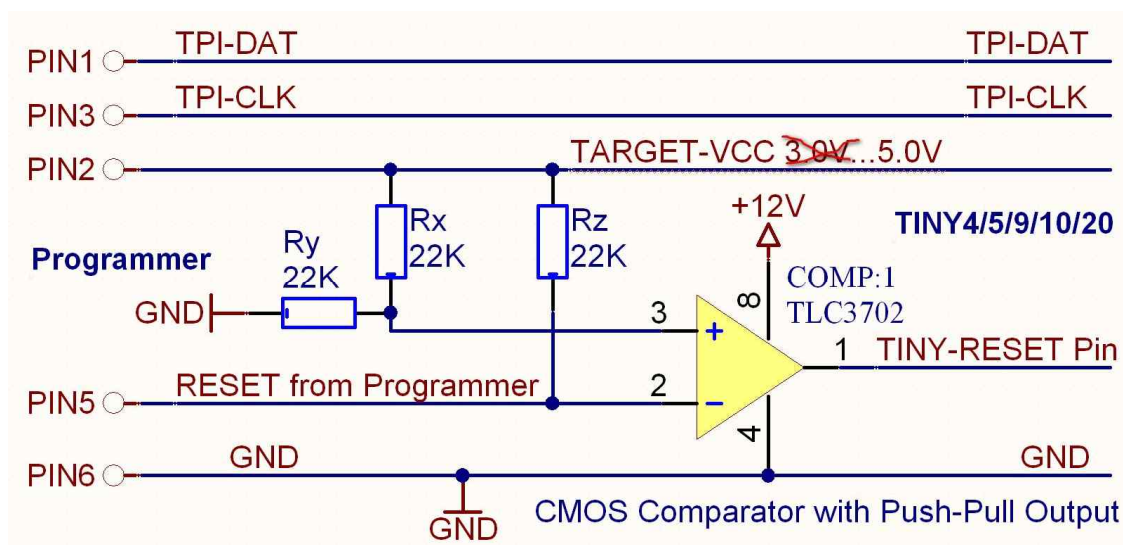
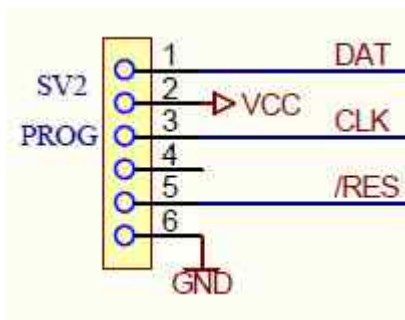
These TINYS must be programmed with 5V through 3 pins.

The programming mode is called TPI. Because the same programming plug is also used by the SPI mode the plug on the target system differs from the Atmel plug. The load on the TPI_DAT pin must not be lower than 80kOhm.

This is the connection of the programmer cable to a TPI Tiny.

Because the RESET pin of these TINYS can be disabled by a fuse a normal re-programming of these devices is not possible. Then the high-voltage programming mode must be used. Fortunately the programming scheme is the same but only the RESET line must be set to 12V.

Here is a simple schematic where the RESET output of the programmer is used to switch a 12V source:

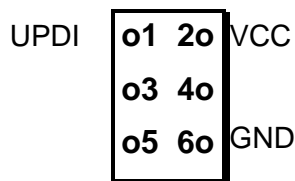


UPDI Tiny and UPDI Mega

Basics

The latest Atmel programming Interface is a one-wire version which only uses one pin of the Chipst. This interface is very fast but needs a special algorithmus. This is supported in the X-versions of the E-LAB programmers.

The pin-out of the programmer plug is showed below. Please note that the UPDI line must not be connected to heavy loads.



TopView TINY UPDI plug

A certain disadvantage of this single-wire systems is that the UPDI pin of the chips can be changed to common IO by fuses. If so programmed the chip can't be re-programmed in a standard way! But UPDI Megas are not concerned.

In this case a special 12V pulse at the UPDI pin can remove this fuse setting. But therefore a special algorithmus is necessary. This is also supported by these programmers. Because all must fit together, timings and voltage actions, a special external hardware must be present.

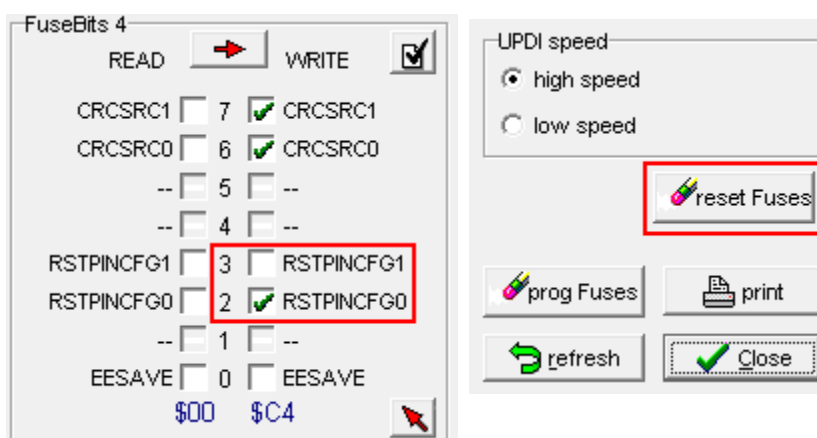
This HVP adapter can be ordered from E-LAB. It doesn't need an extra supply.



If the 12V (HVP) reset is necessary then with the help of the connected HVP adapter the concerned fuses can be reset.

But please note that parts/devices connected to the UPDI of the chips must be 12V tolerant. Also heavy loads must be avoided.

With the dialog below in the tool AVRprog.exe this reset can be executed.



This button is only visible when the UPDI mode is active and the HVP adapter was found

Chipcon

Basics

The UPP1-X firmware also supports the In-Circuit programmable ZigBee Chip family CC1110, CC2510 and CC2430 from TI-Chipcon.

With the creation of a new project for the CC2430 in addition to an existing Flash hexfile also the correct CPU must be selected (CC2430-F32, CC2430-F64 or CC2430-F128).

The generated supply voltage for the target can be set between 2.7V and 3.6V if the programmer has to supply the target. A valid CPU clock must be selected, either 16MHz or 32MHz.

There are no fuse bits but a lock bit block. The meaning of these lock bits can be found in the datasheet of the CC2430.

IEEE Address



Because the CC2430 is a ZigBee device the controller must have a fixed and unique address (IEEE) similar to a MAC address with Ethernet. Chipcon defined that this address (8 bytes) must reside in the upper most and last 8 bytes of the flash memory. With the F128 this is \$1FFF8..\$1FFFF.

The order is MSB at the lower address and LSB on the upper most address.

The programming software supports the address handling so that this address can be preset and becomes auto incremented after each programming cycle.

With **Preserve Serial Number** the number in the Target is read back and used for the next programming cycle, provided that the chip is not protected

With **LSB Serial Number Order** the serial number will be stored beginning with the LSB and ascending in the Flash, otherwise beginning with the MSB.

These options can be set and edited in the dialog **Options/Target Options**

Connectors

The included 6 wire ribbon cable must be used. The connections of the receptacle/header on the target board must be done in a way like the schematic below shows it..

| | | |
|------------|-------|-------------|
| N.C | 01 20 | VCC |
| (P2_2) CLK | 03 40 | DATA (P2_1) |
| /RESET | 05 60 | GND |

TopView header on the Target

Attention!

The PIN2 must be connected with the positive Target supply (VCC). PIN6 must be connected to the ground of the Target. Connect PIN5 to the RESET PIN, PIN3 to the Debug Clock (P2_2) and PIN4 to the DATA PIN (P2_1) of the CC2430.

Verify

While programming the target which is done with 1kB blocks, a verify of the current block is automatically executed after finishing the programming of this block. In case of a verify error the operation becomes aborted and an appropriate message rises.

Chip internal ID



This ID number is read out of the current connected chip and is displayed here. The first byte is always zero and has no meaning. The second one represents the CPU revision (\$01) and the third byte shows the Chip-ID, \$85 for the CC2430 family.

Attention

Please note that a wrong chip-ID is always a result of a defect programmer, cable or board connection. Of course a defect chip can also be the reason for it.

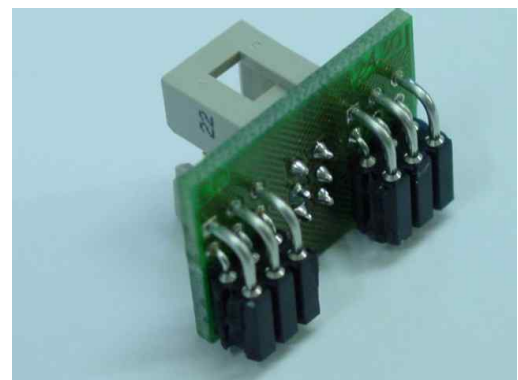
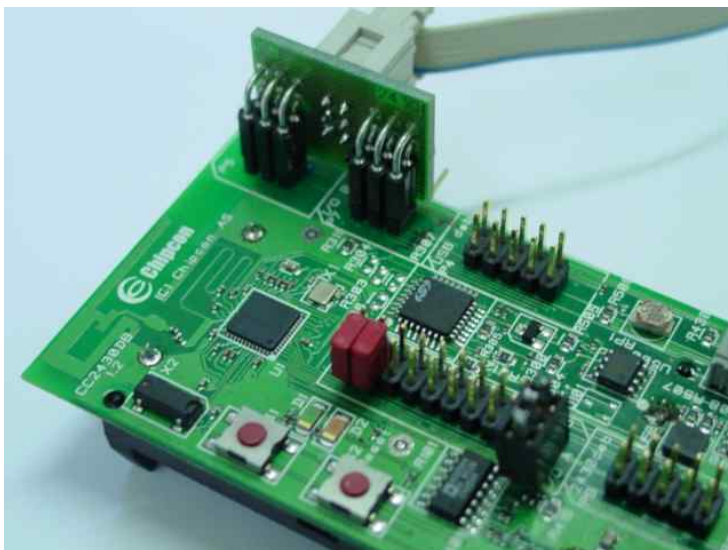
A false chip ID should never be ignored. But the reasons for it should be carefully researched.

Chipcon Evaluation Boards

CC2430DB

This board has two 6pin headers which provide an access for external programmers. But the UPP programmer can not directly plugged into the board. In order support programming of this board by the UPP there is an optional accessory for the UPP which consists of a small adaptor board. This is not included but must be ordered separately. (ord #2109)

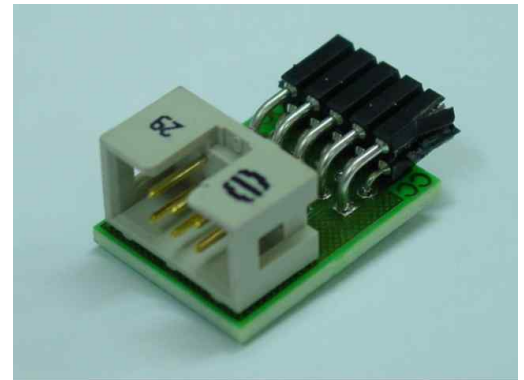
The adaptor must be connected to the evaluation board like the pictures below show.



Chipcon Evaluation Board SOC-BB

this board has one 10pin header which provides an access for external programmers. But the UPP programmer can not directly plugged into the board. In order support programming of this board by the UPP there is an optional accessory for the UPP which consists of the small board **SOC-BB Adaptor**. This is not included but must be ordered separately. (ord #2123)

The adaptor must be connected to the evaluation board like the pictures below show.



AT89LPxx

Basics

The UPP1-X firmware also supports the in-Circuit programmable Atmel Chip family AT89LP2052 und 4052.

The difference to the AVR's and 89Sxx types consists of programming connections of the CPU with in-circuit programming.

Connectors

The included 10 wire ribbon cable must be used. The connections of the receptacle/header on the target board must be done in a way like the schematic below shows it..

| | | |
|--------|--------------|------|
| MISO | 01 20 | VCC |
| SCLK | 03 40 | MOSI |
| /RESET | 05 60 | GND |
| SS | 07 80 | N.C. |
| N.C. | 09100 | N.C. |

TopView header on the Target

Attention!

The PIN2 must be connected with the positive Target supply (VCC). PIN6 must be connected to the ground of the Target. Connect PIN5 to the RESET pin, PIN3 to the SCLK, PIN4 to the MOSI pin, PIN1 to the MISO pin and PIN7 to the SS pin of the target..

Serial Flash (SPI-Flash)

Basics

The UPP1-P/R firmware also supports the in-Circuit programmable SPI-Flash chips of the types AT25DFxxx, S25FLxxx and SST25VFxxx.

Connectors

The included 6pin cable must be used. The connections of the receptacle/header on the target board must be done in a way like the schematic below shows it..

| | | |
|------------|--------------|------------|
| MISO/SO | 01 20 | VCC (3.3V) |
| SCLK | 03 40 | MOSI/SI |
| /CS-SELECT | 05 60 | GND |

TopView header on the Target

OTHER E-LAB PROGRAMMERS



ISP3-X

The ISP3-X programmer always needs a PC and hence is not portable.

It is the cheapest device of the E-LAB programmer family.

The connection to the PC is via USB2.



UPP1-XS

The programmer UPP1-X can be operated by the PC or independently of the PC (portable). In the portable mode upto 10 projects are stored on its microSD flash card.

The project selection is done with the Up/Down buttons with the help of the LED display. The selected project name can be displayed on the display in a marquee mode.

The device is internally Li-Po battery powered in portable mode.

The connection to the PC is via USB2.



UPP2-X

The flagship of the portable E-LAB programmers.

Up to 63 projects can be stored and selected on its microSD flash card.

The device is internally Li-Po battery powered in portable mode.

The connection to the PC is via USB2.

Also best suited for field service and small series production.

UPP1-P and UPP1-PR

The productions implementation of the UPP1-X.

In top hat (DIN) rail case. Can be controlled by a PC with USB or Remote controlled by our DLL version **P** or with wire control version **PR**





Notes





Notes





Notes



©1998-2018 ***E-LAB Computers***

Grombacherstr. 27
D74906 Bad Rappenau

Tel. 07268/9124-0
Fax. 07268/9124-24

Internet: www.e-lab.de
e-mail: info@e-lab.de