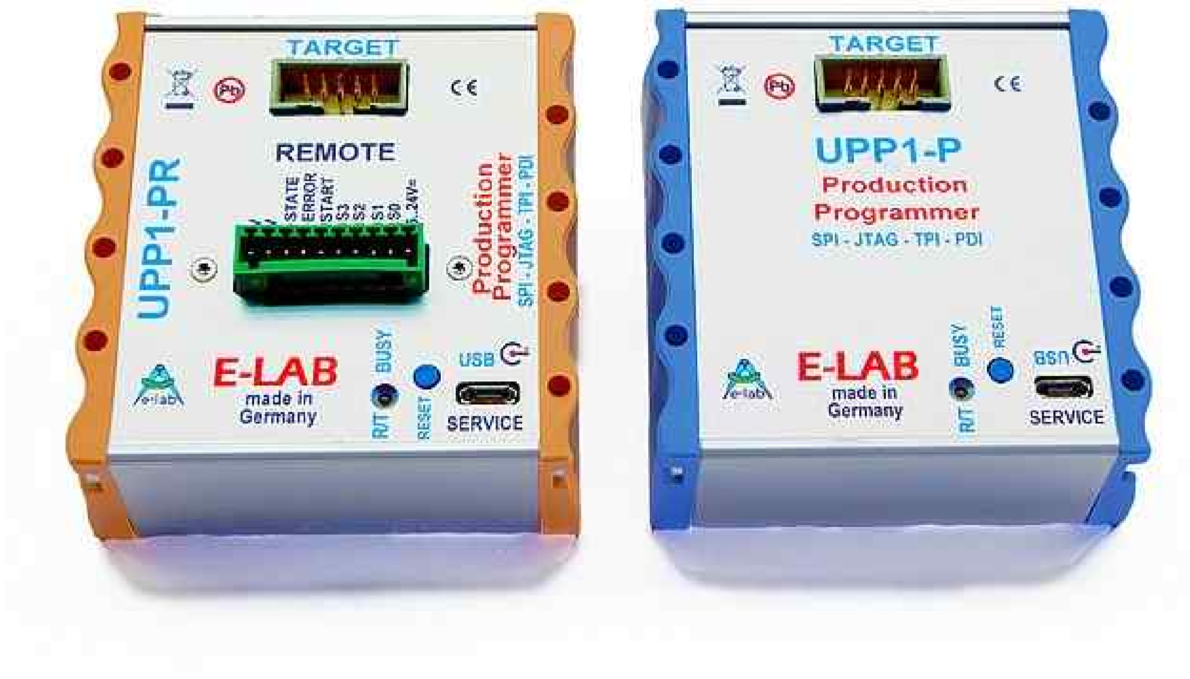
User Manual E-LAB UPP1-P/R Programmer



UPP1-P/R + AES Encryption

Production Programmer

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

2018-06-22

TABLE OF CONTENTS

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-P/R** 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 PC interface (USB) 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..80mA if powered from the target system
- Easy and extensive software. A project can be stored on the build-in micro-SD card.
- Software runs under Windows XP, Vista and Windows7, 8 / 32 and 64bit
- Small, lightweight and handy unit - 75x72x30mm
- Supports all **SPI, JTAG, PDI, UPDI** and **TPI** programmable AVRs
- 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
- Also processes AES encrypted project files (PAC) onboard
- Supports programming cycles limitation with AES-PAC files
- The version “P” can also be remote controlled by the E-LAB ***DLL***
- With the version ***“PR“*** the programming can be remotely controlled via control lines. Up to 16 projects on the flash card can be handled (1 of 16).
- micro-SD card included. FAT16 up to 2GB and FAT32 up to 32GB and SDHC types are supported.
- 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-P/R to ChipCon CC2430 Evaluation Board (not included)
- Option adaptor from E-LAB UPP1-P/R to ChipCon SOC_BB Evaluation Board (not included)

CONNECTIONS

XP, Vista or Windows7/8 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 pullup should not be lower than 100kOhm.

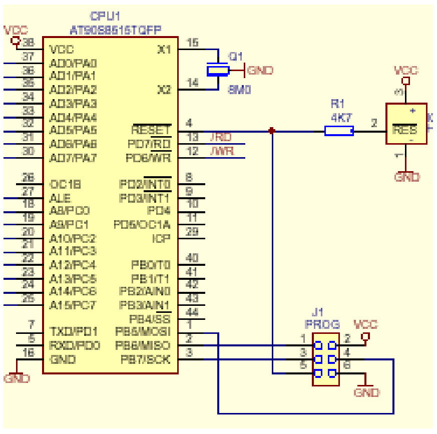
The definition of the 6-pin target plug (0.1 inch pitch male header, dual inline) conforms to a recommendation from Atmel. The **TOP VIEW** onto the connector of the Target is below:

CPU MISO (TxD)	**o1**	**2o**	VCC
CPU SCLK	**o3**	**4o**	CPU MOSI (RxD)
/RES	**o5**	**6o**	GND

Top View AVR SPI header

PDI_DAT	**o1**	**2o**	VCC (3.3V)
	o3	**4o**	
RESET/PDI_CLK	**o5**	**6o**	GND

Top View XMEGA PDI header



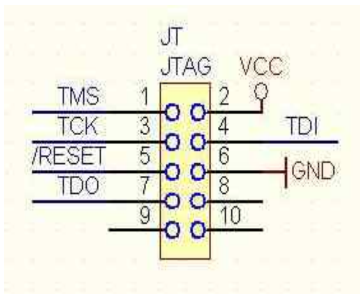
Pin 1 of the 6-pin plug of the programmer can be located by a small triangle on the front of it. A misalignment of the plug leads to malfunction and *can possibly* **DESTROY** the unit.

The working voltage of the Target CPU must be in the range of 1.8V..5.5V. If this voltage is below 3.3V the UPP must be supplied by the PC or an external power supply.

None of the 4 control lines of the device must be shorted. Although these lines are overload protected **continuous short circuits** can destroy the programmer permanently.

Only electrical 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 programmer 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 also note that the **/RESET** line should be connected to the target CPU.

Top View AVR JTAG header

Tiny 4, 5, 9, 10, 20 TPI 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 at the TPI_DAT pin must not be lower than 80kOhm.

TPI_DAT	**o1**	**2o**	VCC (5V)
TPI_CLK	**o3**	**4o**	
/RES	**o5**	**6o**	GND

Top View AVR Tiny TPI header

SOFTWARE

There are two supplied programs that can be used to control the UPP1-P/R 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.

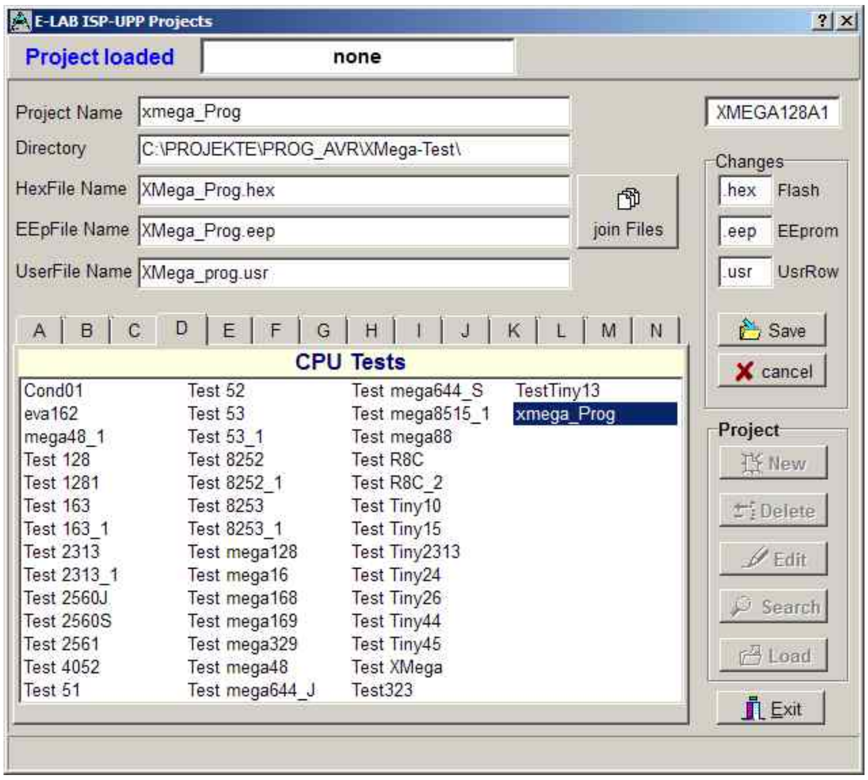
The second program is **PackProg**.exe and it can only be used for programming or verifying using a packed or encrypted file that has been previously created by AVRProg. It is the most suitable program for production use and is described later in this manual.

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 start-up.

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.


E-LAB ISP-UPP Projects

Project loaded: none

Project Name: xmega_Prog
Directory: C:\PROJEKTE\PROG_AVR\XMega-Test\
HexFile Name: XMega_Prog.hex
EEpFile Name: XMega_Prog.eep
UserFile Name: XMega_prog.usr

XMEGA128A1

Changes: .hex Flash; .eep EEprom; .usr UsrRow

CPU Tests

Cond01	Test 52	Test mega644_S	TestTiny13
eva162	Test 53	Test mega8515_1	xmega_Prog
mega48_1	Test 53_1	Test mega88	
Test 128	Test 8252	Test R8C	
Test 1281	Test 8252_1	Test R8C_2	
Test 163	Test 8253	Test Tiny10	
Test 163_1	Test 8253_1	Test Tiny15	
Test 2313	Test mega128	Test Tiny2313	
Test 2313_1	Test mega16	Test Tiny24	
Test 2560J	Test mega168	Test Tiny26	
Test 2560S	Test mega169	Test Tiny44	
Test 2561	Test mega329	Test Tiny45	
Test 4052	Test mega48	Test XMega	
Test 51	Test mega644_J	Test323	

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 extens
Store changed or new project.

<u>Commands</u>
Build a new project

Delete a project

Edit an existing project

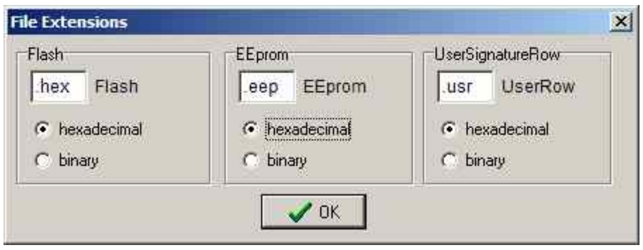
Search project on network

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 <u>must.</u> The following dialog for selecting an EEprom file can be ignored, if nothing exists. The new project must be stored by the **Save** button.


File Extensions

Flash: .hex Flash — [✓ hexadecimal] binary

EEprom: .eep EEprom — [✓ hexadecimal] binary

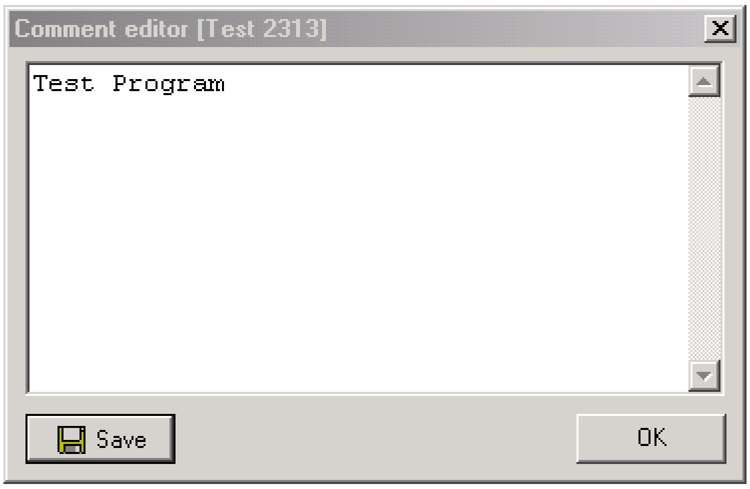
UserSignatureRow: .usr UserRow — [✓ hexadecimal] binary

OK

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 [Test 2313]

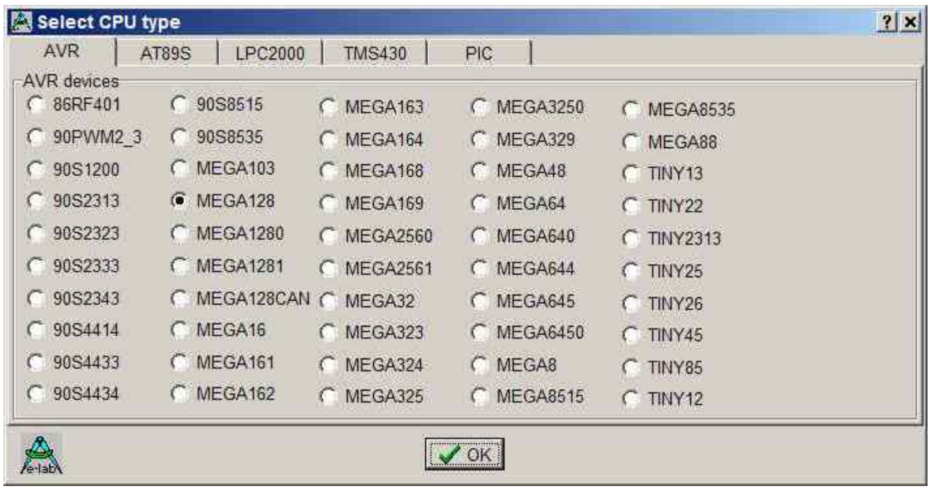
```
Test Program
```

Save　OK

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.


Select CPU type

AVR | AT89S | LPC2000 | TMS430 | PIC

AVR devices

86RF401	90S8515	MEGA163	MEGA3250	MEGA8535
90PWM2_3	90S8535	MEGA164	MEGA329	MEGA88
90S1200	MEGA103	MEGA168	MEGA48	TINY13
90S2313	[✓ MEGA128]	MEGA169	MEGA64	TINY22
90S2323	MEGA1280	MEGA2560	MEGA640	TINY2313
90S2333	MEGA1281	MEGA2561	MEGA644	TINY25
90S2343	MEGA128CAN	MEGA32	MEGA645	TINY26
90S4414	MEGA16	MEGA323	MEGA6450	TINY45
90S4433	MEGA161	MEGA324	MEGA8	TINY85
90S4434	MEGA162	MEGA325	MEGA8515	TINY12

OK

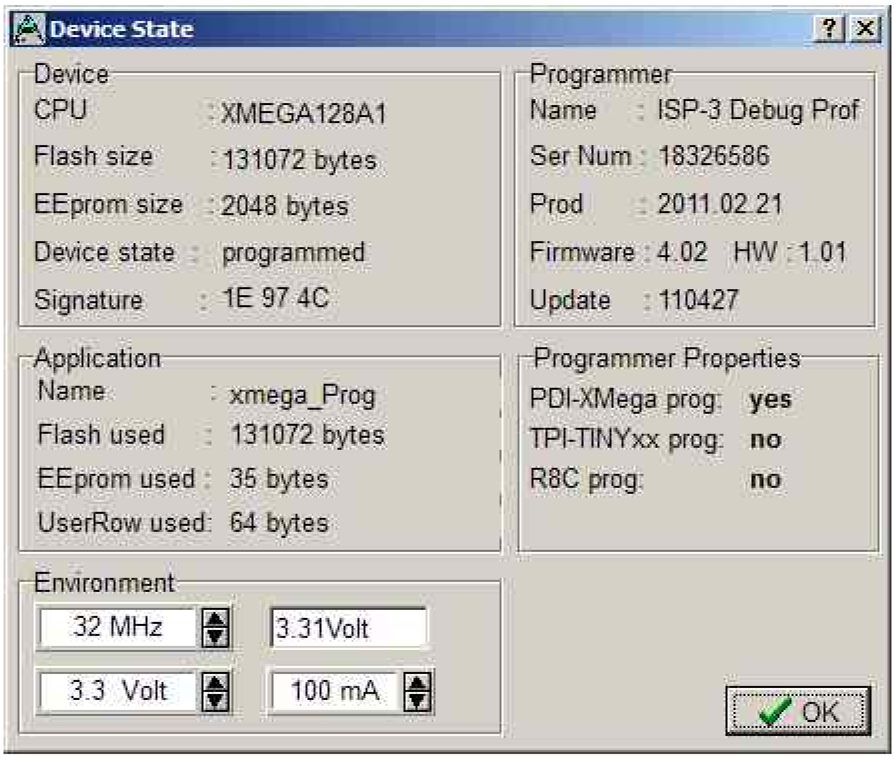
Dialog for CPU Selection

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.

The CPU types LPC, TMS and PIC are not implemented at this time.

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.



Device State

Device
CPU : XMEGA128A1
Flash size : 131072 bytes
EEprom size : 2048 bytes
Device state : programmed
Signature : 1E 97 4C

Programmer
Name : ISP-3 Debug Prof
Ser Num : 18326586
Prod : 2011.02.21
Firmware : 4.02 HW : 1.01
Update : 110427

Application
Name : xmega_Prog
Flash used : 131072 bytes
EEprom used : 35 bytes
UserRow used: 64 bytes

Programmer Properties
PDI-XMega prog: yes
TPI-TINYxx prog: no
R8C prog: no

Environment
32 MHz
3.31Volt
3.3 Volt
100 mA

OK

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-P/R programmer supplies the selected voltage to the target system (only in programming state). The current will be limited to the selected value.

With XMegas the maximum supply voltage is limited to 3.6V
With some TINYs the minimum supply voltage is limited to 5.0V

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.

USB power supply

For powering the target board the UPP1-P/R has an internal step-up voltage converter which generates a voltage of about 5.6V from the USB voltage (4..4.8V). So an external power supply is not necessary provided that the target load is less than what can be provided by the USB port or hub. See the section **Target Power Supply** below for more discussion on power supplies.

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-inifile xxx.ispe in the Explorer automatically invokes AVRprog.exe, which by itself reads the ini-file 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 of no parameter specifies the project path.
-u0 A standard Pack File will be build
-u1 An encrypted Pack File will be build
-g*ProgSerNum* 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, e.g. 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:

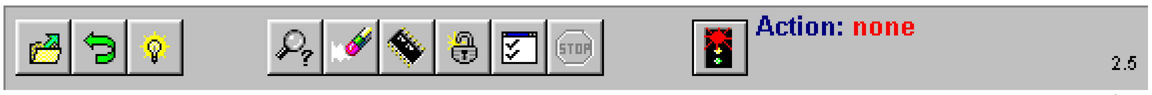
```
[Settings]
ProjExt=.isp
```

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-P/R 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.



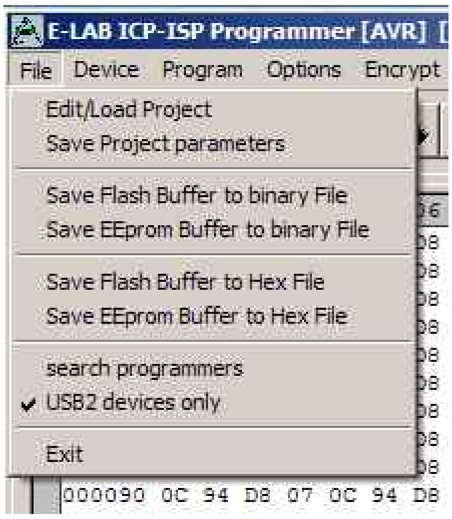
Firmware-revision
display of the actual action
release of the reset of CPU and restart
abort of the actual action
Flash and EEprom Verify
subsequent Fuse programming
Flash, EEprom and Fuses programming
Chip erase
State of the programmer and the target
flush of the Flash- and EEprom buffer
reload of the Flash- and EEprom buffer
another resp. new project load

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

Functions 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


E-LAB ICP-ISP Programmer [AVR]
File
Device
Program
Options
Encrypt
Edit/Load Project
Save Project parameters
Save Flash Buffer to binary File
Save EEprom Buffer to binary File
Save Flash Buffer to Hex File
Save EEprom Buffer to Hex File
search programmers
✓ USB2 devices only
Exit
000090 0C 94 D8 07 0C 94 D8

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 in 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 FileDialog 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 FileDialog 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.

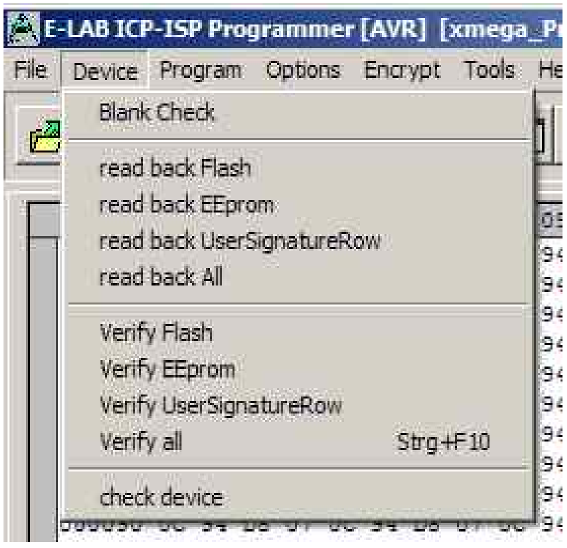
The **Project-Open** button opens the project dialog. A new project can be built or an existing one can be opened and loaded.

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.

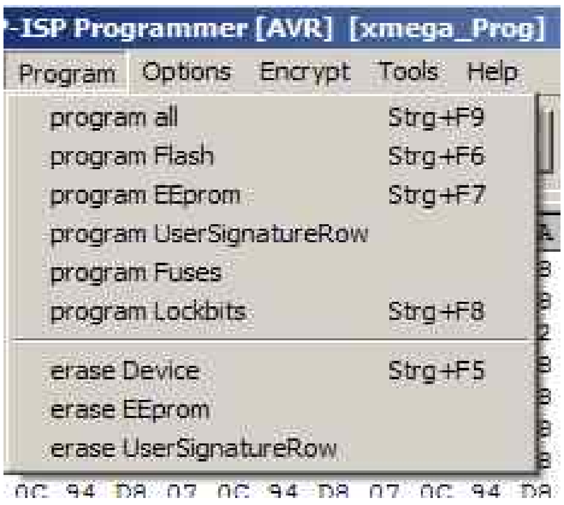
Verify UserSignatureRow if the target is not protected, the contents of the target's UserRow is compared to the UserRow-Buffer. If there is any difference an error message is raised. Only XMega.

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 UserSignaturRow 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 UserSignaturRow 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

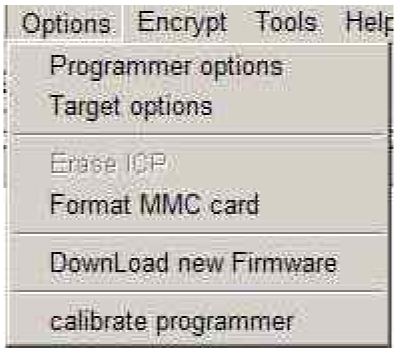
The **Verify** button starts a verify of the target with the buffers. Only possible on an unprotected device.

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 behavior of the programmer and it's additional options must be setup at least once for a project. To do this there a two dialogs: the Options Dialog and the Target Options Dialog. These are called with the menu below.



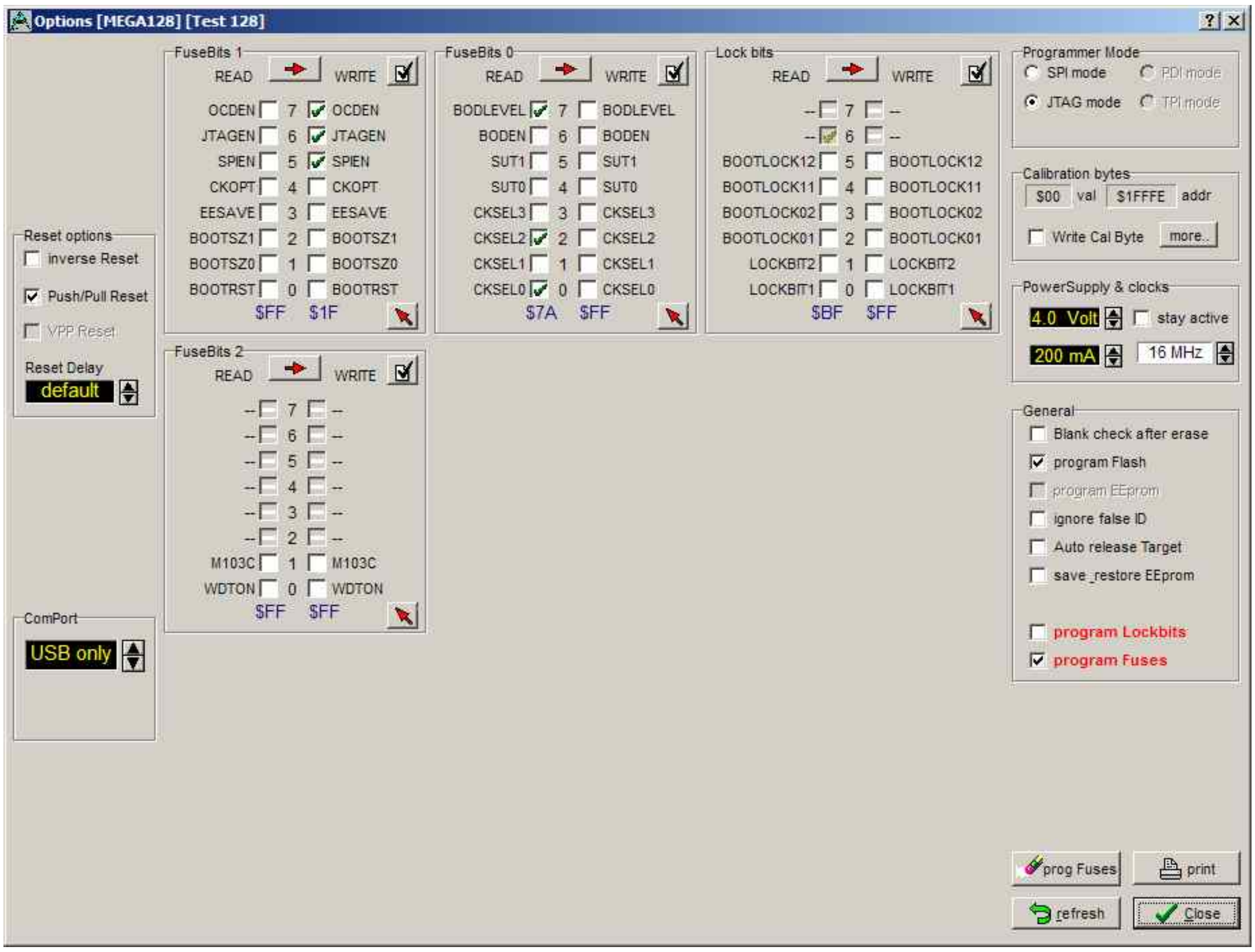
Programmer options starts the Options Dialog where the Fuse and Lock bits can/must be defined and also some other functions.
Target options starts the Target Options dialog where powerful extra functions can be enabled and setup.

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.

Calibrate programmer is not applicable for the UPP1-P/R

Programmer Options Dialog

The Options-Dialog controls the behavior 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..



Options [MEGA128] [Test 128]

FuseBits 1

READ			WRITE
OCDEN		7	✓ OCDEN
JTAGEN		6	✓ JTAGEN
SPIEN		5	✓ SPIEN
CKOPT		4	CKOPT
EESAVE		3	EESAVE
BOOTSZ1		2	BOOTSZ1
BOOTSZ0		1	BOOTSZ0
BOOTRST		0	BOOTRST
$FF			$1F

FuseBits 0

READ			WRITE
BODLEVEL	✓	7	BODLEVEL
BODEN		6	BODEN
SUT1		5	SUT1
SUT0		4	SUT0
CKSEL3		3	CKSEL3
CKSEL2	✓	2	CKSEL2
CKSEL1		1	CKSEL1
CKSEL0	✓	0	CKSEL0
$7A			$FF

Lock bits

READ			WRITE
--		7	--
--	✓	6	--
BOOTLOCK12		5	BOOTLOCK12
BOOTLOCK11		4	BOOTLOCK11
BOOTLOCK02		3	BOOTLOCK02
BOOTLOCK01		2	BOOTLOCK01
LOCKBIT2		1	LOCKBIT2
LOCKBIT1		0	LOCKBIT1
$BF			$FF

FuseBits 2

READ			WRITE
--		7	--
--		6	--
--		5	--
--		4	--
--		3	--
--		2	--
M103C		1	M103C
WDTON		0	WDTON
$FF			$FF

Reset options
inverse Reset
[✓ Push/Pull Reset]
VPP Reset
Reset Delay: default

ComPort: USB only

Programmer Mode
SPI mode · PDI mode
[✓ JTAG mode] · TPI mode

Calibration bytes
$00 val $1FFFE addr
Write Cal Byte

PowerSupply & clocks
4.0 Volt · stay active
200 mA · 16 MHz

General
Blank check after erase
[✓ program Flash]
program EEprom
ignore false ID
Auto release Target
save_restore EEprom
program Lockbits
[✓ program Fuses]

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 behavior 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 AVRs.

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.

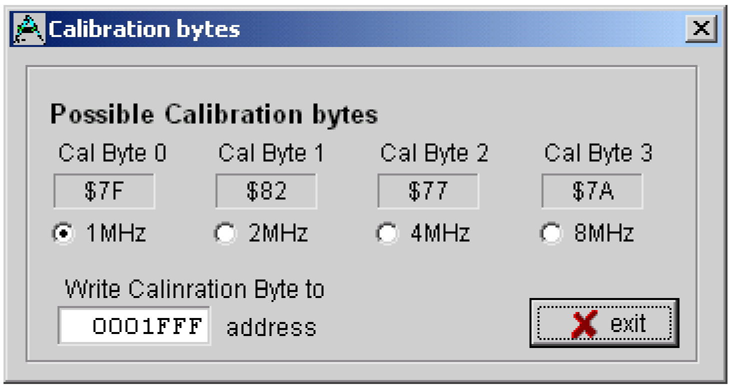
Attention: within the fusebit groups some CPUs have a **SPIEN** fuse. This fuse normally is 'don't care' with SPI-Low-Voltage programming. But there are exceptions, the Tiny12 for example. In this case the SPIEN fuse must be activated. Otherwise the chip is not accessible any more.

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 excludes 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.


Calibration bytes

Possible Calibration bytes

Cal Byte 0	Cal Byte 1	Cal Byte 2	Cal Byte 3
$7F	$82	$77	$7A
[✓ 1MHz]	2MHz	4MHz	8MHz

Write Calinration Byte to
0001FFF address

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 AVRs 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.

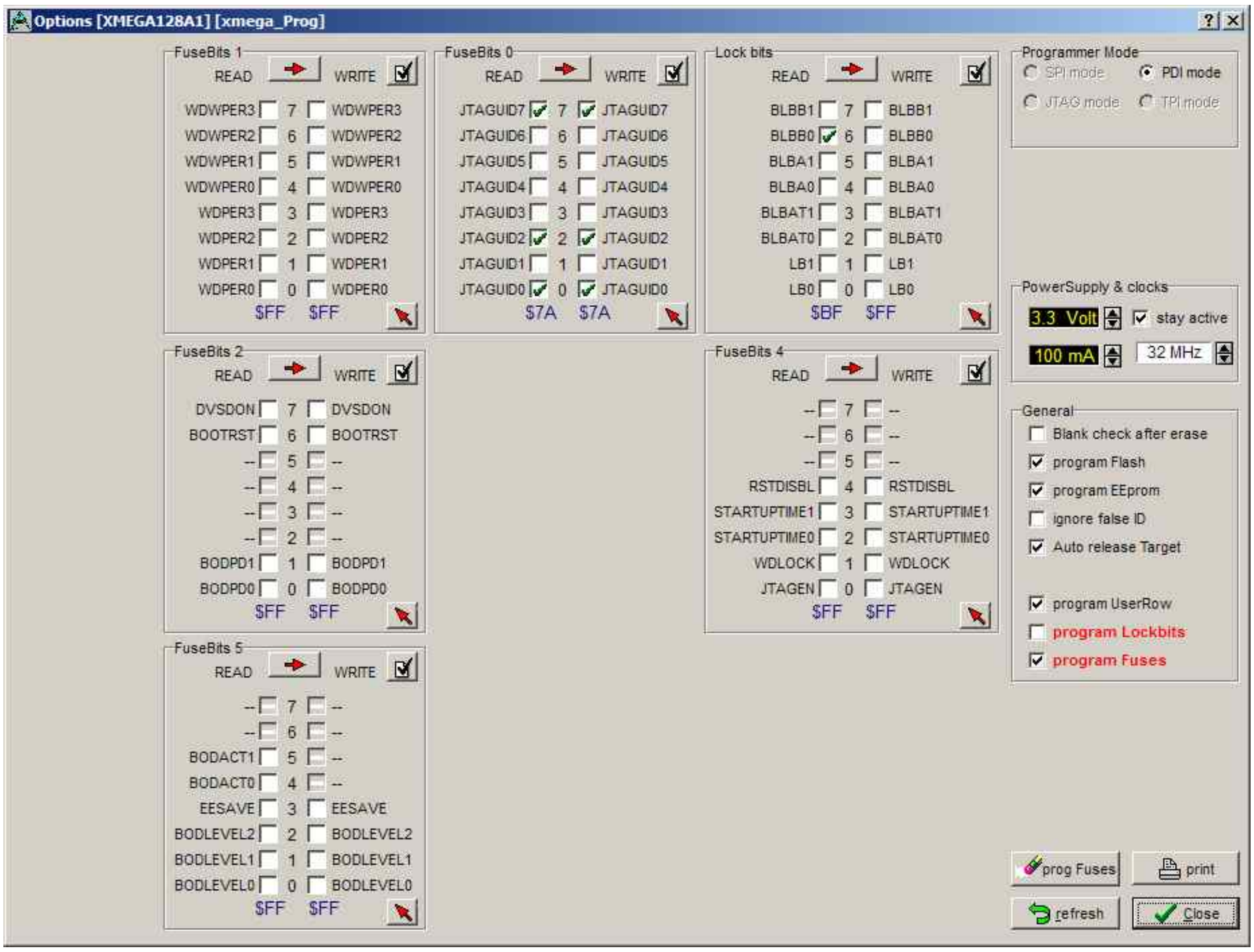
Attention: All Fuse and Lockbits are low active. This means if the data sheet shows a zero '0' for a specific bit the corresponding field in this dialog must show a green 'ok'. Then a '1' bit must show an empty field. Atmel always uses negative logic for Fuse and Lockbits!

With the field **ComPort** the interface to use can be selected. With an UPP1-P/R 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 behavior 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



Options [XMEGA128A1] [xmega_Prog]

FuseBits 1

READ		WRITE
WDWPER3 [ ]	7	[ ] WDWPER3
WDWPER2 [ ]	6	[ ] WDWPER2
WDWPER1 [ ]	5	[ ] WDWPER1
WDWPER0 [ ]	4	[ ] WDWPER0
WDPER3 [ ]	3	[ ] WDPER3
WDPER2 [ ]	2	[ ] WDPER2
WDPER1 [ ]	1	[ ] WDPER1
WDPER0 [ ]	0	[ ] WDPER0
$FF		$FF

FuseBits 0

READ		WRITE
JTAGUID7 [✓]	7	[✓] JTAGUID7
JTAGUID6 [ ]	6	[ ] JTAGUID6
JTAGUID5 [ ]	5	[ ] JTAGUID5
JTAGUID4 [ ]	4	[ ] JTAGUID4
JTAGUID3 [ ]	3	[ ] JTAGUID3
JTAGUID2 [✓]	2	[✓] JTAGUID2
JTAGUID1 [ ]	1	[ ] JTAGUID1
JTAGUID0 [✓]	0	[✓] JTAGUID0
$7A		$7A

Lock bits

READ		WRITE
BLBB1 [ ]	7	[ ] BLBB1
BLBB0 [✓]	6	[ ] BLBB0
BLBA1 [ ]	5	[ ] BLBA1
BLBA0 [ ]	4	[ ] BLBA0
BLBAT1 [ ]	3	[ ] BLBAT1
BLBAT0 [ ]	2	[ ] BLBAT0
LB1 [ ]	1	[ ] LB1
LB0 [ ]	0	[ ] LB0
$BF		$FF

FuseBits 2

READ		WRITE
DVSDON [ ]	7	[ ] DVSDON
BOOTRST [ ]	6	[ ] BOOTRST
-- [ ]	5	[ ] --
-- [ ]	4	[ ] --
-- [ ]	3	[ ] --
-- [ ]	2	[ ] --
BODPD1 [ ]	1	[ ] BODPD1
BODPD0 [ ]	0	[ ] BODPD0
$FF		$FF

FuseBits 4

READ		WRITE
-- [ ]	7	[ ] --
-- [ ]	6	[ ] --
-- [ ]	5	[ ] --
RSTDISBL [ ]	4	[ ] RSTDISBL
STARTUPTIME1 [ ]	3	[ ] STARTUPTIME1
STARTUPTIME0 [ ]	2	[ ] STARTUPTIME0
WDLOCK [ ]	1	[ ] WDLOCK
JTAGEN [ ]	0	[ ] JTAGEN
$FF		$FF

FuseBits 5

READ		WRITE
-- [ ]	7	[ ] --
-- [ ]	6	[ ] --
BODACT1 [ ]	5	[ ] --
BODACT0 [ ]	4	[ ] --
EESAVE [ ]	3	[ ] EESAVE
BODLEVEL2 [ ]	2	[ ] BODLEVEL2
BODLEVEL1 [ ]	1	[ ] BODLEVEL1
BODLEVEL0 [ ]	0	[ ] BODLEVEL0
$FF		$FF

Programmer Mode
SPI mode [✓ PDI mode] JTAG mode TPI mode

PowerSupply & clocks
3.3 Volt — [✓] stay active
100 mA — 32 MHz

General
- [ ] Blank check after erase
- [✓] program Flash
- [✓] program EEprom
- [ ] ignore false ID
- [✓] Auto release Target
- [✓] program UserRow
- [ ] program Lockbits
- [✓] program Fuses

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 behavior 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. Please note that PC-powered external USB HUBs may supply only 100mA.

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’ behavior 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.

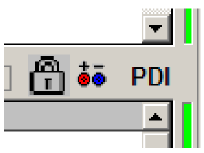
3. Low Frequency Crystal.
   If this mode is activated, a 32kHz watch quartz must be connected to the target. Otherwise the target CPU will be never accessible.

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.

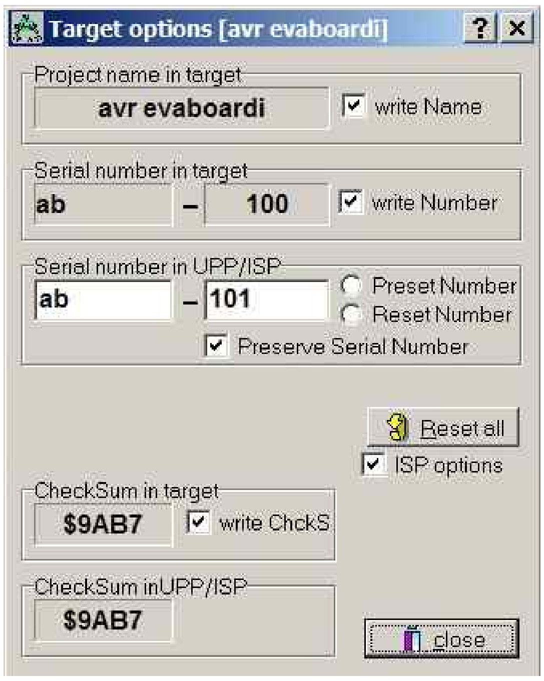
The settings of the Lockbits (protected/unprotected) and the voltage supply of the target by the ISP are also displayed in the main program by two symbols.



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.


Target options [avr evaboardi]

Project name in target: avr evaboardi [✓] write Name

Serial number in target: ab – 100 [✓] write Number

Serial number in UPP/ISP: ab – 101 ( ) Preset Number ( ) Reset Number [✓] Preserve Serial Number

[✓] ISP options

CheckSum in target: $9AB7 [✓] write ChckS

CheckSum inUPP/ISP: $9AB7

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.

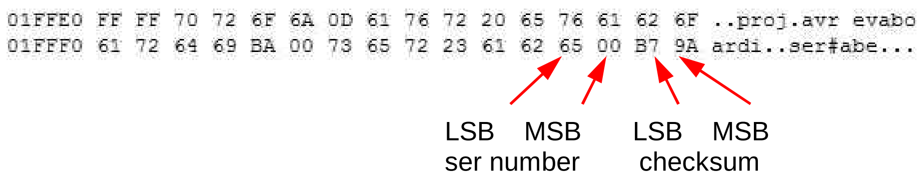
CheckSum into Flash

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.

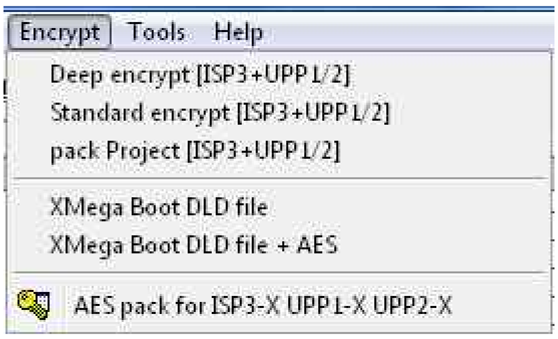
The application/firmware always has access to this parameters. The parameters are stored into the last 16 resp. 32 bytes of the flash. The project name has a lead-in of ‘proj’. The serial number has a preamble of ‘ser#’. The number is always the fourth and third last byte. The checksum, if present, always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



```
01FFE0 FF FF 70 72 6F 6A 0D 61 76 72 20 65 76 61 62 6F ..proj.avr evabo
01FFF0 61 72 64 69 BA 00 73 65 72 23 61 62 65 00 B7 9A ardi..ser#abe...
```

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


Encrypt
Tools
Help
Deep encrypt [ISP3+UPP1/2]
Standard encrypt [ISP3+UPP1/2]
pack Project [ISP3+UPP1/2]
XMega Boot DLD file
XMega Boot DLD file + AES
AES pack for ISP3-X UPP1-X UPP2-X

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 <u>*not*</u> 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 und 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 if 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.

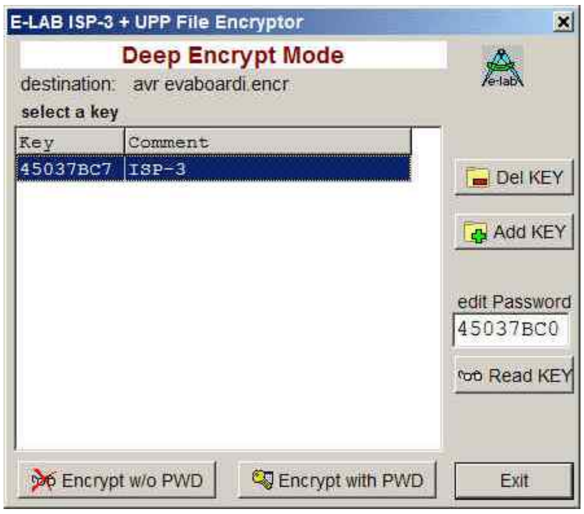
After entering a new password ***Add Key*** adds it to the list.

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 processes 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:



destination: avr evaboardi.encr

Key	Comment
45037BC7	ISP-3

edit Password: 45037BC0

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

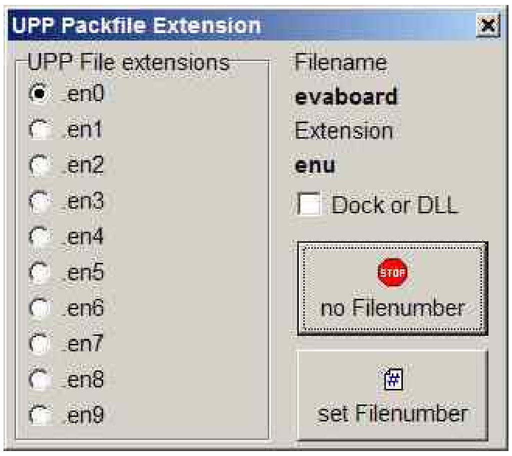
Encrypt with PWD button creates a protection password. This ties the generated file to the programmer that generated this password.

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

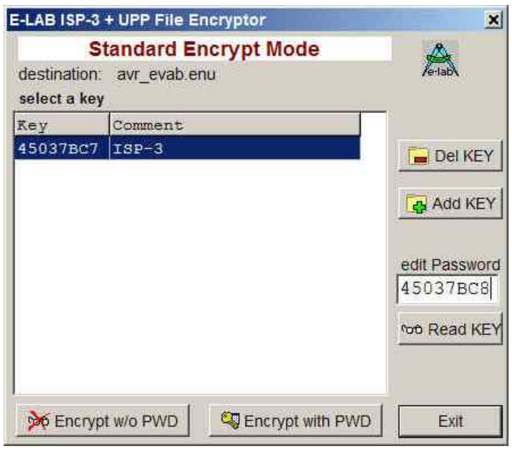
This option builds a packed and **well encrypted** project which only can be processed with the ***PackProg*** program in conjunction with an ISP-3 programmer type. But also a UPP programmer can directly load (using its memory card) and process such a file. As an option, a password can be included so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



Filename: evaboard
Extension: enu

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.



destination: avr_evab.enu
select a key

Key	Comment
45037BC7	ISP-3

edit Password 45037BC8

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.

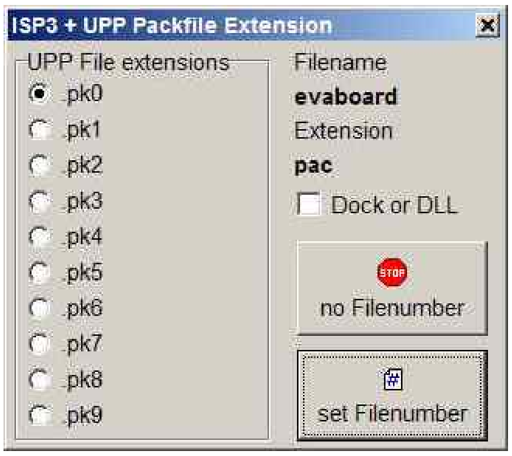
This option must be selected by the button *Encrypt with PWD* or disabled with the button *Encrypt w/o PWD*.

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.



Filename **evaboard**
Extension **pac**

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

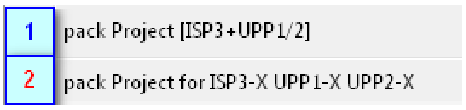
The program *‘PackProg.exe’* supports all programmer types *ISP3, UPP1* and *UPP1*.
Find additional information below.

AES PAC files

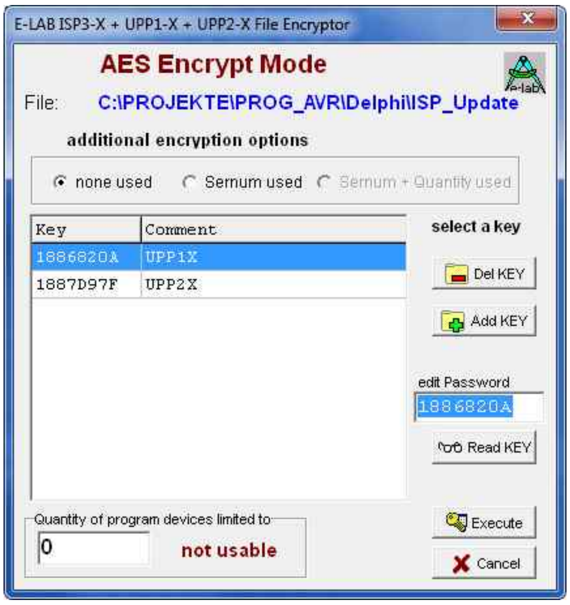
If absolutely secure PAC files are needed for the types ISP3-X, UPP1-X or UPP2-X ('X' programmer) they must be created with **AES encryption**. AES is an absolutely secure encryption which can't be 'hacked'. Because the encryption can only be done **in** the 'X' programmer types, listening (sniffing)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



E-LAB ISP3-X + UPP1-X + UPP2-X File Encryptor

AES Encrypt Mode

File: C:\PROJEKTE\PROG_AVR\Delphi\ISP_Update

additional encryption options

[✓ none used] Sernum used Sernum + Quantity used

Key	Comment
1886820A	UPP1X
1887D97F	UPP2X

select a key

edit Password: 1886820A

Quantity of program devices limited to: 0 not usable

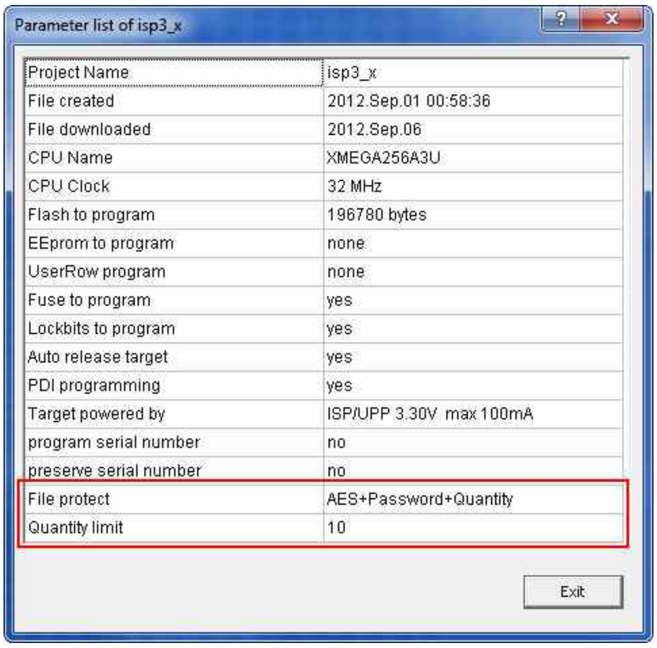
To create an AES PAC it is mandatory to have an 'X' programmer connected to the PC.

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

A. Standard Encryption (only AES used)

B. 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.

C. 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.



Parameter list of isp3_x

Project Name	isp3_x
File created	2012.Sep.01 00:58:36
File downloaded	2012.Sep.06
CPU Name	XMEGA256A3U
CPU Clock	32 MHz
Flash to program	196780 bytes
EEprom to program	none
UserRow program	none
Fuse to program	yes
Lockbits to program	yes
Auto release target	yes
PDI programming	yes
Target powered by	ISP/UPP 3.30V max 100mA
program serial number	no
preserve serial number	no
File protect	AES+Password+Quantity
Quantity limit	10

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.

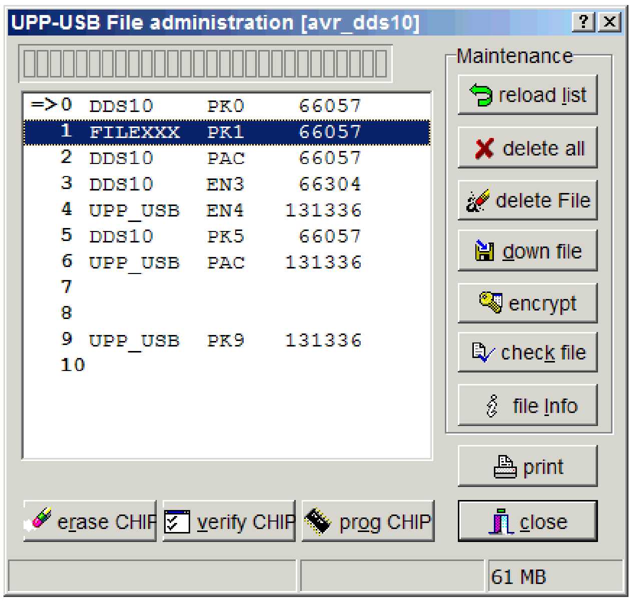
Programming directly

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.



```
=>0  DDS10     PK0     66057
  1  FILEXXX   PK1     66057
  2  DDS10     PAC     66057
  3  DDS10     EN3     66304
  4  UPP_USB   EN4    131336
  5  DDS10     PK5     66057
  6  UPP_USB   PAC    131336
  7
  8
  9  UPP_USB   PK9    131336
 10
```

Select the project by a click in the list box.
This enables the erase, verify and pro buttons.

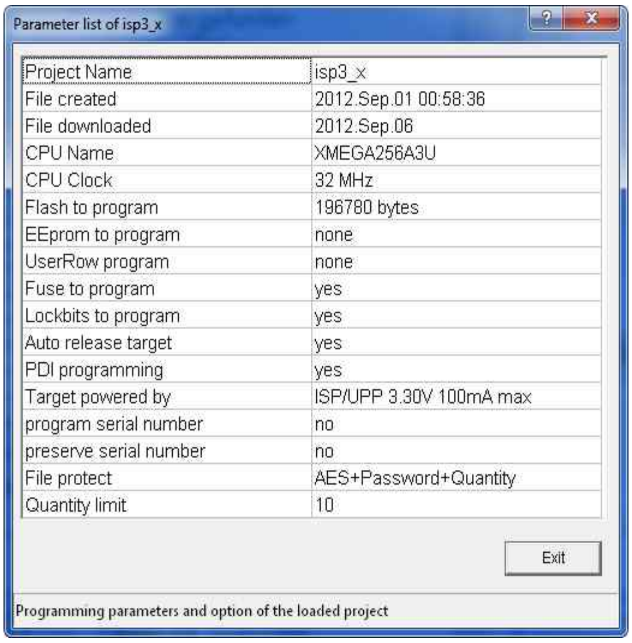
prog Chip programs the CPU

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

UPP/**S** has a rotary switch on the rear that allows the selection of one from 10 possible projects.
The file numbers correspond to the switch position (0..9). The arrow in the list shows the actual position of the rotary switch. Without option "S" it shows always Position.0.



Project Name	isp3_x
File created	2012.Sep.01 00:58:36
File downloaded	2012.Sep.06
CPU Name	XMEGA256A3U
CPU Clock	32 MHz
Flash to program	196780 bytes
EEprom to program	none
UserRow program	none
Fuse to program	yes
Lockbits to program	yes
Auto release target	yes
PDI programming	yes
Target powered by	ISP/UPP 3.30V 100mA max
program serial number	no
preserve serial number	no
File protect	AES+Password+Quantity
Quantity limit	10

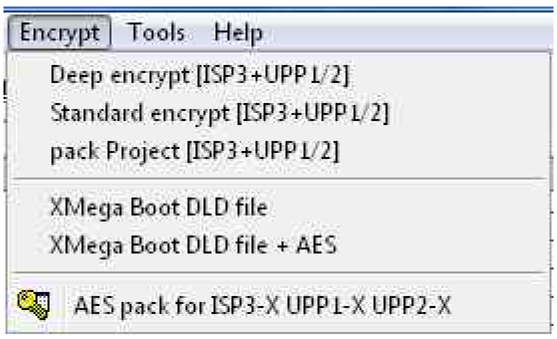
File Info shows the relevant parameters of the selected project.

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


Encrypt
Tools
Help
Deep encrypt [ISP3+UPP1/2]
Standard encrypt [ISP3+UPP1/2]
pack Project [ISP3+UPP1/2]
XMega Boot DLD file
XMega Boot DLD file + AES
AES pack for ISP3-X UPP1-X UPP2-X

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

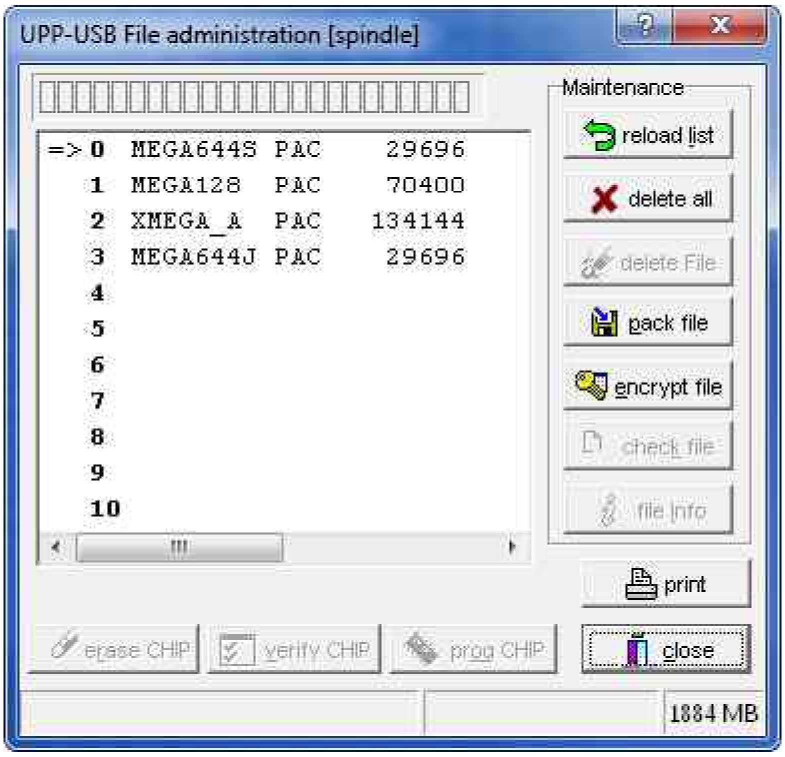
Packed files can be loaded into the UPP programmer types

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.



UPP-USB File administration [spindle]

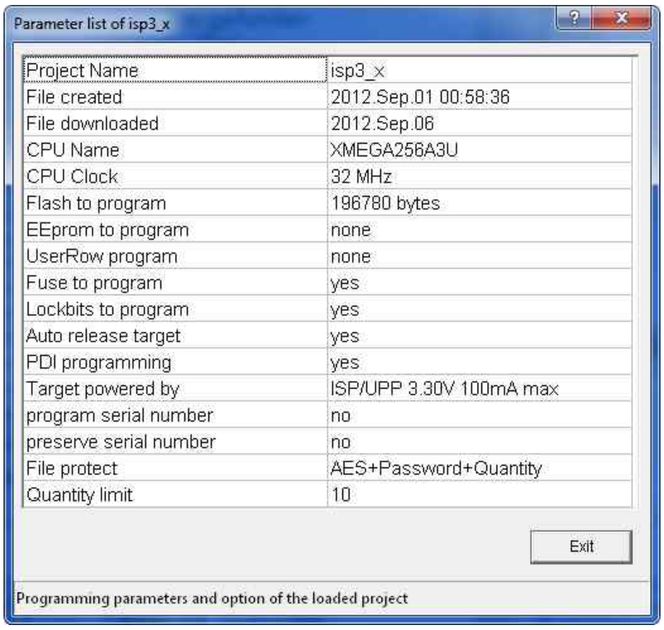
```
=> 0  MEGA644S PAC     29696
   1  MEGA128  PAC     70400
   2  XMEGA_A  PAC    134144
   3  MEGA644J PAC     29696
   4
   5
   6
   7
   8
   9
   10
```

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.



Parameter list of isp3_x

Project Name	isp3_x
File created	2012.Sep.01 00:58:36
File downloaded	2012.Sep.06
CPU Name	XMEGA256A3U
CPU Clock	32 MHz
Flash to program	196780 bytes
EEprom to program	none
UserRow program	none
Fuse to program	yes
Lockbits to program	yes
Auto release target	yes
PDI programming	yes
Target powered by	ISP/UPP 3.30V 100mA max
program serial number	no
preserve serial number	no
File protect	AES+Password+Quantity
Quantity limit	10

Programming parameters and option of the loaded project

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.

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.

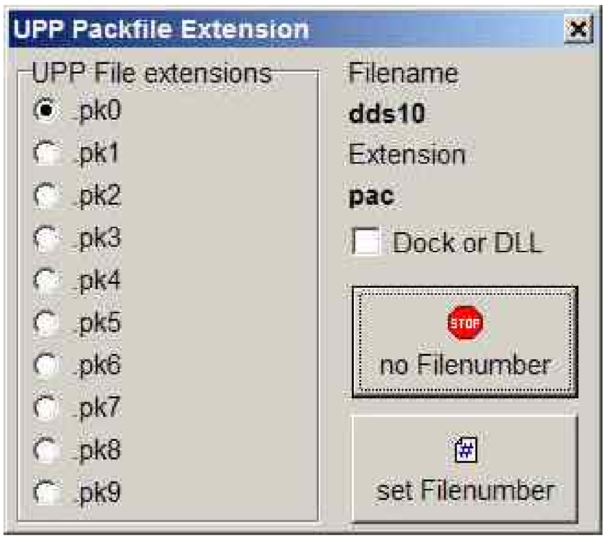
With selecting a project in these ways there is the problem that the relation of projects on the SD card to the position of the switch is not static. It absolutely depends on the order of the FAT16 directory entries on the SD card. If a card becomes completely erased and then projects are stored sequential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

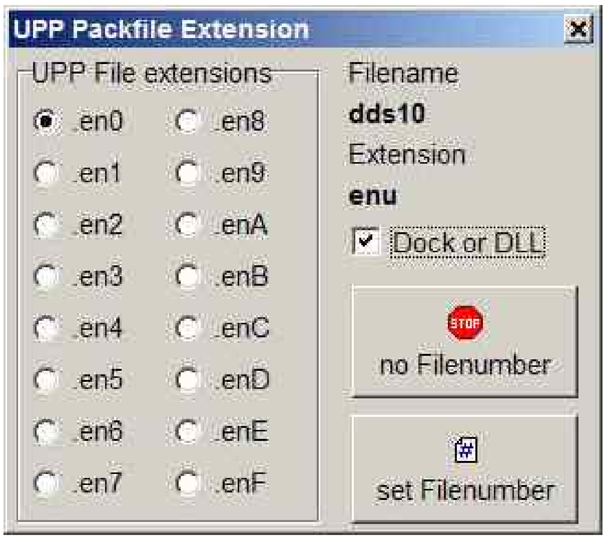
Basically all UPP Pack Files have the file extension ***.pac*** and the encrypted types have the extension ***.enu***.
To set a fixed relation between such a project and the selection switch a number between 0 and 9 can be appended to the extension which forces the UPP to fix this project to a switch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

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


UPP Packfile Extension

UPP File extensions: [✓ .pk0] .pk1 .pk2 .pk3 .pk4 .pk5 .pk6 .pk7 .pk8 .pk9

Filename
dds10
Extension
pac
☐ Dock or DLL


UPP Packfile Extension

UPP File extensions: [✓ .en0] .en1 .en2 .en3 .en4 .en5 .en6 .en7 .en8 .en9 .enA .enB .enC .enD .enE .enF

Filename
dds10
Extension
enu
☑ Dock or DLL

If the button ***no Filenumber*** is pressed a standard *.pac or *.enu project will be build. If the button ***set Filenumber*** 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.

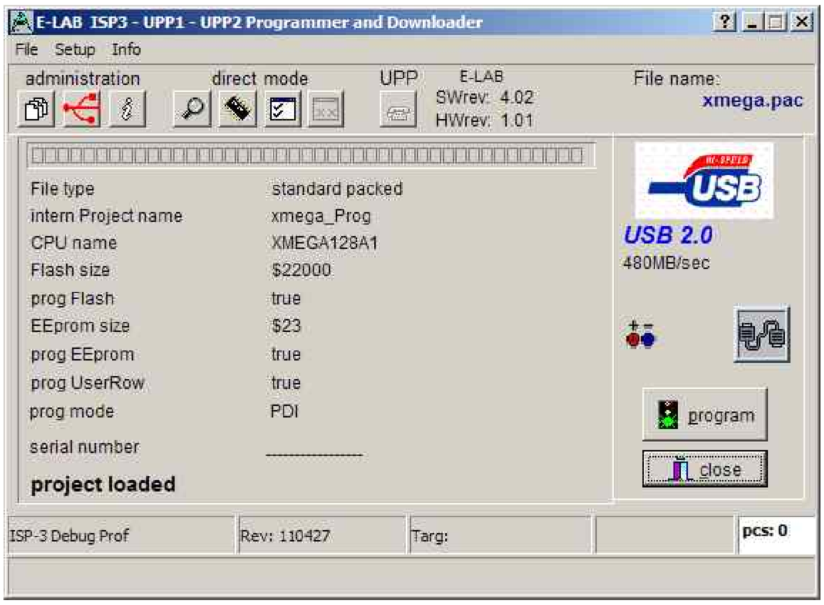
Most of these functions are not necessarily desired in the production and service area. And furthermore, they distract and can be possible sources for handling errors.

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

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.

Project Import



E-LAB
SWrev: 4.02
HWrev: 1.01

File name:
xmega.pac

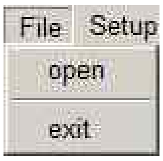
File type	standard packed
intern Project name	xmega_Prog
CPU name	XMEGA128A1
Flash size	$22000
prog Flash	true
EEprom size	$23
prog EEprom	true
prog UserRow	true
prog mode	PDI
serial number	----------------

project loaded

ISP-3 Debug Prof | Rev: 110427 | Targ: | pcs: 0

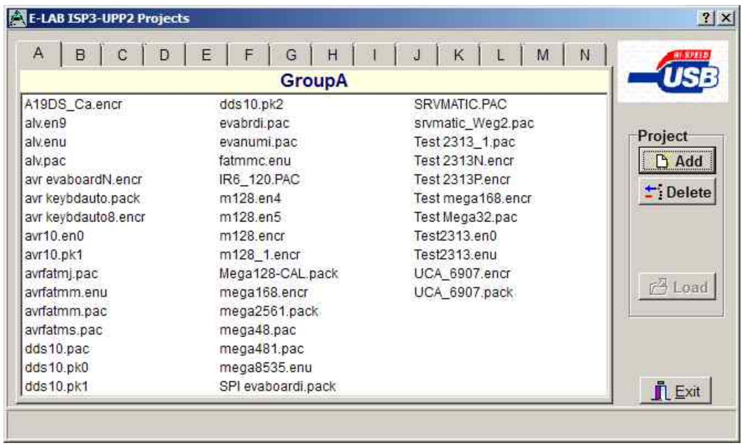
This means that a new project must be registered first in order to work with it.

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



E-LAB ISP3-UPP2 Projects

A | B | C | D | E | F | G | H | I | J | K | L | M | N

GroupA

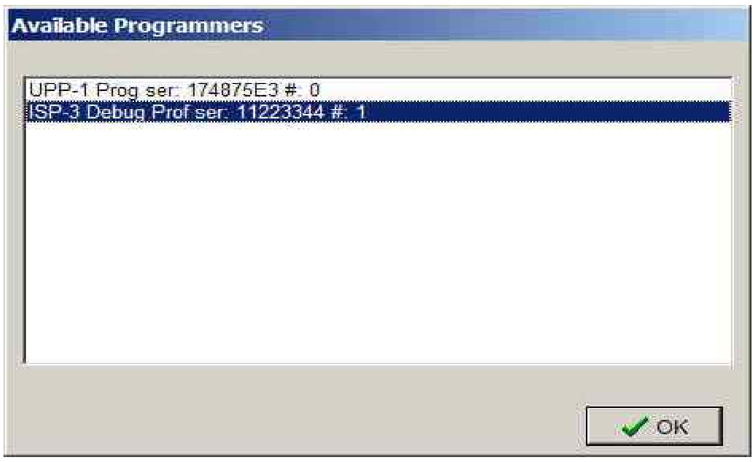
```
A19DS_Ca.encr          dds10.pk2            SRVMATIC.PAC
alv.en9                evabrdi.pac          srvmatic_Weg2.pac
alv.enu                evanumi.pac          Test 2313_1.pac
alv.pac                fatmmc.enu           Test 2313N.encr
avr evaboardN.encr     IR6_120.PAC          Test 2313P.encr
avr keybdauto.pack     m128.en4             Test mega168.encr
avr keybdauto8.encr    m128.en5             Test Mega32.pac
avr10.en0              m128.encr            Test2313.en0
avr10.pk1              m128_1.encr          Test2313.enu
avrfatmj.pac           Mega128-CAL.pack     UCA_6907.encr
avrfatmm.enu           mega168.encr         UCA_6907.pack
avrfatmm.pac           mega2561.pack
avrfatms.pac           mega48.pac
dds10.pac              mega481.pac
dds10.pk0              mega8535.enu
dds10.pk1              SPI evaboardi.pack
```

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



Available Programmers

```
UPP-1 Prog ser: 174875E3 #: 0
ISP-3 Debug Prof ser: 11223344 #: 1
```

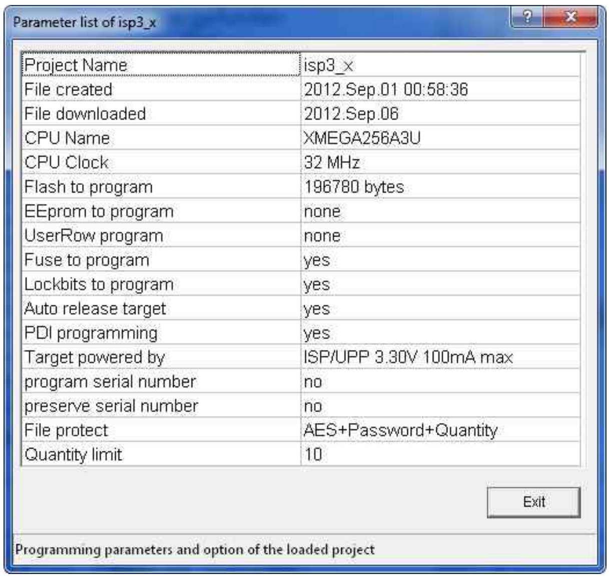
Before one can work with the selected and loaded project the programmer must be searched for. This must be done with the Check-USB button.

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



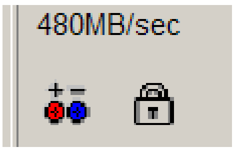
Parameter list of isp3_x

Project Name	isp3_x
File created	2012.Sep.01 00:58:36
File downloaded	2012.Sep.06
CPU Name	XMEGA256A3U
CPU Clock	32 MHz
Flash to program	196780 bytes
EEprom to program	none
UserRow program	none
Fuse to program	yes
Lockbits to program	yes
Auto release target	yes
PDI programming	yes
Target powered by	ISP/UPP 3.30V 100mA max
program serial number	no
preserve serial number	no
File protect	AES+Password+Quantity
Quantity limit	10

Programming parameters and option of the loaded project

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

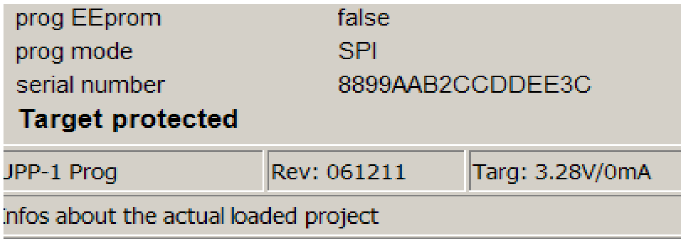
Device Check. With this button a check is done of the programmer and the target device. A check of the target's supply voltage and reading and checking the device ID of target is included, if possible.

Verify

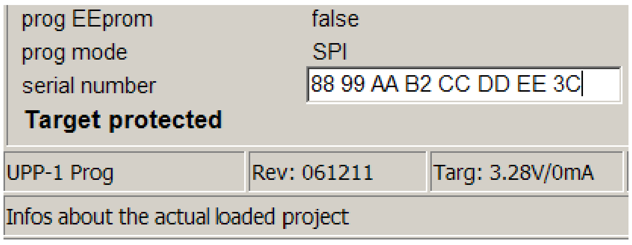
Verify Chip. Verify the contents of the Flash and EEPROM. Of course this is only possible if the chip is not protected.

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.



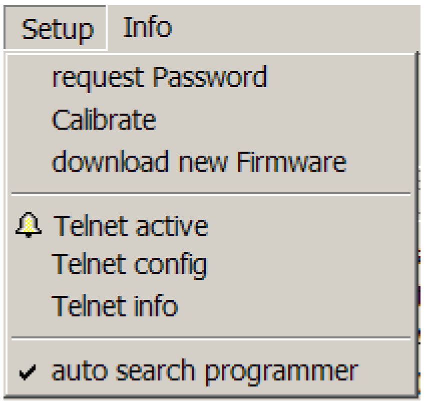
```
prog EEprom          false
prog mode            SPI
serial number        8899AAB2CCDDEE3C
 Target protected
JPP-1 Prog | Rev: 061211 | Targ: 3.28V/0mA
Infos about the actual loaded project
```



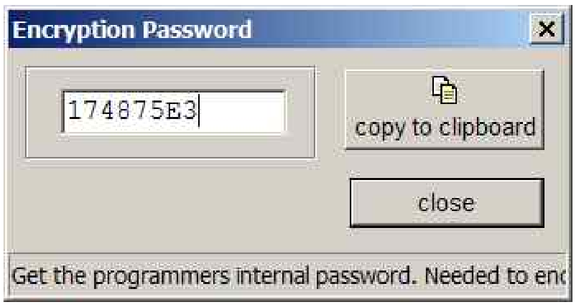
```
prog EEprom          false
prog mode            SPI
serial number        88 99 AA B2 CC DD EE 3C
 Target protected
UPP-1 Prog | Rev: 061211 | Targ: 3.28V/0mA
Infos about the actual loaded project
```

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



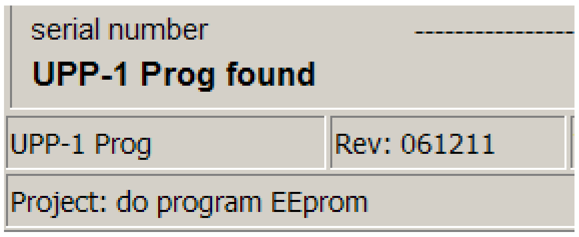
request Password must be used in the Setup Menu.



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**

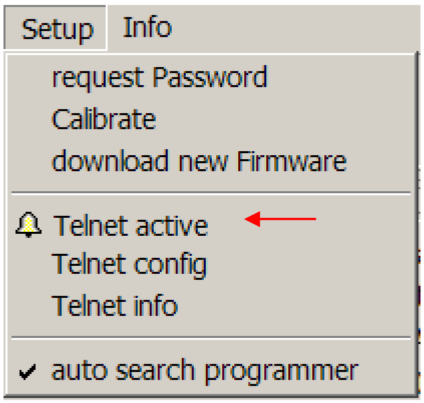
Return Codes

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, e.g. Hexfile, was not found
14	dsCalReq	Programmer returns a lost or illegal calibration

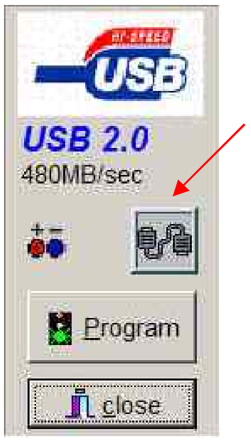
Telnet Interface

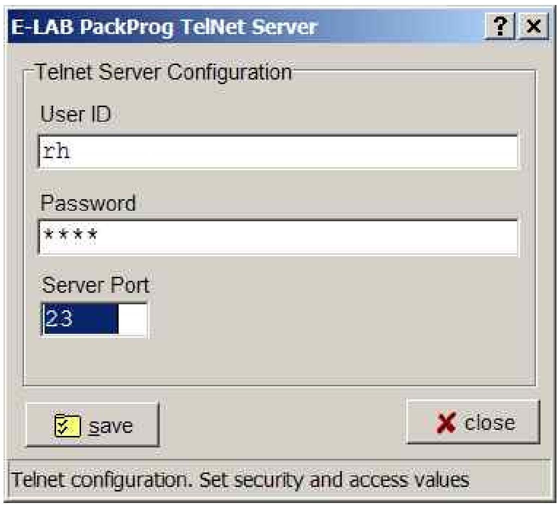
There is an easy to use remote control interface in ***PackProg.*** To simplify this remote control the Telnet protocol is used. So other applications and also other PCs can remotely control a programmer via Telnet command strings.



This menu item enables and disables the PackProg Telnet Server.

The active Telnet Server is displayed in the main window:





Telnet Server Configuration

User ID: rh

Password: ****

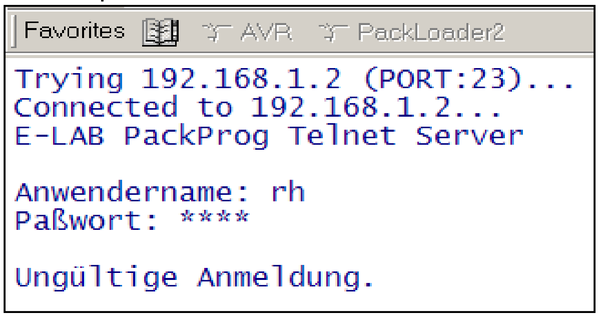
Server Port: 23

Telnet configuration. Set security and access values

Telnet config

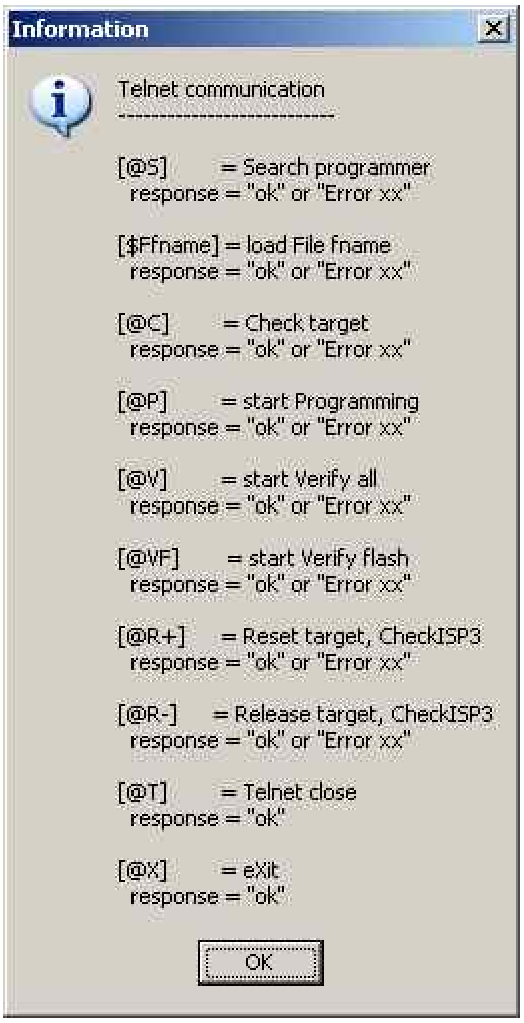
The Telnet Server needs an USER ID and a password. If a Telnet Client (another application) connects to the Server this Client must provide the correct ID and password. This disables unauthorized Telnet Clients so they can't take control of the Server or influence it.

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



```
Trying 192.168.1.2 (PORT:23)...
Connected to 192.168.1.2...
E-LAB PackProg Telnet Server

Anwendername: rh
Paßwort: ****

Ungültige Anmeldung.
```

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



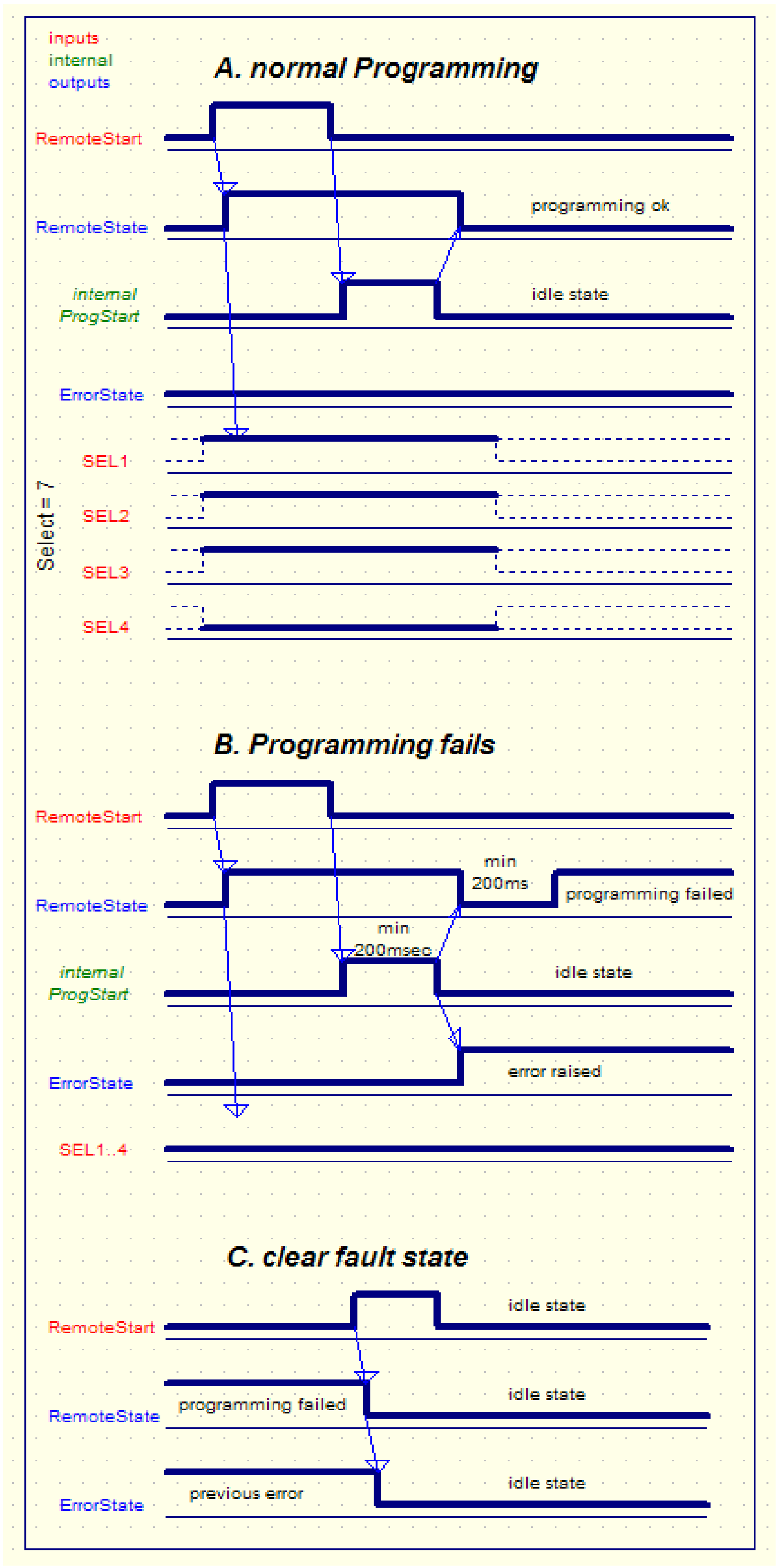
```
Telnet communication
----------------------------

[@S]       = Search programmer
  response = "ok" or "Error xx"

[$Ffname] = load File fname
  response = "ok" or "Error xx"

[@C]       = Check target
  response = "ok" or "Error xx"

[@P]       = start Programming
  response = "ok" or "Error xx"

[@V]       = start Verify all
  response = "ok" or "Error xx"

[@VF]      = start Verify flash
  response = "ok" or "Error xx"

[@R+]      = Reset target, CheckISP3
  response = "ok" or "Error xx"

[@R-]      = Release target, CheckISP3
  response = "ok" or "Error xx"

[@T]       = Telnet close
  response = "ok"

[@X]       = eXit
  response = "ok"
```

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

WIRED REMOTE CONTROL

If the programming system must be able to select remote one of the projects stored in the SD card, the version "**PR**" must be used. It also provides an easy interface to any kind of controlling hardware.

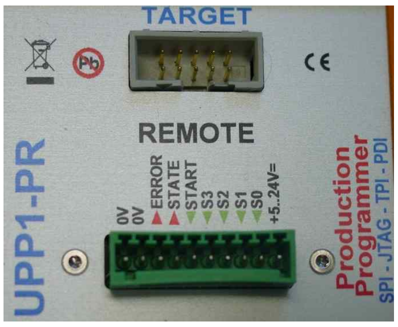


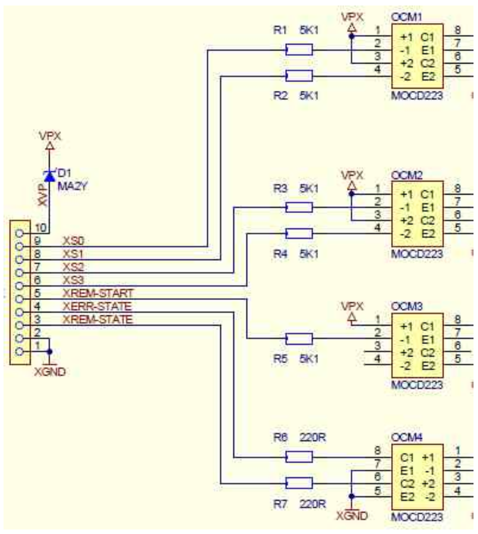
This interface works with switches, resp. open-collector. A switch is closed if active.

The diagram on the left shows the functionality of the remote interface. Please note that logic levels are shown here. A „high" means the input resp. the output is „active".

Active now means that on the an input a switch or contact is closed and pulls down the input to 0V.
The same is true for outputs. A „high" means that the internal switch (optocoupler) is closed and the output is pulled down to ground.
All this is a so called ***negative logic***.

With 4 binary select lines select out of 16 projects can be selected.
All 4 inputs open means that „Project 0" is selected. All 4 inputs closed means that „Project 15" is selected. Input0 active means „Project 1" is selected.





Attention: If the UPP1-PR must also supply the external hardware (Target) so it is possible that a 5V supply at the Remote connector is not sufficient. Then 6V or higher should be provided there.

Switching from USB to Remote and vice versa

Within a production environment sometimes it is necessary to switch between Remote mode (wired controlling) and the USB control. For example while programming is in progress a new project must be downloaded or exchanged on the internal SD card of the UPP. This needs a controlled switch from the Remote mode to the USB mode. After that the system must be switched back to the Remote mode.

Because the USB interface is inactive in the Remote mode so this switching can’t be done via the USB interface. On the other hand it is nearly impossible to switch from the USB mode to the Remote mode using the USB interface.

Because with very most cases the Wire/Remote control is always connected to the UPP1-PR so this switching must always be executed through the Remote/Wire interface. So the UPP1 always recognizes the Remote Start signal in both modes.

If the USB mode is active an active Start signal longer than 5sec starts a change to the Remote mode and the USB interface becomes deactivated. The Remote/wire interface now gains the control and is ready to be used.

If the Remote mode is active an active Start signal of the wire control longer than 5sec changes the system to the USB mode. The Remote/Wire control now is deactivated except the Start input.

The Start input is always ready and reacts on the activation which lasts longer than 5 sec with a mode switching

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

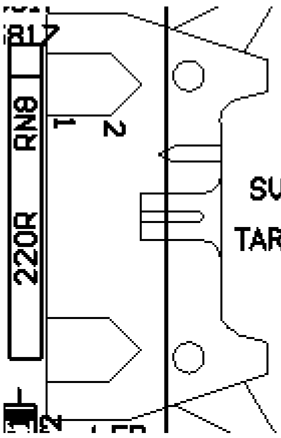
Support for direct patching of files or targets

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.



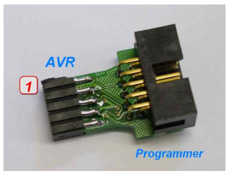
So it's a good idea to design system with less or no loads on the programming lines. Resistance below 2k Ohm and capacitors larger than 100pF should be avoided in conjunction with the programming lines. This is true for both SPI-programming and also for JTAG-programming.

If it becomes necessary to drive low resistance loads the internal protection resistors (10pin single-inline-network) can be replaced by 5x 50 Ohm resistor or short circuits. The resistor network is located directly behind the 10 pin ribbon cable header and is removable because it has a single-inline socket.

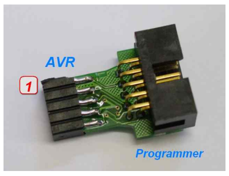
A separate 50 Ohm resistor network is included as a spare part.

Miscellaneous Adaptors

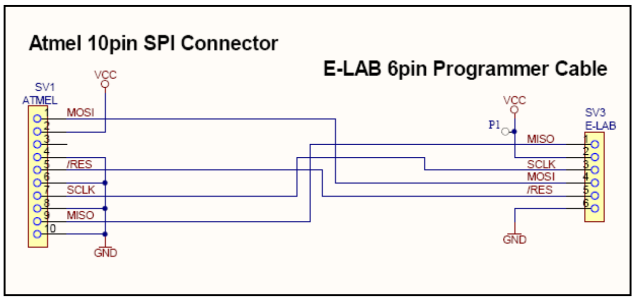
These adapters are useful if your target board has connectors with Atmel standard pinouts.

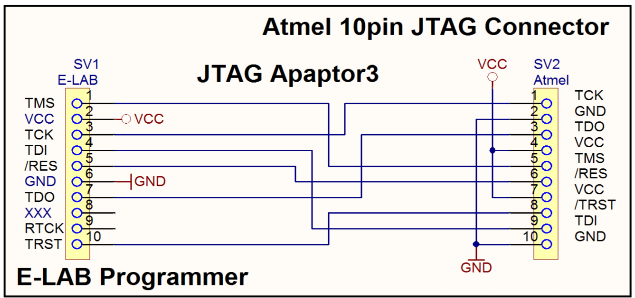


As an option an adaptor is available from E-LAB 6 pin SPI to Atmel 10 pin SPI   #2125



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





TARGET POWER SUPPLY

All USB 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 >=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 80mA. So maximum achievable current to the target can be approximately 20mA.

If the programmer is directly connected to the PC the internal power supply can deliver up to 300mA. With the version “PR” the programmer is supplied by the remote control supply voltage (5..24V).

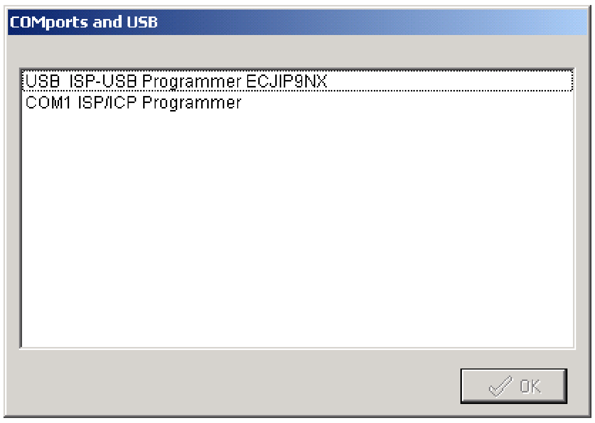
```
o1  2o
o3  4o
o5  6o
o7  8o
o9 10o
```

The programming connector of the UPP1-P/R 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

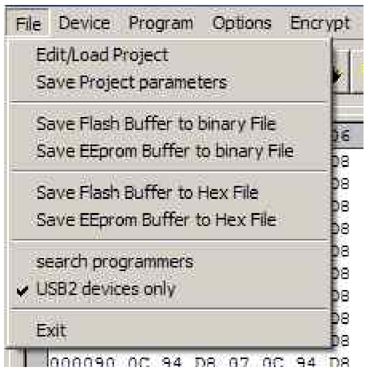


USB ISP-USB Programmer ECJIP9NX
COM1 ISP/ICP Programmer

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.


File
Device
Program
Options
Encrypt
Edit/Load Project
Save Project parameters
Save Flash Buffer to binary File
Save EEprom Buffer to binary File
Save Flash Buffer to Hex File
Save EEprom Buffer to Hex File
search programmers
✓ USB2 devices only
Exit

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 WIN7/8 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.

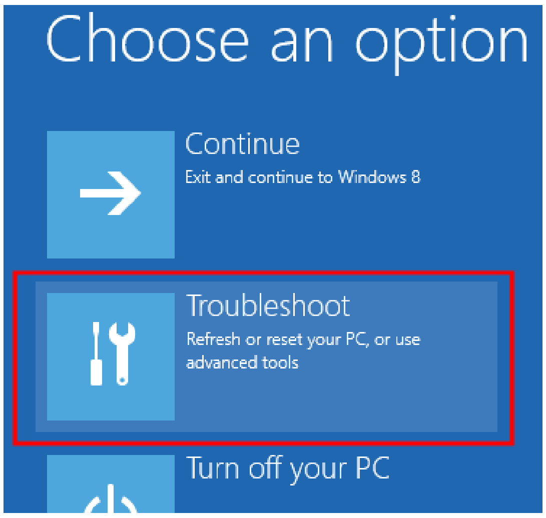
Installing unsigned drivers

In order to force the installation of unsigned (not Microsoft certified) drivers under Windows8/64 this must be enabled before the start-up of the Windows system with “cmd shutdown /r /o“. Windows now executes a restart.

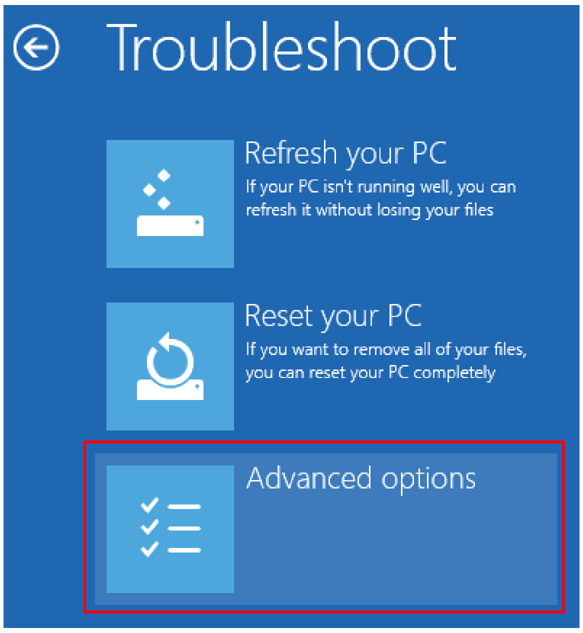
This concerns the E-LAB AVRprog USB driver and also the AVRco USB driver.

Step 1:

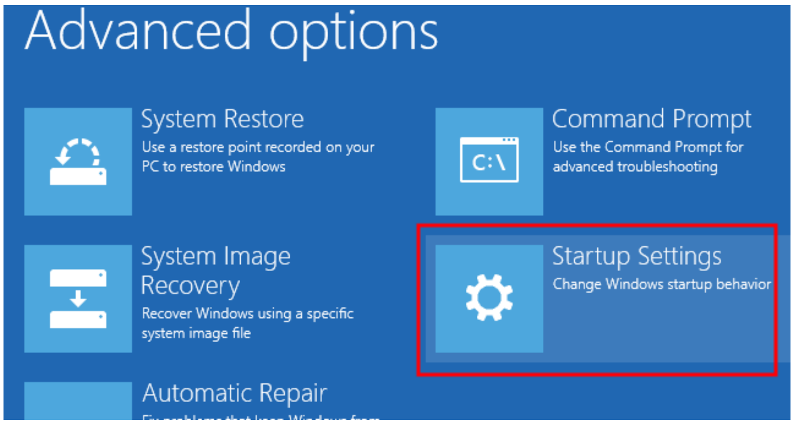
If Windows 8 has started then the Boot-Options-Menu must be started. To do this, on the Windows 8 desktop the button combination of ***Win+R*** must be pressed to open the “Execute”-dialog. In this dialog the following command must be typed in to start the “Options”-menu at boot time:


Choose an option
Continue
Exit and continue to Windows 8
Troubleshoot
Refresh or reset your PC, or use advanced tools
Turn off your PC

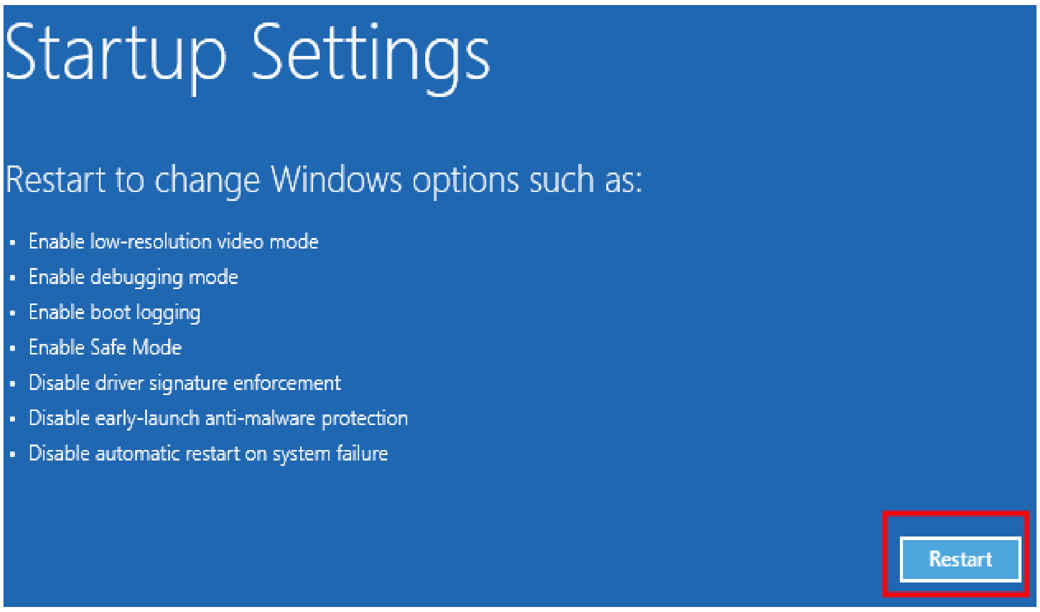
Step 2: In the „Options“-menu click onto ”Troubleshoot”.


Troubleshoot
Refresh your PC
If your PC isn't running well, you can refresh it without losing your files
Reset your PC
If you want to remove all of your files, you can reset your PC completely
Advanced options

Step3: click onto “advanced options”.


Advanced options
System Restore
Use a restore point recorded on your PC to restore Windows
Command Prompt
Use the Command Prompt for advanced troubleshooting
System Image Recovery
Recover Windows using a specific system image file
Startup Settings
Change Windows startup behavior
Automatic Repair

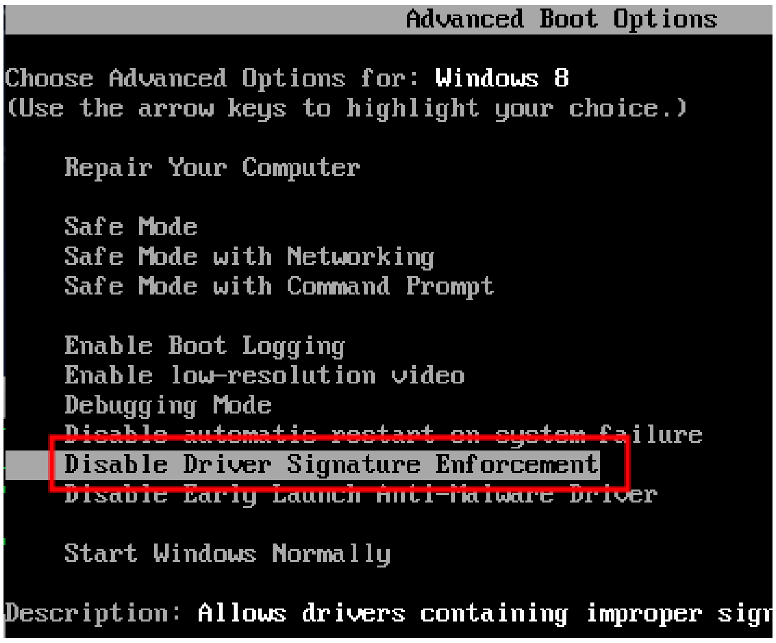
Step4: click onto “Startup Settings”.


Startup Settings

Restart to change Windows options such as:

- Enable low-resolution video mode
- Enable debugging mode
- Enable boot logging
- Enable Safe Mode
- Disable driver signature enforcement
- Disable early-launch anti-malware protection
- Disable automatic restart on system failure

Step5: click onto “Restart”.


```
                        Advanced Boot Options

Choose Advanced Options for: Windows 8
(Use the arrow keys to highlight your choice.)

    Repair Your Computer

    Safe Mode
    Safe Mode with Networking
    Safe Mode with Command Prompt

    Enable Boot Logging
    Enable low-resolution video
    Debugging Mode
    Disable automatic restart on system failure
    Disable Driver Signature Enforcement
    Disable Early Launch Anti-Malware Driver

    Start Windows Normally

Description: Allows drivers containing improper sign
```

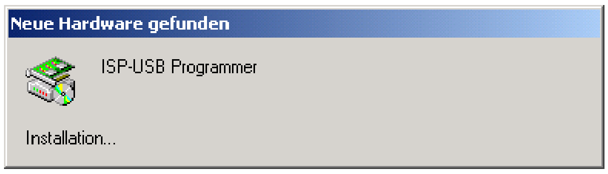
Step6: Choose **Disable Driver Signature Enforcement** and Enter key to start Windows

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.



Neue Hardware gefunden

ISP-USB Programmer

Installation...

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

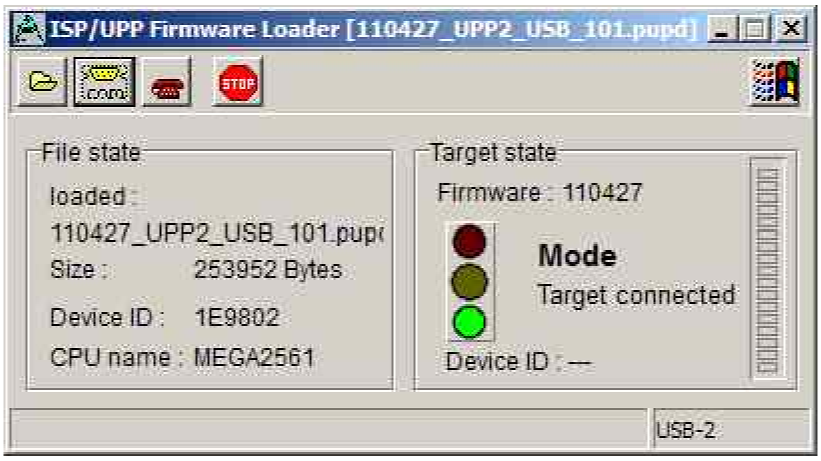
The latest E-LAB programmers support downloading of new firmware into the programmer. You can determine whether your device has this feature if you open the **Device Status** dialog. If this device is updateable then in the line Update a number appears which shows the current firmware date, which is 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_P.pupd*** or from WEB in file: ***UPP1_PR_update.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



File state
loaded :
110427_UPP2_USB_101.pup
Size : 253952 Bytes
Device ID : 1E9802
CPU name : MEGA2561

Target state
Firmware : 110427
Mode
Target connected
Device ID : ---

USB-2

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-P/R has a RESET button on the top of the case.

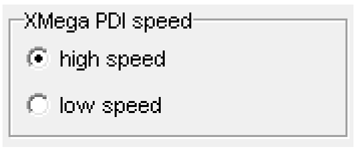
This RESET button has two functions:

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.

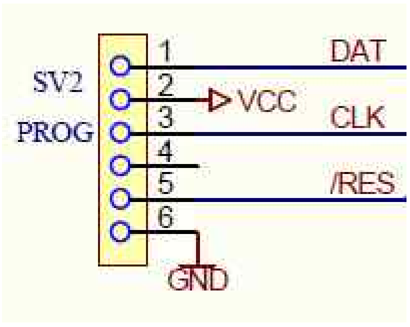

XMega PDI speed
[✓ high speed]
low speed

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.

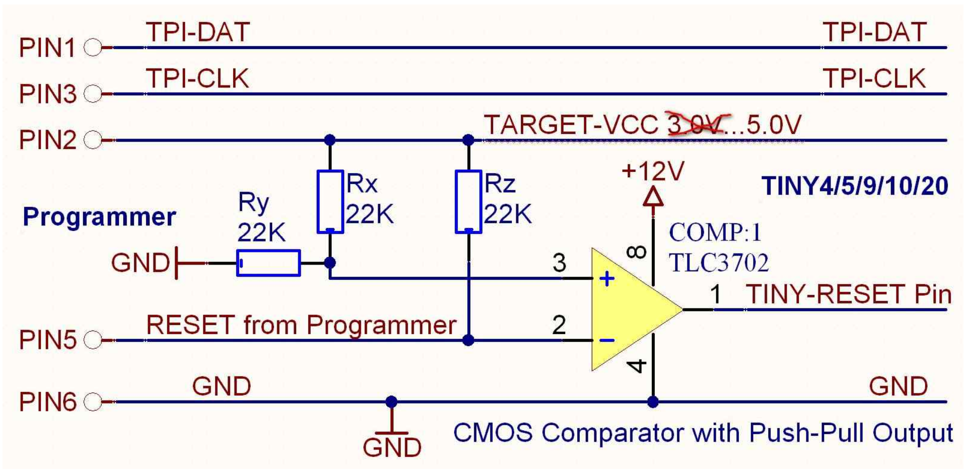
The programming mode is called TPI. Because the same programming plug is also used by the SPI mode the plug on the target system differs from the Atmel plug. The load on the TPI_DAT pin must not be lower than 80kOhm.



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:

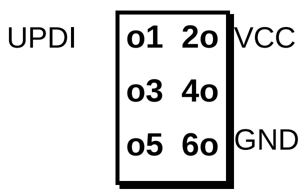


UPDI Tiny and UPDI Mega

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 oft he E-LAB programmers.

The pin-out of t he 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.

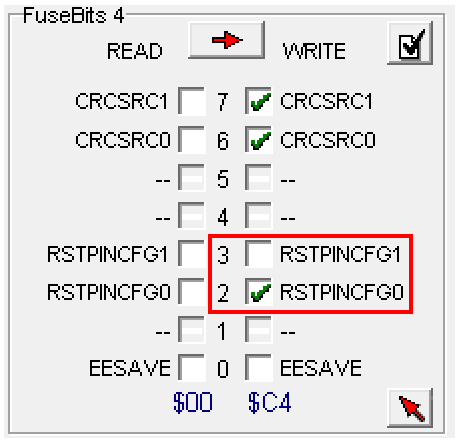
In this case a special12V pulse at the UPDI pin cab remove this fuse setting. But therefore a special algorithmus is necessary. This is also supported by these programmers. Because all must fit together, timings and voltage actions, a special external hardware must be present.

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.

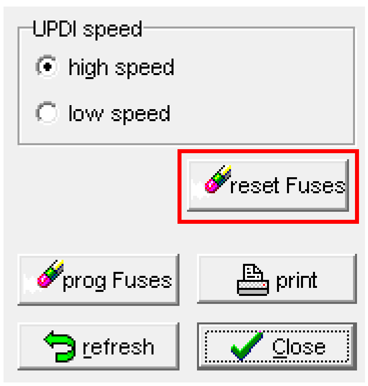
But please note that parts/devices connected to the UPDI of the chips must be 12V tolerant. Also heavy loads must be avoided.

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



FuseBits 4

READ		WRITE
CRCSRC1 ☐	7	✓ CRCSRC1
CRCSRC0 ☐	6	✓ CRCSRC0
-- ☐	5	☐ --
-- ☐	4	☐ --
RSTPINCFG1 ☐	3	☐ RSTPINCFG1
RSTPINCFG0 ☐	2	✓ RSTPINCFG0
-- ☐	1	☐ --
EESAVE ☐	0	☐ EESAVE
$00		$C4



UPDI speed: [✓ high speed] low speed

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

Chipcon

Basics

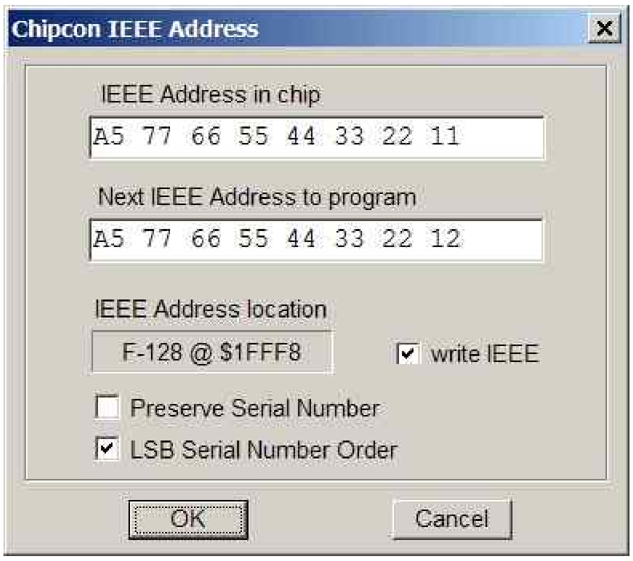
The UPP1-P/R firmware also supports the in-Circuit programmable ZigBee Chip family CC1110, CC2510 and CC2430 from TI-Chipcon. .

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



Chipcon IEEE Address

IEEE Address in chip: A5 77 66 55 44 33 22 11

Next IEEE Address to program: A5 77 66 55 44 33 22 12

IEEE Address location: F-128 @ $1FFF8

[✓] write IEEE

[ ] Preserve Serial Number

[✓] LSB Serial Number Order

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..

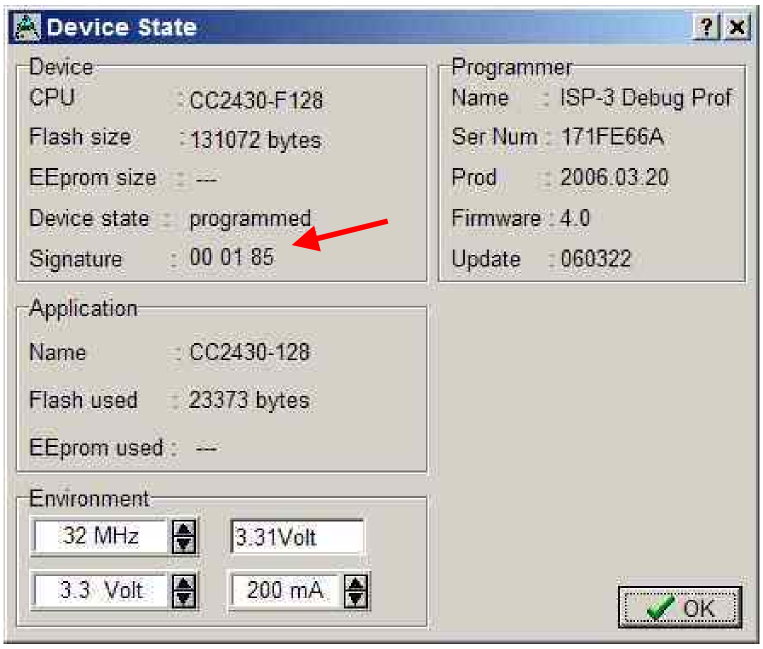
N.C	o1	2o	VCC
(P2_2) CLK	o3	4o	DATA (P2_1)
/RESET	o5	6o	GND

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


Device State

Device
CPU : CC2430-F128
Flash size : 131072 bytes
EEprom size : ---
Device state : programmed
Signature : 00 01 85

Programmer
Name : ISP-3 Debug Prof
Ser Num : 171FE66A
Prod : 2006.03.20
Firmware : 4.0
Update : 060322

Application
Name : CC2430-128
Flash used : 23373 bytes
EEprom used : ---

Environment
32 MHz
3.31Volt
3.3 Volt
200 mA

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.

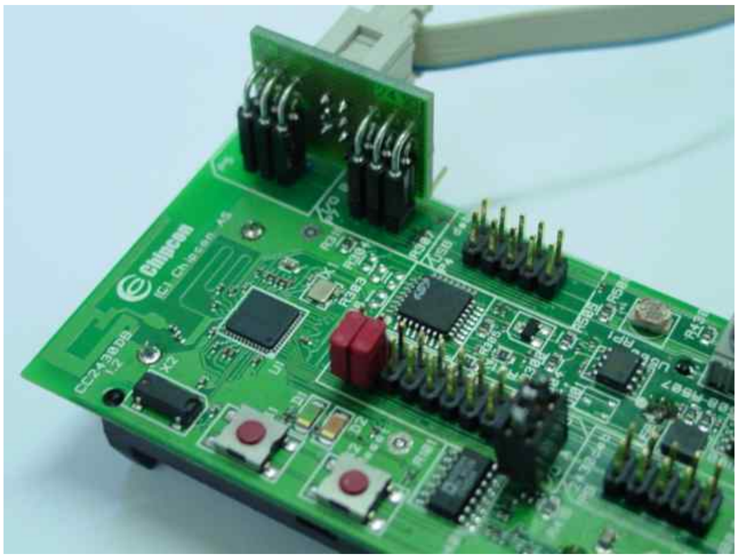
A false chip ID should never be ignored But the reasons for it should be carefully researched.

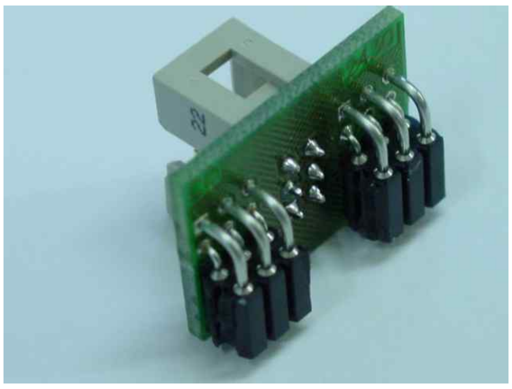
Chipcon Evaluation Boards

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.

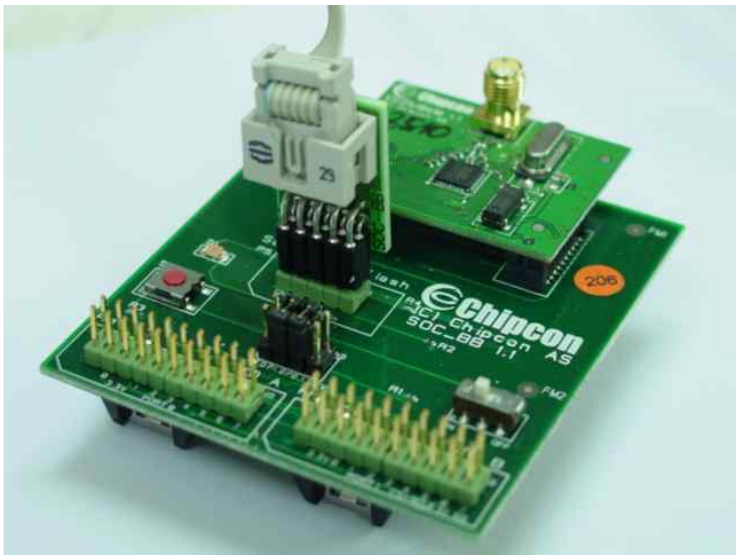


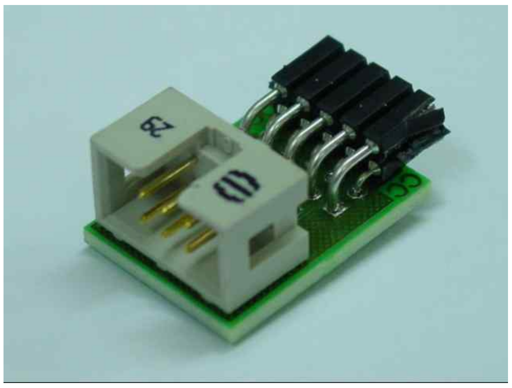


SOC-BB

his 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-P/R firmware also supports the in-Circuit programmable Atmel Chip family AT89LP2052 und 4052.

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

Connectors

The included <u>10 wire</u> 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	o1	2o	VCC
SCLK	o3	4o	MOSI
/RESET	o5	6o	GND
SS	o7	8o	N.C.
N.C.	o9	10o	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	**o1**	**2o**	VCC (3.3V)
SCLK	**o3**	**4o**	MOSI/SI
/CS-SELECT	**o5**	**6o**	GND

TopView header on the Target

OTHER E-LAB PROGRAMMERS

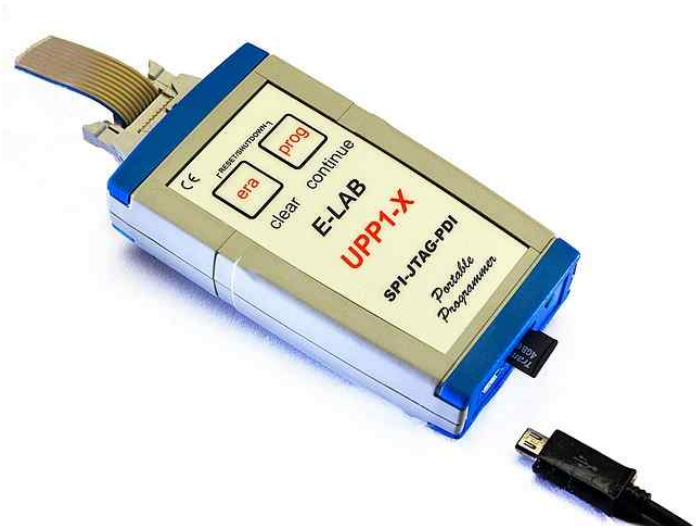
ISP3-X

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.

ISP3-X is an update of the older ISP3-USB model and is faster.



UPP1-X

The Universal Portable Programmer .

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

The connection to the PC is via USB2.

Best suited for field service and small series production work.

UPP1-X is an update of the older UPP1-USB version and is faster.

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 connection to the PC is via USB2.

Also best suited for field service and small series production.

UPP2-X is an update of the older UPP2-USB model and is faster.

Notes