

BATRONIX

PROG-EXPRESS

Manual

TABLE OF CONTENTS

| | |
|---|-----------|
| TABLE OF CONTENTS | 2 |
| SYSTEM REQUIREMENTS | 4 |
| 32-BIT OPERATING SYSTEMS..... | 4 |
| 64-BIT OPERATING SYSTEMS..... | 4 |
| LICENSING INFORMATION FOR PROG-EXPRESS | 5 |
| FREEMWARE LICENSE CONDITIONS | 5 |
| INSTALLATION..... | 6 |
| SOFTWARE INSTALLATION | 6 |
| DEVICE DRIVER INSTALLATION..... | 7 |
| SOFTWARE MODES | 10 |
| SUMMARY | 10 |
| PROGRAM CHIP..... | 11 |
| CHIP COPY | 12 |
| READ CHIP | 13 |
| PRODUCTION MODE | 14 |
| DATA ENTRY FIELDS..... | 17 |
| PROCESS CONTROL..... | 19 |
| PROCESS STEPS | 20 |
| LOG TEXT..... | 22 |
| HEX-EDITOR | 23 |
| HEX-EDITOR MENU AND TOOLBAR..... | 24 |
| HEX EDITOR SHORTCUT KEYS | 25 |
| HEX-EDITOR CONTEXT MENU | 26 |
| WORKING WITH SELECTED DATA..... | 26 |
| CHIP PROGRAMMING IN THE HEX-EDITOR..... | 28 |
| SOFTWARE OPTIONS | 29 |
| GENERAL OPTIONS | 29 |
| ADVANCED OPTIONS..... | 31 |

| | |
|---|-----------|
| LANGUAGE OPTIONS..... | 33 |
| CHIP BROWSER | 34 |
| CHIP AUTO IDENTIFY | 35 |
| CHIP OPTIONS | 36 |
| OFFSET OPTIONS | 36 |
| SPLIT OPTIONS..... | 37 |
| SERIAL NUMBERS | 38 |
| SERIAL NUMBER FILE | 39 |
| SERIAL NUMBER GENERATOR | 40 |
| PROJECTS41 | |
| SAVING | 41 |
| LOADING..... | 41 |
| REMOTE CONTROL OF THE PROG-EXPRESS SOFTWARE..... | 42 |
| REMOTE CONTROL USING COMMAND LINE PARAMETERS..... | 42 |
| REMOTE CONTROL USING SCRIPT FILES | 42 |
| REMOTEFILE COMMAND | 43 |
| POLL ON AND POLL OFF COMMANDS..... | 43 |
| OPEN COMMAND..... | 43 |
| MODE COMMAND..... | 43 |
| SELECTFILE COMMAND | 44 |
| PROCESSSTEP COMMAND..... | 44 |
| AUTOIDENTIFY COMMAND..... | 44 |
| RUN COMMAND..... | 44 |
| SAVELOG COMMAND | 45 |
| SAVEDEVICEINFO COMMAND | 45 |
| CLEARLOG COMMAND | 45 |
| ADDITIONAL COMMANDS..... | 45 |
| SAMPLE APPLICATIONS: PROGRAMMING OF SPECIFIC DATA | 46 |

SYSTEM REQUIREMENTS

This chapter contains the system requirements for Prog-Express and Microsoft® .NET Framework 2.0.

Prog-Express requires Microsoft® .NET Framework 2.0.

32-BIT OPERATING SYSTEMS

- Supported Operating Systems: Windows 7, Windows Vista, Windows XP Service Pack 2 or higher, Windows Server 2003, Windows 2000 Service Pack 3, Windows ME, Windows 98 Second Edition
- Prerequisites: Windows Installer 3.0 (with the exception of Windows 98/ME which require Windows Installer 2.0 or higher). Windows Installer 3.1 or higher is recommended. IE 5.01 or higher: For any installation of .NET Framework Microsoft Internet Explorer 5.01 or higher is required.
- Minimum Hard Disk Space: 300 MB (x86)

64-BIT OPERATING SYSTEMS

- Supported Operating Systems: Windows 7 - 64 Bit, Windows Vista 64 Bit, Windows XP 64-bit, Windows Server 2003 x64 Edition
- Prerequisites: Windows Installer 3.0 (with the exception of Windows 98/ME which require Windows Installer 2.0 or higher). Windows Installer 3.1 is recommended. IE 5.01 or higher: For any installation of .NET Framework Microsoft Internet Explorer 5.01 or higher is required.
- 64-Bit Support: To support 64-Bit CPU's the current 64-Bit version of Windows XP Professional or Windows Server 2003 is required.
- Minimum Hard Disk Space: 630 MB (64-Bit)

LICENSING INFORMATION FOR PROG-EXPRESS

FREWARE LICENSE CONDITIONS

LIMITATION OF WARRANTY

The software and documentation is made available for your use as is. Since the possibility of malfunctions can never be excluded even with thoroughly tested software due to the multitude of computer configurations, the author does not accept any liability for any damages that may result through the direct or indirect use of the software or documentation. Under no circumstances shall the author be held liable for any damages due to lost profits, interruption of business operations, loss of information or data and damages to other software, even if the underlying issue is known to the author. The user accepts full liability for any consequences arising out of the use of this software.

REPRODUCTION

Dissemination of the programs, publishing on CD-ROM within a shareware collection or print media as well as publishing as Bookware are only permissible with the express written permission of the author.

Manipulation, decompiling and disassembly of the software and attached files will result in a maximum 5-year prison term or fine according to § 263a German StGB. The author will file a complaint and seek compensation for damages.

OTHER

If these terms are not met the author has the right to cancel the user's license for the software.

All product names and trademarks used are hereby recognized as belonging to their owners, regardless of whether or not they are identified as such.

The law of the Federal Republic of Germany shall apply.

INSTALLATION

First, please install the Prog-Express software. You can find the software on the supplied CD or at www.batronix.com in the download area.

After installing the software, you can connect the USB Chip Programmer for the first time.

SOFTWARE INSTALLATION

This chapter contains all information that is required for the installation of Prog-Express.

To initiate setup double-click on the Setup file or use the CD-ROM auto play function.

Please follow the instructions on the screen.

The installation of Microsoft® .NET Framework 2.0 may be required as this is a prerequisite for the software. If an internet connection is available your computer will download the required files if needed, otherwise they are also available on the CD.

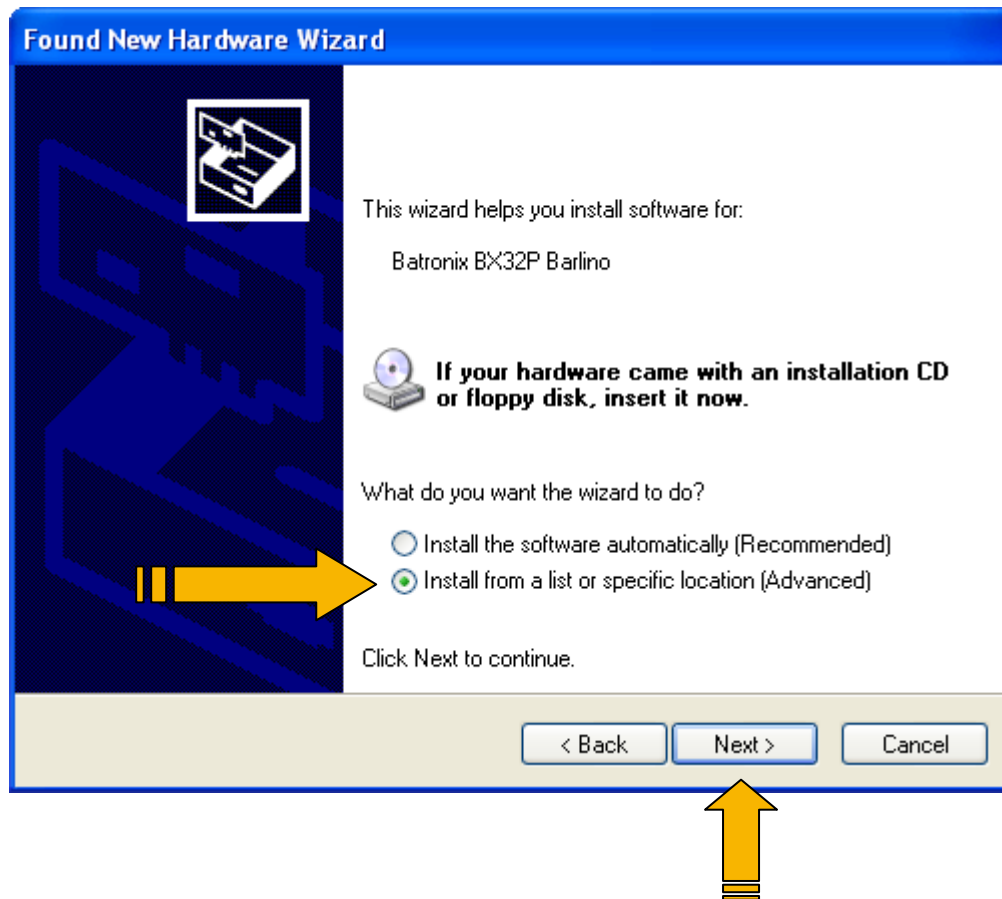
.NET VERSION X64, X86 OR IA64

If you are using a 32-bit operating system please download the x86 version, the x64 version is only compatible with 64-bit operating systems. IA64 is only required for 64-bit Intel processors running a 64-Bit operating system.

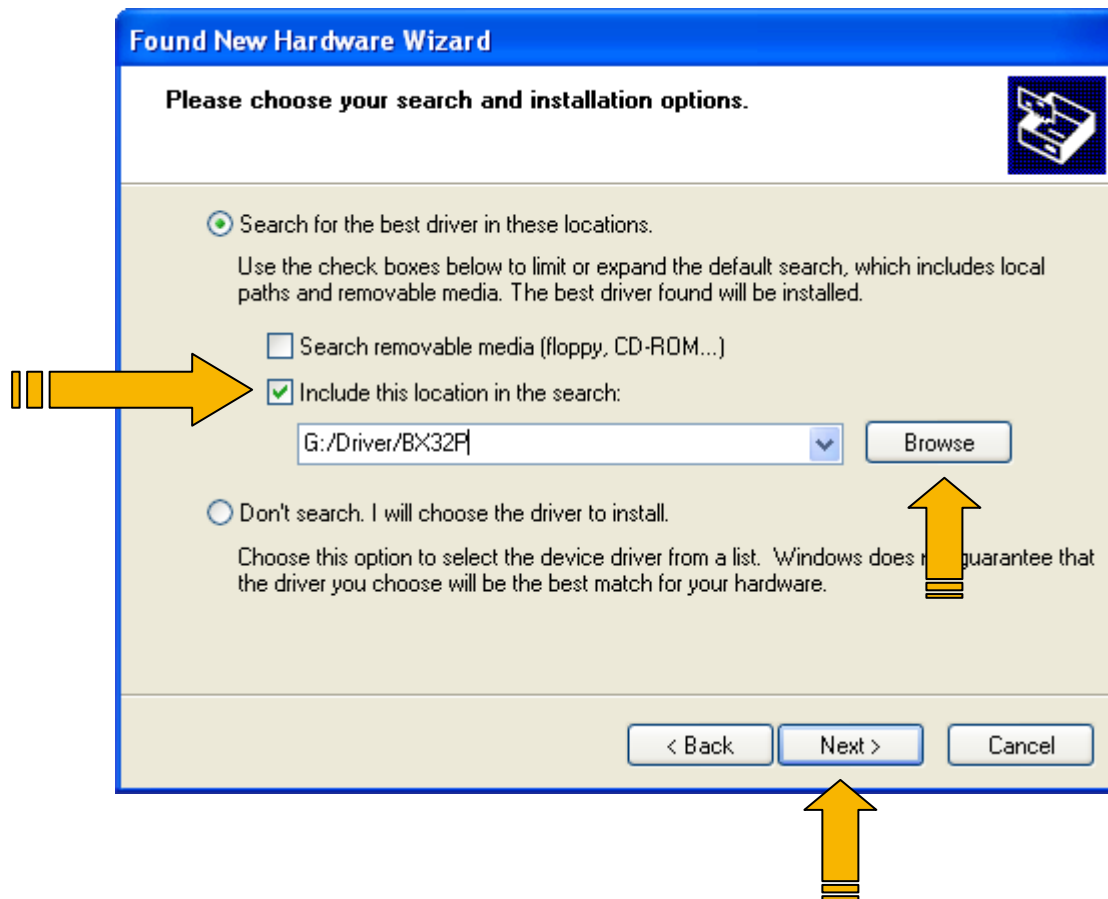
DEVICE DRIVER INSTALLATION

The Prog-Express Setup will install all Batronix USB device programmer drivers automatically. However if you need to install them manually, you can follow the steps below.

After installing the software, you can connect the Batronix USB Programmer for the first time. Windows will immediately detect the device as a new USB device and display the following message on the screen:



Please select the second item “install from a list or specific location“. Confirm this window with “Next”.

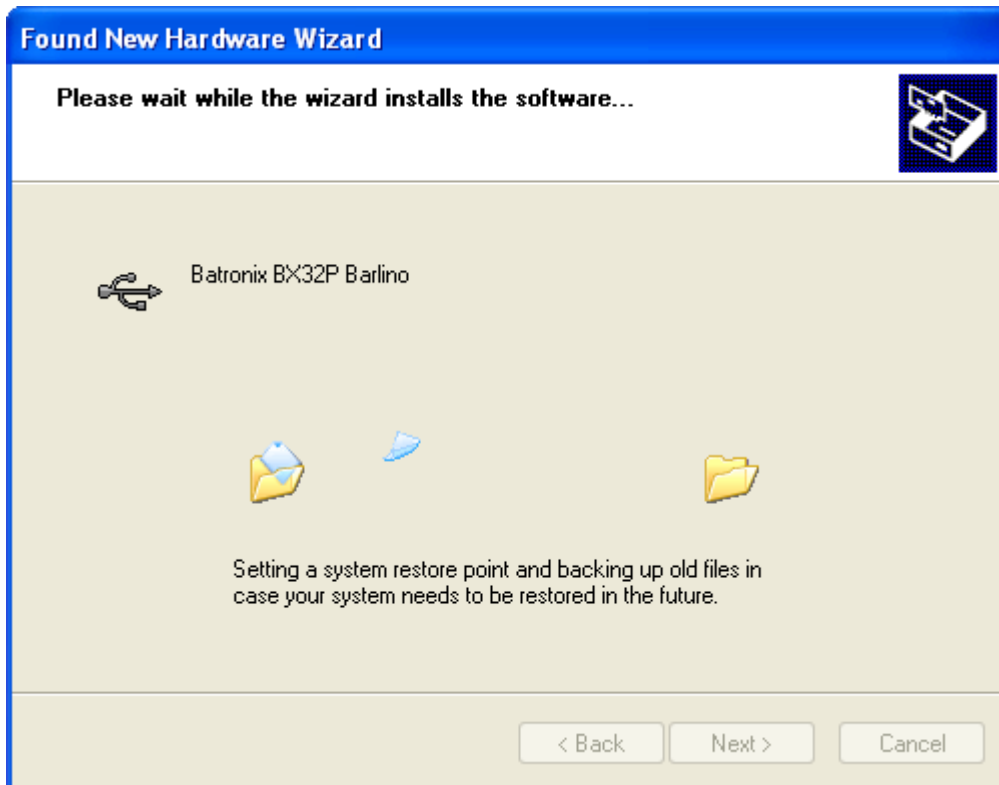


Activate “Include this location in the search” and indicate the directory belonging to your programmer within the “driver” directory of your Prog-Studio installation or the “driver” directory of the CD. For example: C:\Program Files\Batronix\Prog-Express\driver\BX32P

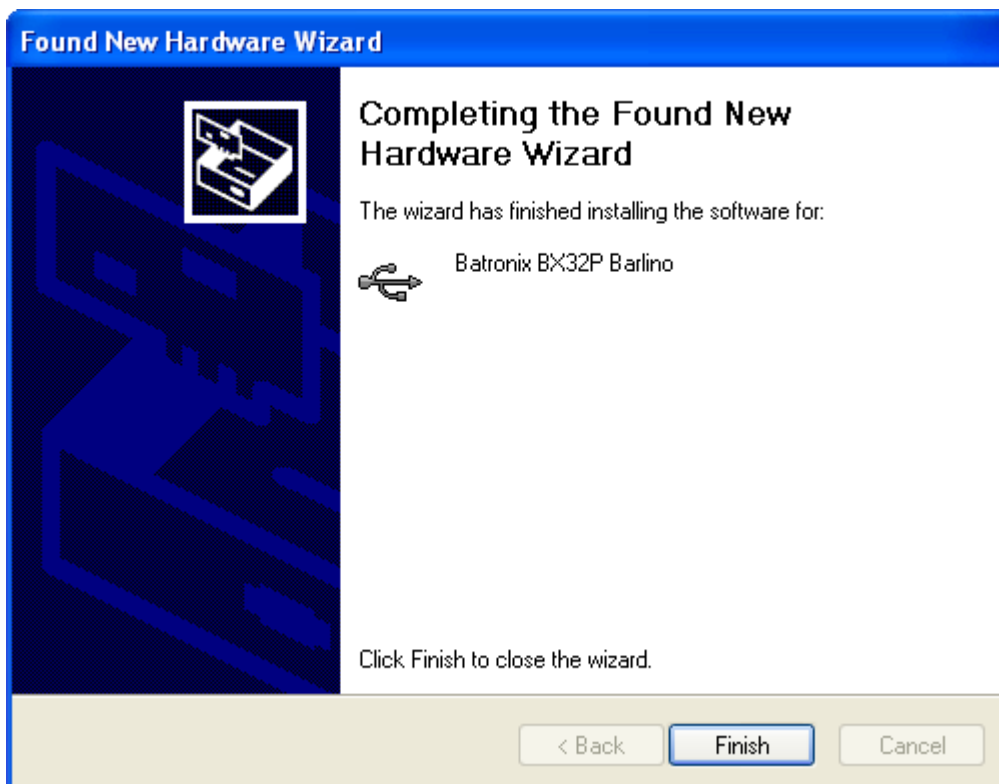
Confirm this window with “Next”.



If you get the message “Device has not passed Windows Logo testing” simply click on “Continue Anyway”.



Your PC is now searching for the indicated sources and will find the programmer. The driver will be installed.



The wizard finishes with the message "The wizard has finished installing the software for...". Confirm the message by clicking on the "Finish" button.

The driver was installed successfully, and you can now use the device.

SOFTWARE MODES

This chapter contains information about the various software modes for Prog-Express.

SUMMARY

Prog-Express has five different basic modes and an options dialog which can be picked from the selection menu. With these modes you can select the optimal interface for your application.

Following modes are available:

- Program Chip: Program one or more chips
- Copy Chip: Copy one or more chips
- Read Chip: Read data from a chip
- Production Mode: Program chips using multiple devices
- Buffer Hex-Editor: Edit data in the hex-editor
- Software Options: Edit the software options








PROGRAM CHIP

The “Program Chip” mode offers an interface optimized for programming data from a source such as a hard disk or CD to one or more chips.



Program Chip

| | | |
|---|--|---|
|  | BX32P Barlino Serial number: AD1504929 Firmware version: 02.17 | <input type="button" value="Refresh"/> |
|  | Winbond W29EE011P-90 Adapter: PLCC32-DIP32 128 KBytes (1 MBits) | <input type="button" value="Chip Auto-Identify"/> |
|  | ProgramData1MBit.bin Path: Y:\Files\ Last change: 07.01.2008 10:21:37 | <input type="button" value="Browse"/> |
|  | Chip options (optionally) No special options used | <input type="button" value="Adjust options"/> |
|  | Serial numbers (optionally) Don't insert serial numbers | <input type="button" value="Adjust options"/> |
| <input type="button" value="Start program process"/> | | Copies <input style="width: 50px;" type="text" value="1"/> |

The interface is composed of an upper section with data entry fields and process control functions in the lower section.

If you have connected several programming devices, select the device you want to use in the first field. Then select the chip you want to use in the second field. Then select the file you want to program in the third field. The chip options and serial number settings are only required in special cases and usually don't need to be turned on. Click on the blue play button to start the programming process.


The data entry fields in the upper section are described in detail in the chapter “Data Entry Fields”. General indications for process control are covered in the chapter “Process Control”.


CHIP COPY

The “Copy Chip” mode offers an interface optimized for programming data from a single chip to one or more chips.



Copy Chip

 **BX32P Barino**
Serial number: AD1504929
Firmware version: 02.17


 **Source chip: Winbond W29EE011P-90**
Adapter: PLCC32-DIP32
128 KBytes (1 MBits)

 **Destination chip: Winbond W29EE011P-90**
Adapter: PLCC32-DIP32
128 KBytes (1 MBits)

Refresh ▾

Chip Auto-Identify ▾

Chip Auto-Identify ▾

 **Start copy process**

Copies

1

The interface is composed of an upper section with data entry fields and process control functions in the lower section.

If you have connected several programming devices, select the device you want to use in the first field. Then select the chip you want to copy in the second field, and select the target chip in the third field. It is possible to select different source and target chips. For the copy to work the same way in the device as the original, both chips should have the same amount of memory, the same pin assignments and use the same control algorithms. Click on the blue play button to start the copying process.

The data entry fields in the upper section are described in detail in the chapter “Data Entry Fields”. General indications for process control are covered in the chapter “Process Control”.





Before the step “Verify dest. chip signature” the user is prompted to insert the destination chip (regardless whether or not the “Verify dest. chip signature” function is activated or not).

READ CHIP

The “Read Chip” mode offers an interface optimized for reading data from a chip and storing it to a file or viewing it in the hex-editor.



Read Chip

| | | |
|---|---|---|
|  | BX32P Barlino Serial number: AD1504929 Firmware version: 02.17 | <input type="button" value="Refresh"/> |
|  | Source chip: Winbond W29EE011P-90 Adapter: PLCC32-DIP32 128 KBytes (1 MBits) | <input type="button" value="Chip Auto-Identify"/> |
|  | ReadData.bin Path: Y:\Files\ | <input type="button" value="Browse"/> |
|  | Chip options (optionally) No special options used | <input type="button" value="Adjust options"/> |

The interface is composed of an upper section with data entry fields and process control functions in the lower section.

If you have connected several programming devices, select the device you want to use in the first field. Then select the chip you want to read in the second field.

If the process option “Save buffer data” is activated, the data is stored to a file as soon as it is read. If this option is activated a third field is available where you can specify the file path and filename.

If the process option “Show hex editor” is activated, the data is displayed in the hex editor as soon as it is read.

Click on the blue play button to start the copying process.

The data entry fields in the upper section are described in detail in the chapter “Data Entry Fields”. General indications for process control are covered in the chapter “Process Control”.

PRODUCTION MODE

The “Production” mode offers an interface optimized for programming chips simultaneously on multiple programming devices.



Production Mode

Winbond W29EE011P-90
Adapter: PLCC32-DIP32
128 KBytes (1 MBits) **Chip Auto-Identify**

ProgramData1MBit.bin
Path: Y:\Files\
Last change: 07.01.2008... **Browse**

Chip options (optionally)
No special options used **Adjust options**

Serial numbers (optionally)
Don't insert serial numbers **Adjust options**

Start production process Copies: 1

Programmer

Refresh

BX32P Barlino (AD1504929)
Waiting for new job...

BX32P Barlino (JK0504152)
Waiting for new job...

BX40 Baqero (AD1504925)
Waiting for new job...

BX40 Baqero (AD1504927)
Waiting for new job...

Log text Programmer

In this mode up to eight USB Chip Programmers and eight of the Professional Programmer Series (BX-Programming Device) can be controlled at one time. All recognized devices are displayed in the device selection list (right part of the picture).

The individual programming devices are controlled independently so that a high level of productivity can be attained. The programming speed for each individual programming device is almost as quick as it is when using only a single device.

If a hub is used the USB connection of the PC and the hub must support the USB 2.0 High Speed mode so that the data transfer rates are not limited unnecessarily.

The data entry fields in the upper section are described in detail in the chapter “Data Entry Fields”. General indications for process control are covered in the chapter “Process Control”.

In production mode the right side of the screen shows the programming device summary. A toolbar is displayed at the top of the screen. This allows you to activate settings for chips, files, and options individually for each programming device. If a setting which is specific to a particular programming device is activated the corresponding button is highlighted in orange.



Use the refresh button to refresh the list of programming devices which are currently connected.

Information about the device and an additional five or six buttons are displayed for each connected programming device.



Click on this button to add or remove a programming device from the production process.

If the button is highlighted in orange the programming device is included in the production process.



To find a particular programming device on the workbench among several other programming devices, all it takes is clicking on this button. The green operation light on the respective device will flash briefly.



Use this button to select a separate chip for each individual programming device. In this way you can, for example, use four programming devices to program four different chips within one production process.



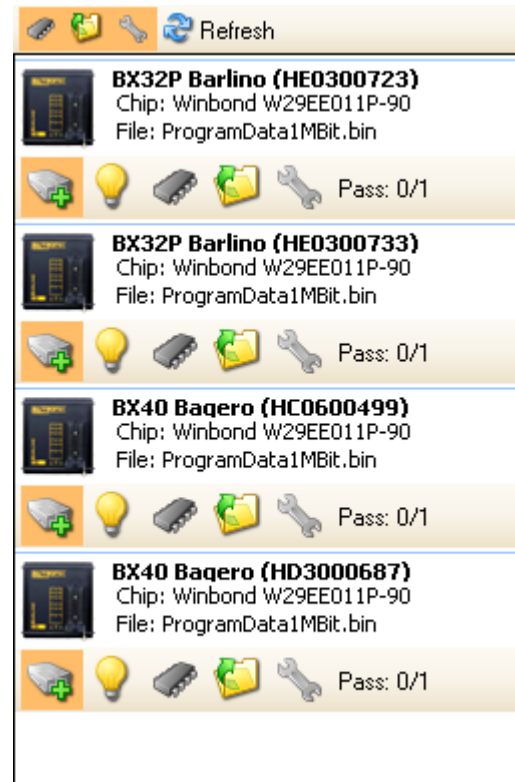
Use this button to select a separate file for each individual programming device. In this way you can, for example, use four programming devices to program four different files within one production process.



With this button you can select different options (offset, split, etc.) for each individual programming device. In this way you can, for example, use two programming devices with different split settings (odd, even) within one production process.



If the desired chips don't contain a signature, the insertion and removal of these chips cannot be detected automatically. The process step "Wait for chip" can not be activated for these chips. Therefore, the process sequence for the respective device must be started using the device's start button after inserting the chip.



A symbol indicating the corresponding status for each programming devices is shown in the upper right-hand corner.



The device is waiting for a chip to be inserted.



The inserted chip is being erased.



The chip is being checked to see if it is blank.



The chip is being programmed.



The programmed data is being verified.



The chip is being write-protected.



The device is waiting for a chip to be removed.

DATA ENTRY FIELDS

In the modes “Program Chip”, “Chip Copy”, “Read Chip”, and “Production Mode” a number of data entry fields are available in the upper section of the screen.

PROGRAMMER



The serial number and firmware version of the selected programming device is displayed below the device name in this field. Clicking on the left area of this button makes the green LED on the selected programming device blink several times. This function is useful when several programming devices are connected if you need to identify a particular device.

Clicking on the arrow on the right-hand side opens a list which shows all connected programming devices. When you open the selection list the currently connected programming devices are detected and the list is refreshed. The desired device is selected by clicking on it in the selection list.

SOURCE CHIP AND DESTINATION CHIP

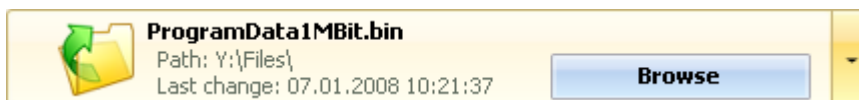


In this screen the name of the chip and the size of its memory are displayed, as well as the name of the suitable adapter if one is required. Clicking on the left area of the button opens the Chipbrowser which allows you to select a chip.

Using the “Chip Auto-Identify” button allows a chip to be selected automatically using its signature (see chapter “Chip Auto Identify”).

Clicking on the arrow on the right-hand side opens a list which shows the last 10 chips which were used. The desired chip is selected by clicking on it in the selection list.

SOURCE FILE



Use the “Open file” data entry field to select the file to be loaded. The path of the file and the last time the file was changed are displayed below the filename. Clicking on the left area of the button opens the file browser which allows you to select a file.

Clicking on the arrow on the right-hand side opens a list which shows the last 10 files which were used. The desired file is selected by clicking on it in the selection list.

SAVE FILE



In the “Save file” data entry field you can specify where and with what filename the chip data should be saved after it is read. The current contents of the buffer are saved using the “Save buffer data” process step.

Clicking on the arrow on the right-hand side opens a list which shows the last 10 files which were used. The desired file is selected by clicking on it in the selection list.

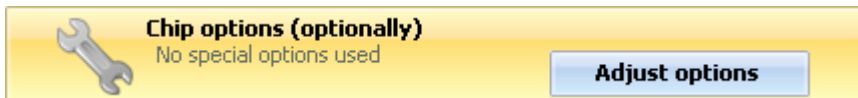
In the “Save File” field you can use the following special characters which will be dynamically replaced:

! The exclamation mark will be replaced by the name of the selected chip.

The lozenge will be replaced by the number 1 or higher. If there is a file with this name already, the number will be increased by one until there is no file with this name.

Example: You read the data from an AT27C010 and enter the filename “File-!-#.bin” in the “Save File” dialog. The data will be saved with the filename „File-AT27C010-1.bin“. If you read and save the same chip again the data will be saved with the filename “File-AT27C010-2.bin“.

CHIP OPTIONS



Special chip options can be used to modify the data to be burned or to use special functions in the chips. If you are not sure what these settings are used for or if you simply want to write data to a chip, do not turn the chip options on.

Clicking on this button brings up a screen with chip option settings. More information can be found in the chapter “Chip Options”.

SERIAL NUMBER

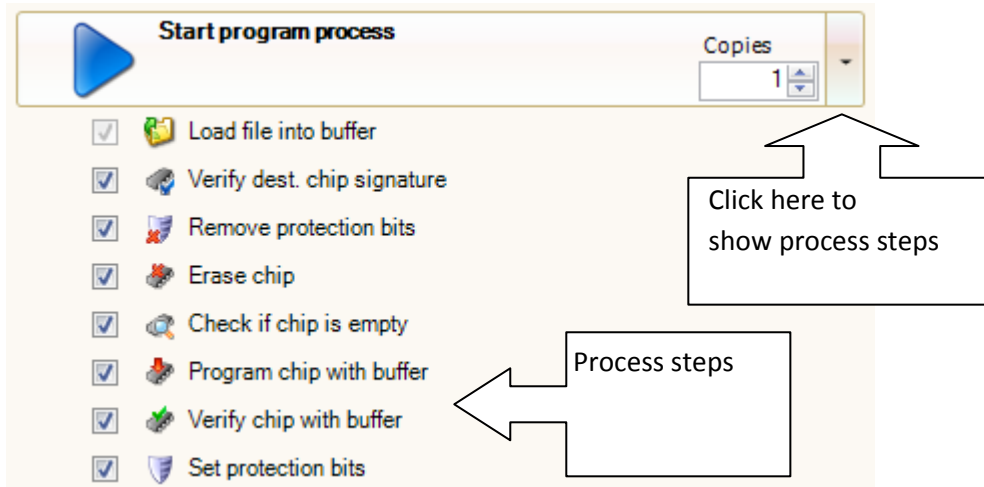


For production purposes serial numbers can be added to the data before it is written.

Clicking on this button brings up a screen with serial number settings. More information can be found in the chapter “Serial Numbers”.

PROCESS CONTROL

The process control is similar for the software modes “Program Chip”, “Chip Copy”, “Read Chip” and “Production” and it is described in this chapter. Each of these modes has a blue bar with a “Play” symbol (blue triangle).





Clicking on the arrow on the right-hand side opens a list which shows all steps of a process. When a process is started this list is opened automatically.

Every process is composed out of individual process steps. These process steps can be activated or deactivated using the respective checkbox. A process step can be executed individually and independent of the overall process by clicking on the button for that process step.

The data entry field “Copies” can be used to specify how many times the process should be executed. That is, for the “Programming” mode this specifies how many chips should be programmed, and in the “Copying” mode it specifies how many copies of the source chip should be made.

CONTROLS


 The selected process steps are carried out in order from top to bottom after the process is started by clicking on this button.

 Clicking on this symbol (only shown while a process is running) stops the process and no further chips are programmed.

SPECIAL CONTROLS (ONLY IN “PRODUCTION-MODE“)

Clicking on this symbol (only shown while a process is running) stops the process and no further chips are programmed. However the currently active process step will be completed.



This symbol will appear after clicking on the stop button and only in the production mode. It indicates  that the normal process sequence was stopped and only the currently active process step is being completed. Clicking on this button will terminate all currently running processes immediately.

PROCESS STEPS



Load file into buffer

The file specified in the “Open file” field is loaded into the buffer during this process step.



Save buffer to file

Data from the buffer is saved to a file. Non-existing files are automatically created and existing files are overwritten.



Verify source / dest. chip signature

The signature of the inserted chip is compared to the setting specified for the “Source Chip” / “Destination Chip”. If a discrepancy is found, a dialog with the options “Cancel Process”, “Re-Check Signature”, and “Ignore discrepancy and continue process” is displayed.



Erase chip

This process erases the chip in the programming device. Typically erasing a chip means that all bits are set to 1. Therefore all bytes in an erased chip are at FFh (Byte FFh = 11111111).

Not all chips can be erased using a programming device. For example, the 27c EPROM's with glass window can only be erased using intense UV-C light in a special EPROM eraser. 27c EPROM's without a glass window cannot be erased and therefore cannot be programmed with new data. These chips are often marked as OTP = One Time Programmable.



Check if chip is empty

This checks if the chip is erased, i.e. if all bits are set to 1. During this process the entire chip contents are read and all bits are checked.



Read chip data into buffer

The source chip is read and its data are loaded into the buffer. Data in the buffer can be viewed and edited using the Hex-Editor (see chapter “The Hex-Editor”).



Program chip with buffer

The data in the buffer is written to the chip, using any “Programming Options” and “Serial Number Settings” that have been set.



Verify chip with buffer

The complete chip contents are read and all bytes are compared with those in the buffer.



Set protection bits

This sets so-called “Protection Bits” which prevent the chip from being accidentally overwritten at a later date. This function is not supported by all chips.



Show Hex-Editor buffer

This will switch the software to the Hex-Editor Mode.



Auto wait for chip inserted

The corresponding programming device waits until a new chip is inserted. When this occurs the chip signature is checked.

If the desired chips don't contain a signature the removal and insertion of these chips can not be detected automatically. The process function “Auto-Wait for Chip inserted” can not be activated for these chips. In this case the process is started via the start button on the corresponding device.



Auto wait for chip removed

The corresponding programming device waits until the programmed chip is removed.

When the process is completed the operating light on the devices blinks to indicate that the chip can be removed. The corresponding programming device waits until the chip is removed. If the green operating LED blinks after the chip is removed this indicates that the device is waiting for the next chip to program. If the green operating light does not blink this indicates that no further chips are set to be programmed by this device.

If the desired chips don't contain a signature the removal and insertion of these chips can not be detected automatically. The process function “Auto-Wait for Chip inserted” can not be activated for these chips.

This symbol is shown beside the process step if this step is not supported by the selected chip or



programming device.

LOG TEXT

Within the log text all details for already finished and running process will be recorded

With a mouse click on a plus or minus character, the nodes can be expanded and reduced. After a new process started, the software will reduce the last process node.

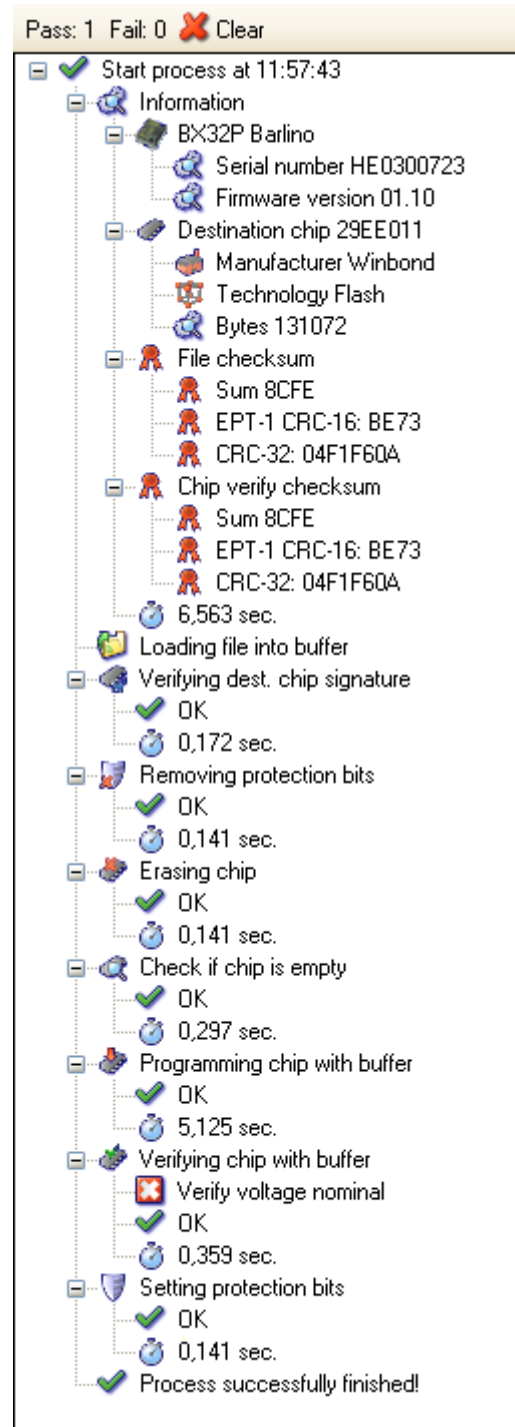
For each process there is an information node and one node for each used process step in the process. The information node contains general information about the process like the used programmer, the chip and file and chip checksums.

The file checksum can be different from the chip checksum for several reasons. The file can be bigger and contain more data than the chip if you choose to program only a part from the chip. It can also be different if you use special programming options or serial numbers. The “Verify chip with buffer” function ensures the correct programming. Generally we can say that the Prog-Express “Verify chip with buffer” function is much more secure than comparing the file checksum with the chip checksum as the verify function compares all bytes exactly and not only compares the checksums of the data.

Within the information node there is also the complete process time including the user time for confirming messages / etc... The time for each process step is listed in the process step nodes.

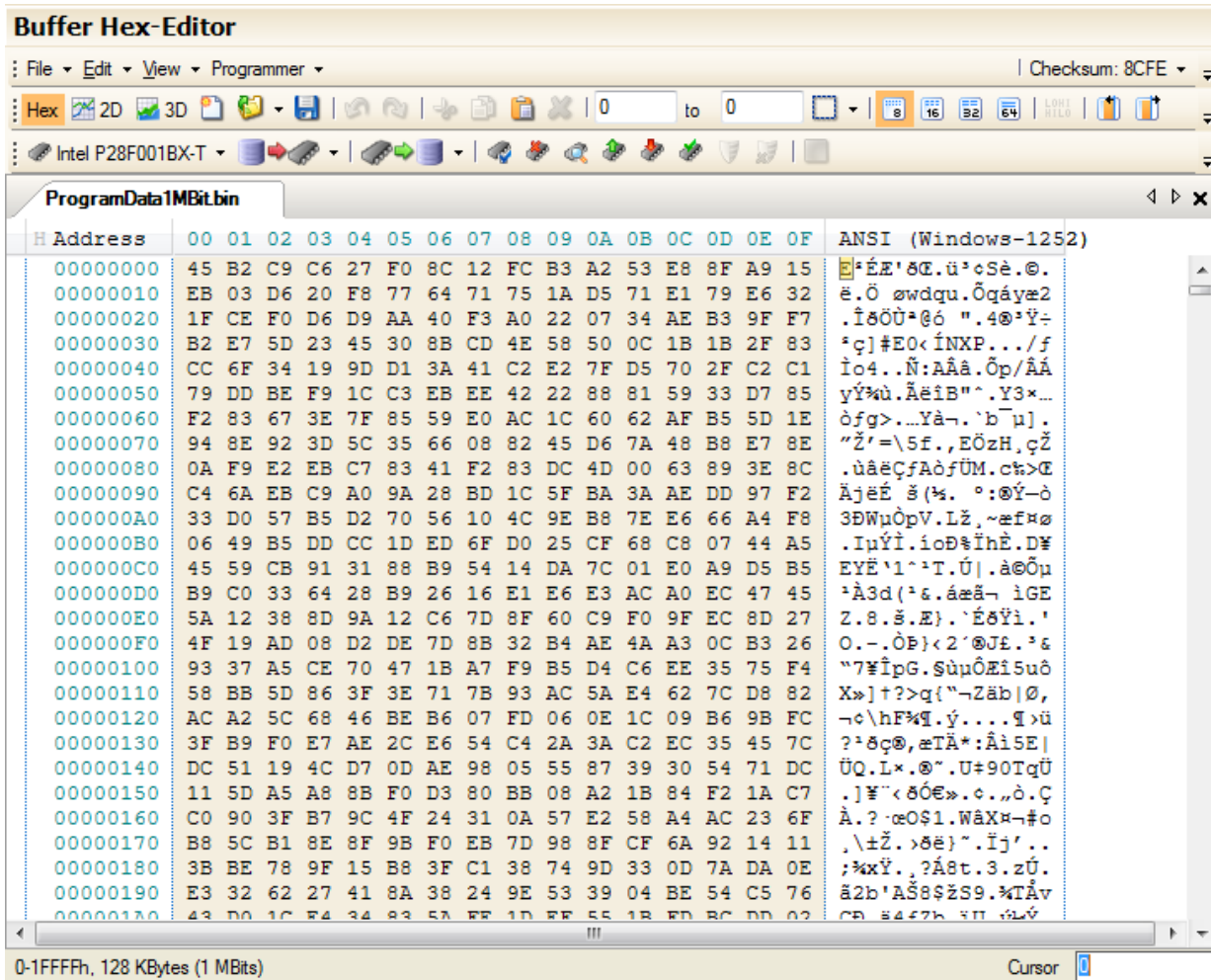
Aborted processes are marked with a red stop sign, failed processes are marked with a red X, and processes which completed successfully are marked with a green checkmark.

With a click on the red X the whole Log text can be cleared.



HEX-EDITOR

The Hex-Editor is used to view and edit binary data. It includes comprehensive display functions as well as many easy-to-use data editing functions.









The editing functions can be selected via toolbar buttons or from the context menu. The context menu can be accessed via the right mouse button.

















The currently marked address is shown below the status display when you move the cursor. Clicking on the address display allows you to enter a specific address which is then displayed.

When the programming device functions are used the current process step is shown in the status display, and the current progress is shown in the progress indicator (bottom right).


HEX-EDITOR MENU AND TOOLBAR

This toolbar contains functions for navigating and manipulating the data in the Hex-Editor.

-  Opens a new blank Hex Editor window
-  Opens an existing file. A list of the last 10 files used can be opened by clicking on the arrow to the right of this button.
-  Imports an existing file with offset and/or additionally to Hex-Editor data
-  Compares Hex-Editor data to data from an existing file
-  Saves the current buffer contents
-  Saves the data under a new filename.

File format: The file format can be detected automatically (preset) or specified manually. The setting is used for all Hex Editor opening and saving functions.
-  Cuts the selected data
-  Copies the selected data
-  Pastes copied data at the cursor position
-  Erases the selected data
-  Opens the search and replace dialog
-  Contains a list of functions for the selected data area
-  Rolls back the last operation (undo)
-  Rolls forward the last operation that was rolled back (redo)
-  Reduces the number of bytes displayed in one row of the editor
-  Increases the number of bytes displayed in one row of the editor
-  Shows the data in byte format
-  Shows the data in word format
-  Shows the data in double-word format
-  Shows the data in quad-word format
-  Switches between the ANSI and bar display modes.
-  Jumps to the first byte in the data that is not FFh

 Jumps to the next byte in the data that is not FFh

 Jumps to the last byte in the data that is not FFh

HEX EDITOR SHORTCUT KEYS

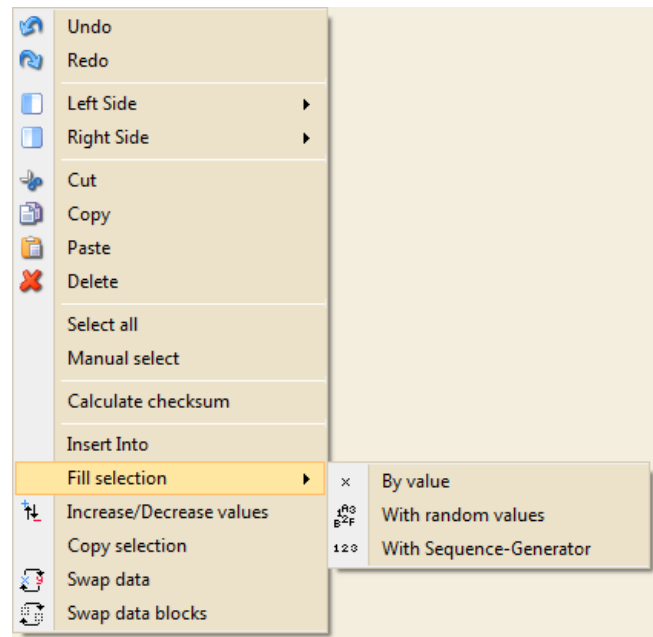
The Hex Editor supports the following shortcut keys:

- <Page Up>: Moves to the first line of the currently displayed page. If the cursor is already on the first line the display moves up by one whole page.
- <Page Down>: Moves to the last line of the currently displayed page. If the cursor is already on the last line the display moves down by one whole page.
- <Home>: Jumps to the first byte in the current line.
- <CTRL+Home>: Jumps to the first byte in the Hex Editor.
- <End>: Jumps to the last byte in the current line.
- <CTRL+End>: Jumps to the last byte in the Hex Editor.
- <Insert>: Switches back and forth between overwrite and insert modes. In overwrite mode (standard setting) the data at the current cursor position is overwritten when new data is entered. In insert mode new data is inserted without overwriting existing data.
- <CTRL+A>: Selects all data.
- <Shift+Cursor key>: Expands the selection.
- <CTRL+X>: Cuts the selected data and copies it to the clipboard.
- <CTRL+C>: Copies the selected data to the clipboard.
- <CTRL+V>: Pastes data from the clipboard.
- <Delete>: Deletes the selected data.
- <Tab>: Switches back and forth between HEX and ANSI modes.

HEX-EDITOR CONTEXT MENU

You can open the context menu by clicking with your right mouse button onto the Hex-Editor.

Here you can find the basic functions such as “Copy & Paste” or “Undo & Redo”. The “Selection” menu expands to show further functions that can be applied to the current selection.



WORKING WITH SELECTED DATA

These options can also be selected via the selection box in the toolbar of the Hex-Editor as well as within the context menu.

MANUAL SELECT

Here you can specify exactly what data should be selected.

CALC SELECTION CHECKSUM

Calculates a checksum value for the selected range using a selectable algorithm (sum, MD5, SHA-1, EPT1 CRC16 or CRC32).

FILL SELECTION - BY VALUE

Fills the selected area with a specified value

FILL SELECTION – WITH RANDOM VALUES

Fills the selected area with random values

FILL SELECTION – WITH SEQUENCE-GENERATOR

Fills the selected area with a specified range of values

INCREASE/DECREASE VALUES

Increases or decreases all values in the selected area by a specified amount or percentage value.

COPY SELECTION

Copies the selected data to a specified address. You can select whether the target range should be expanded or overwritten.

SWAP DATA

Depending on the setting, switches the first and second byte/word/double word or quad word values with each other. If more than one pair is selected the switching is continued by the same method (for example, byte 1 is switched with byte 2, byte 3 with byte 4, etc.).

SWAP DATA BLOCKS

Switches the selected data with data starting at a specified offset address.

CHIP PROGRAMMING IN THE HEX-EDITOR

All basic functions required to program a chip are also available in the Hex-Editor. Use the left drop-down box to select a chip to work with. Here you can also access the Chipbrowser to simplify chip selection.



Programmer selection



Makes the operation light on the selected device flash so the device can be identified



Chip selection



Opens the Chip Browser



Chip Auto-Identify (see chapter “Chip Auto Identify”)

Single process steps:



Verifies the chip signature



Erases the chip



Verifies that the chip is erased



Reads the chip data into the buffer



Burns the buffer data to the chip



Compares the buffer data to the chip data



Set protection bits



Clear protection bits



Stops the current programming process (does not apply to running processes in other modules!)



Starts a complete programming process. The process default settings contains the process steps “Verify chip signature”, “Remove protection bits”, “Erase chip”, “Check if chip is empty”, “Program chip with buffer”, “Verify chip with buffer” and “Set protection bits”. Process steps which are not supported by the chip will be automatically deactivated. With a click on the black arrow you can open the process step list and activate or deactivate process steps.



Starts a complete reading process. The process default settings contains the process steps “Verify chip signature”, “Read chip data into buffer” and “Verify chip with buffer”. With a click on the black arrow you can open the process step list and activate or deactivate process steps.

SOFTWARE OPTIONS

The software options can be used to configure the behaviour when Prog-Express is opened and closed, the display of messages, the playback of audio files, special programming device options, the operator mode and the language settings.



The software options are displayed on four separate pages, “General”, “Advanced”, “File Associations” and “Language”.

GENERAL OPTIONS

Software Options

General
Project
File Associations
Language

Prog-Express start options

Auto load last settings

Auto load last used project

Auto load the following project:

Browse

Prog-Express exit options

Auto save project file

Working directory

Start browser in working directory

Browse

Automatic software updates

Search for online software updates at start up

Messages

Show warning if the buffer data is bigger as chip size at programming

Sound

Play sound file when process finished successfully:

Sounds\Plop.wav
Browse
Test

Play sound file when process failed:

Sounds>Error1.wav
Browse
Test

PROG-EXPRESS START OPTIONS

The software can load the last used project or a specified project directly at start up. A project contains the selected mode, the selected chips, programming options, serial number options and the activated process steps.

PROG-EXPRESS EXIT OPTIONS

When exiting Prog-Express the software can automatically save the actual settings into the last loaded project file (default).

WORKING DIRECTORY

The software can start the project browser always in a specific directory. If this option is not used, the browser starts in the last used directory.

AUTOMATIC SOFTWARE UPDATES

The software can check for updates online when you run it. If an update is available it can be downloaded and installed automatically.

MESSAGES

The display of separate messages can be switched in or out here.

SOUND

After a process finished or after a programming error the software can play a sound file. Some files come with the Prog-Express software in the sub folder "Sounds". You can also select your own .wav sound files from your PC.

ADVANCED OPTIONS

Software Options


General
Project
File Associations
Language

Operator mode

Lock mode selection also

Lock options dialog and operator mode by password

Password:

 Click onto the lock symbol at the Prog-Express title bar to activate the operator mode.

Data handling

Fill unused bytes with:

hex

Mirror buffer data on higher free chip address areas:

Activate this option if you want to program the data from a chip onto a larger replacement chip.

Byte order at 16 bit chips:

Program lower byte before higher byte (LSB-MSB, standard)

Program higher byte before lower byte (MSB-LSB)

Multi-Pass verification (BX40 & BX48 only)

Verify voltage minimal

Verify voltage nominal

Verify voltage maximal

Security- and test functions

Test pin contacts (BX48 only)

Overcurrent monitoring enabled (BX48 only)

OPERATOR MODE

With activating the operator mode the software is protected against inadvertent or wanted changes. This operator mode is recommended in productions environments, where a software engineer adjusts all settings and an operator uses it to run the series production.

While the operator mode is activated, the file, the selected chips, programming options, serial number settings and the activated process steps are protected against changes. In the software options the mode change can also be disabled.

The operator mode can be activated and deactivated with a click onto the lock symbol on the Prog-Express title bar. If there the password protection is enabled, the software asks for the password before it activates or deactivates the operator mode.

DATA HANDLING

Several special settings can be specified in this area.

Unused bytes are bytes for which no data is defined. This can happen when, for example, a smaller file is programmed into a larger chip.

The byte sequence is only applicable to programming of 16-bit chips. Here you can specify in which order the bytes from the file are used.

MULTI-PASS COMPARISON

The BX40 supports a multi-pass comparison. During this process the chip data are read and compared several times with different supply voltages. If the process finds differences in the data the comparison is aborted and an error is displayed.

The voltages used are dependent on the allowable operating voltage range of the chip. For example, many chips allow for a voltage range of $\pm 10\%$. In this case if, for example, the nominal voltage is 5 volts, the comparison can be carried out at 4.5, 5.0, and 5.5 volts. The repeated comparison of the programmed chips provides a higher level of program data assurance. This eliminates chips which are not 100% programmed and could thus fail in the end device under strongly fluctuating voltage conditions.

LANGUAGE OPTIONS



The desired language can easily be selected by clicking on it. If applicable, download the latest version of Prog-Express from our website www.batronix.com so that you can select all currently supported languages.


The first entry “automatic” uses the language settings of your operation system. The following languages are sorted alphabetically depending from the English spelling of all languages.

CHIP BROWSER

The chip browser can be activated from many locations in the program, for example the chip selection for the target chip in the software mode “Program Chip”.

The chip browser has various functions to help find and select the correct chip among all the different possible chips.

The chips are organized in a tree structure and it is possible to determine the depth of this structure using the following options:

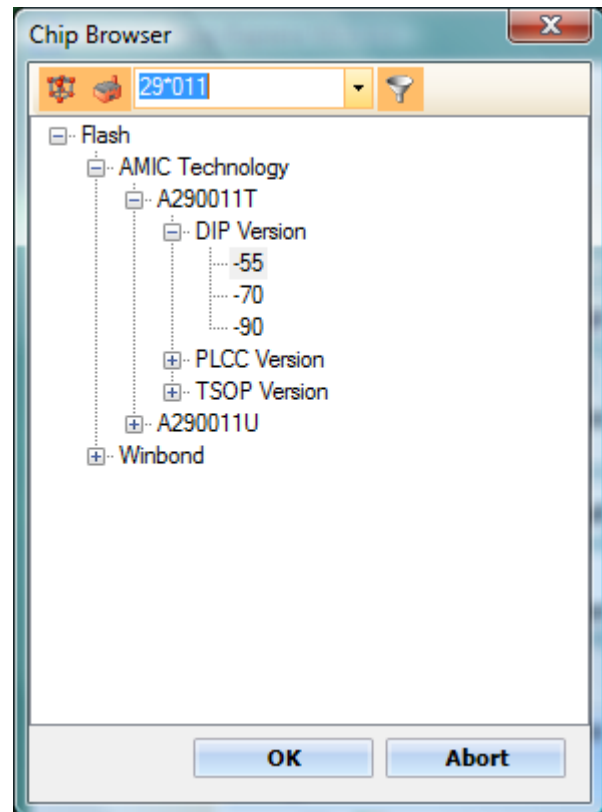
 Group chips according to technology.

If this option is activated all chips are grouped according to technology and it is easy to differentiate between, for example, flash chips and EPROM's.

 Group chips according to manufacturer.

If this option is activated all chips are grouped according to manufacturer.

Both of these options can be activated at the same time. Chips are grouped first by technology and then by manufacturer.



THE FILTER OPTION

Search criteria can be entered in the text field, and then by activating the filter option only chips that contain the search criteria in the name are shown.

Four different wildcards can be used for filtering.

% This symbol can represent any type and number of characters.
For example, filtering for “27%512” returns results like „27512“ / „27c512“ / „27SF512“.

* The star has the same functionality as the % symbol.

_ An underscore can replace a single character. For example, filtering for “27_512” returns results like „27C512“ and „27E512“ but no results like „27512“ or „27SF512“.

? The question mark has the same functionality as the underscore.

CHIP AUTO IDENTIFY

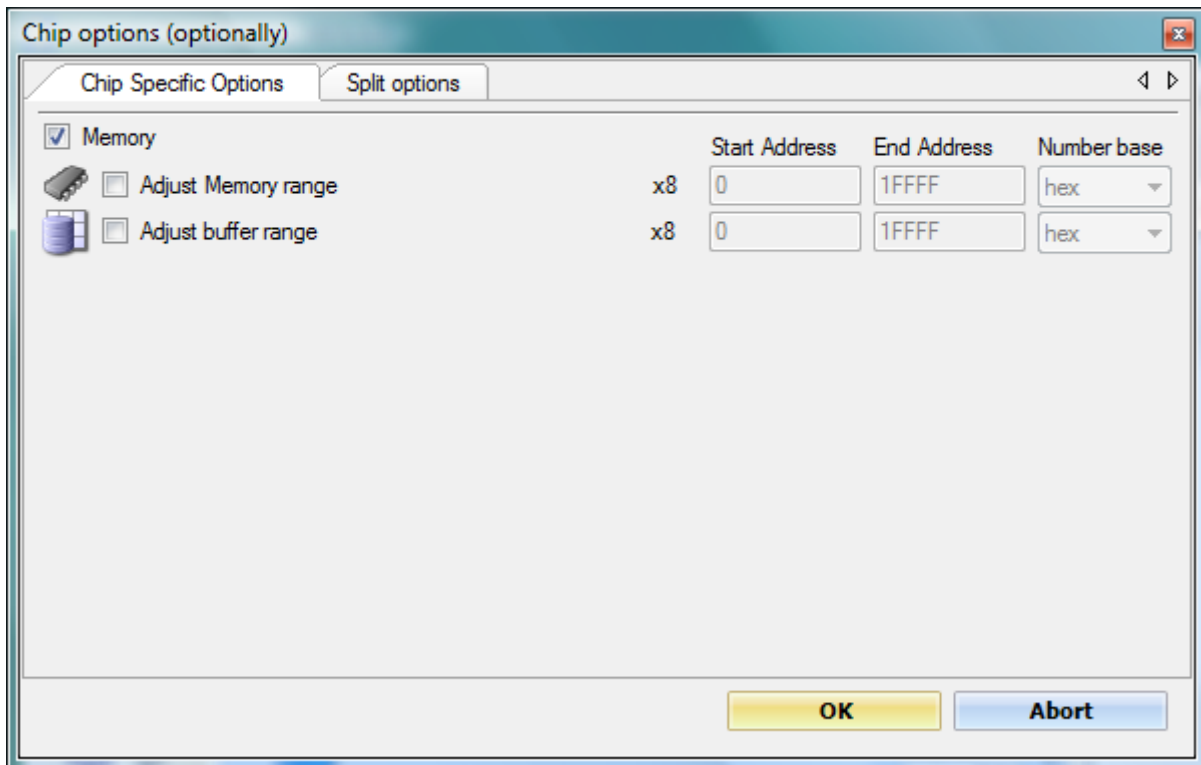
Most chips contain a signature that can be read via special functions. This signature is usually composed of both a manufacturer and a chip identifier combined, with which a chip type can be clearly identified. There are also some chips that don't contain a signature and can therefore not be automatically identified by the software.

Please note that many chips require a relatively high voltage (12.5V) on address line A9 to read this signature. This voltage is sufficient to damage other chips that cannot handle this voltage at the respective pins.

Prog-Express uses the auto-identify feature automatically depending on process settings.

CHIP OPTIONS

OFFSET OPTIONS



MEMORY

Some ICs have more than one memory area. For example several microcontrollers have a separate program memory, a data memory and a configuration memory. For each memory you will find separate adjustment controls in this tab.

ADJUST MEMORY RANGE

In default setting, the whole range of the memory will be used. If you want to specify a partial range, please check the “Adjust Memory range”. The used range can be set with the “Start Address” and “End Address”. Only memory addresses within this range will be programmed, read and verified.

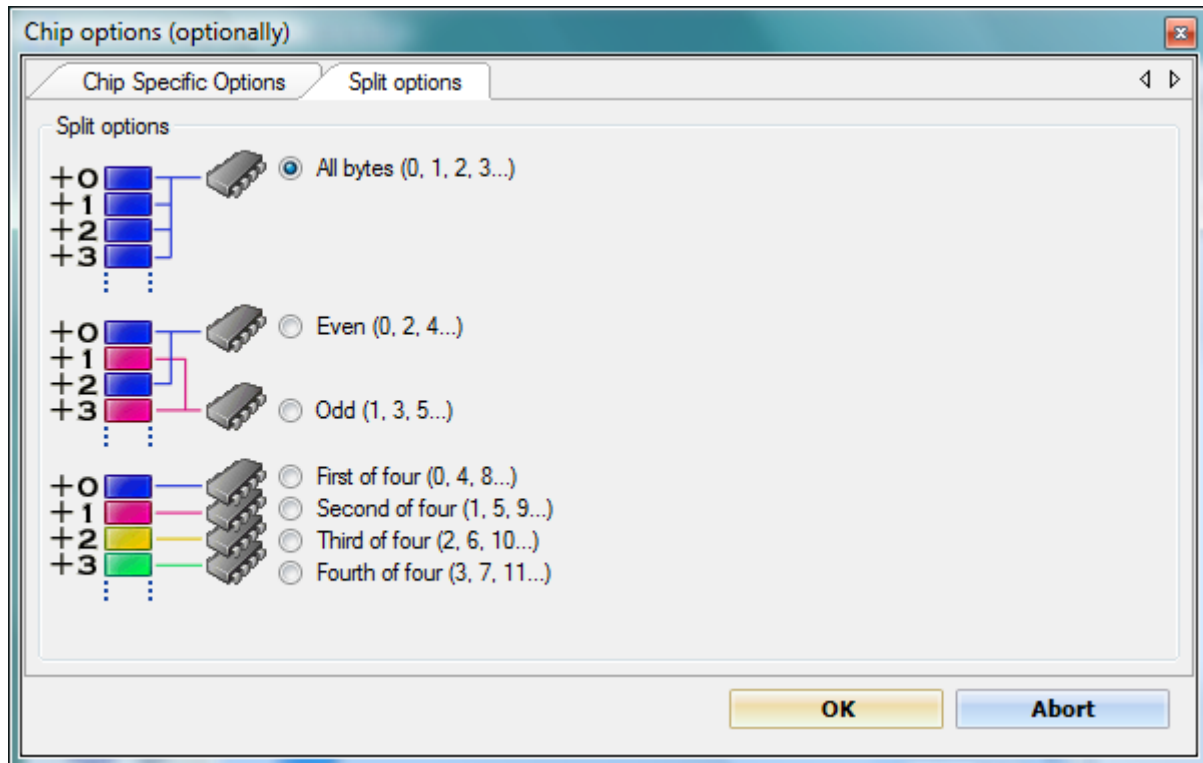
ADJUST BUFFER RANGE

Usually the data will be used starting from the first file (buffer) address. If required you can specify a “Start Address” as offset here. The offset information is based on the data width of the chip. For a 16-bit chip (2 bytes of data per address) the first 4 bytes are skipped if you use an offset address of 2.

SPECIAL CHIP OPTIONS

If the chip has special chip options like a SEEPROM serial address or configuration settings, they will be displayed here also.

SPLIT OPTIONS



Three different split-functions are possible:

1. No Split Function: All data is programmed in the chip in the normal manner.
2. Split according to even/odd address: When using the setting “Even” all buffer data with even addresses are programmed into the chip, using the setting “Odd” all data with odd addresses is programmed.
3. Split into four address sections: Here you can select which address locations in each block of four is to be programmed. Each first, second, third, or fourth address out of every four sequential addresses can be selected.

SERIAL NUMBERS

For chip production it is often necessary for each chip to have a unique serial number or address. For this purpose the following settings are available.

Serial numbers

Don't insert serial numbers

Load serial numbers from file

Current position: 1

Use serial number generator

| | |
|---|--|
| <p>General</p> <p>Chip address for the first serial number byte 0 hex</p> <p>Number of serial number bytes 0 dec</p> <p>Next serial number 0 dec</p> <p>Hexcode: 30 30 30 30 30 30 30 30 ASCII: 00000000</p> <p>Increment each step by: 0 hex</p> | <p>Endian</p> <p><input type="radio"/> Little endian (L-H)</p> <p><input checked="" type="radio"/> Big endian (H-L)</p> <p>Number base</p> <p><input checked="" type="radio"/> Decimal (dec)</p> <p><input type="radio"/> Hexadecimal (hex)</p> <p>Number format</p> <p><input checked="" type="radio"/> Text (ASCII)</p> <p><input type="radio"/> Binary (bin)</p> |
|---|--|

OK Abort

SERIAL NUMBER FILE

Under “Load Serial Numbers from File:” a serial number file can be specified, from which the desired serial numbers are then read. After each successful burn process the next line is read from the file and used for the next burn process.

The serial number file must have the following format:

Lines starting with “#” are comment lines and will be skipped. No comments are allowed in lines with serial numbers.

Spaces and tabs are allowed between the individual values.

Serial number lines must be formatted as follows: The line must start with a hexadecimal address followed by a colon and then a list of comma-separated hex values.

EXAMPLE:

```
#Testdata...
#This comment is allowed
1A0h:1, 2, 3, 4, 5, 6, 7, 8, 9
1A0h:11,12,13,14,15,16,17,18,19           #This comment is not allowed
1A0h:21h, 0x22, 23 , &H24 ,25, 26, 27 , 28 , 29
1A0h:*Line with errors*...
```

The hexadecimal values can be entered in various formats.

4E,10,F2 : simple hexadecimal without additions

4Eh,10h,F2h : simple hexadecimal with h as suffix

0x4E,0x10,0xF2 : hexadecimal number with 0x as prefix

&h4E, &h10, &hF2 : hexadecimal number with &h as prefix

It is also possible to enter ASCII data, however these have to be converted to their hexadecimal equivalents. For example if a list of MAC addresses (hardware addresses of computer network cards) is listed this would look as follows:

The Address is: 7F-3C-01-47-FF-04 and should be stored in the chip starting at 05A2h :

```
7 F - 3 C - 0 1 - 4 7 - F F - 0 4
05A2h: 37,46,2D,33,43,2D,30,31,2D,34,37,2D,46,46,2D,30,34
```

Each serial number is written to the chip starting at the specified address for the length of its values.

SERIAL NUMBER GENERATOR

The generator produces serial numbers according to its settings. Following settings are possible:

Chip address for the first serial number byte: The first byte of the serial number is stored at this address. Following bytes in the serial number are stored to sequential locations from this address.

Number of serial number bytes: The length of the serial number in bytes. The setting 8 results in 8-digit serial numbers.

Next serial number: The serial number that will be used for the next programming cycle. After each successful programming cycle this is incremented by the "Increment each step by" value.

Increment each step by: After each successful programming cycle the serial number is incremented by this value.

Endian: The Endian specifies whether the LSB (Least Significant Byte) or the MSB (Most Significant Byte) of the serial number is stored to the first location "Chip address for the first serial number byte".

Number Base: The generated serial number can be based on the decimal or hexadecimal number system.

Number Format: The generated serial numbers can be generated as ASCII-Text or binary values.

Hex code / ASCII: These two fields show a preview of the next serial number to be generated (Next Serial Number) according to the specified settings.

PROJECTS

Projects allow all settings related to the "Programming", "Copying", "Reading", and "Series Production" software modes to be saved and recalled.

The chip selection, file selection, chip options, serial number settings, the number of copies and activated/deactivated process steps are saved for each software mode. In addition, the chip, file, and chip option settings specific to every connected programming device are also saved for the production mode.

The project files use the "pep" extension. This is the abbreviation for "Prog-Express Project". The files are saved as standard text files which can also be opened with a standard text editor. For automation and remote control purposes the project files can also be created in another application and loaded into Prog-Express.

SAVING



The save button opens a file browser. Here the location of the project file and its name can be entered and then saved. If the file exists it can either be overwritten or the process can be aborted.

All settings from all dialogues and all software modes are stored.

LOADING



When loading all stored settings are restored after the desired project file is selected using the file browser.

Warning! All previous settings are erased when loading a project, only the buffer data is kept.

REMOTE CONTROL OF THE PROG-EXPRESS SOFTWARE

For special applications you can also control the Prog-Express software remotely from an external program. This can be used to, for example, perform fully automated programming of device-specific adjustment or measurement values.

Commands can be sent to the Prog-Express software when it is run using either a control file or command line parameters.

REMOTE CONTROL USING COMMAND LINE PARAMETERS

One or more commands can be passed using command line parameters. Every command begins with a slash. Some commands require parameter values such as the name of the file to be loaded. The parameters are separated by spaces, and file paths must be in quotes.

Examples:

```
Prog-Express.exe /open „C:\Directory\FileName.bin”
```

```
Prog-Express.exe /mode program /run 2 /exit
```

Prog-Express is a single-instance application. This means that you can send new commands to the software using command line parameters while the software is running without starting a new instance of the software every time.

REMOTE CONTROL USING SCRIPT FILES

If Prog-Express is started with the command line parameter “/remotefile filename”, then all commands in the file are executed. Using the additional “poll” command the file can be checked continuously for changes. As soon as the file is changed all commands in the file are executed.

Script files can also contain comments which are marked with a semicolon.

Example:

```
; Sample script  
open "C:\Directory\FileName.bin"  
mode program  
run 2  
exit
```

REMOTEFILE COMMAND

The “remotefile” command can be used to read commands from a file and execute them.

Example:

- Sample: `remotefile „C:\Directory\Remote.txt“` Loads the commands from the specified file and executes them.

POLL ON AND POLL OFF COMMANDS

Once the “poll on” command has been executed the file previously specified with the “remotefile” command is continuously checked for changes. As soon as the file is modified by an external program the commands contained in the file are executed by Prog-Express.

Monitoring is turned off using the “poll off” command. The “poll on” command has an additional parameter with which the interval in which the file is checked for changes can be specified in milliseconds.

Example:

- `Poll on` Turns on monitoring of the remote control file.
- `Poll off` Turns off monitoring of the remote control file.
- `Poll on 2000` Turns on monitoring of the remote control file. The file is checked every 2 seconds (2000 milliseconds) for changes.

OPEN COMMAND

With the “open” command a project file (.pep) or a data file (.bin, .hex, .mhx,...) can be opened. All files which do not have the “.pep” extension are loaded into the Hex Editor. All .pep files are recognized as project files and opened as such.

Examples:

- `open “C:\Directory\FileName.bin”` Loads the specified file into the Hex Editor.
- `open “C:\Directory\ProjectSettings.pep”` Loads the file with the stored Prog-Express settings.

MODE COMMAND

The software mode can be set using the “mode” command. Valid parameters are “program”, “copy”, “read”, “production”, “hexeditor”, and “options”. Example: “Prog-Express.exe /mode program” puts the software into “programming” mode.

Examples:

- `mode program` Puts the software into “programming” mode.
- `mode production` Puts the software into “series production” mode.

SELECTFILE COMMAND

The “selectfile” command sets a data source file for the “program chip” or “production” mode or a data target file for the “read chip” mode. Example:

- `selectfile "C:\Directory\FileName.bin"` Sets the specified file as data source/target in the currently selected “programming”, “reading” or “production” mode.

PROCESSSTEP COMMAND

Individual process steps can be turned on or off using the “processstep” command, the number of the process step, and the “on” or “off” parameter.

Examples:

- `processstep 3 on` Turns process step number 3 of the current software mode on.
- `processstep 5 off` Turns process step number 5 of the current software mode off.

AUTOIDENTIFY COMMAND

The chip can be automatically identified using the “autoidentify” command.

A filename can be specified as an additional parameter. If this is specified, Prog-Express saves the chip variant ID and the chip name into this file.

Examples:

- `autoidentify` Performs the auto chip identify function in the active Prog-Express mode.
- `autoidentify "C:\Directory\chip.txt"` Saves the chip variant ID and name into the specified file.

RUN COMMAND

The process of the active software mode can be started using the "run" command. The number of process cycles can be specified as an optional parameter.

If another command is sent after the "run" command, this command is only executed after the process is finished. In this manner several processes can be run one after the other.

Examples:

- `run` Starts the process.
- `run 10` Starts the process for 10 copies.

SAVELOG COMMAND

The contents of the log screen can be saved to a file using the “savelog” command.

A filename can be specified as an additional parameter. If this is not specified, Prog-Express saves the log data to the file “log.txt” onto the desktop.

Examples:

- `savelog` Saves the log data to the file log.txt.
- `savelog "C:\Directory\mylog.txt"` Saves the log data to the specified file.

SAVEDEVICEINFO COMMAND

Information about all connected Batronix USB devices can be saved to a file using the “savedeviceinfo” command.

This command saves the device number, name, serial number, firmware version, actual job, progress percentage and other information about the connected devices.

A filename can be specified as an additional parameter. If this is not specified, Prog-Express saves the device data to the file “deviceinfo.txt” onto the desktop.

Examples:

- `savedeviceinfo` Saves the device data to the deviceinfo.txt file.
- `savedeviceinfo "C:\Directory\devices.txt"` Saves the device data to the specified file.

CLEARLOG COMMAND

The contents of the log screen can be cleared using the “clearlog” command.

ADDITIONAL COMMANDS

- `hide` Hides the Prog-Express software.
- `show` Shows the Prog-Express software after a “hide” command.
- `exit` Closes Prog-Express.

SAMPLE APPLICATIONS: PROGRAMMING OF SPECIFIC DATA

Case study: In a production environment special software captures measurement data from specific devices and corresponding adjustment values then need to be programmed to a memory chip.

First all desired special settings such as chip options or serial numbers as well as the chip and the file to be programmed are set during normal Prog-Express operation and saved as a project.pep file.

Then the special software starts Prog-Express with the command line call:

```
Prog-Express.exe /remotefile "C:\Directory\Remote.txt" /poll on
```

This starts Prog-Express which then continues to monitor the file remote.txt for changes. The special software first tests the device, then saves the adjustment data to the file adjustments.bin and then saves the remote.txt file with the following contents:

```
;Sample application
open "C:\Directory\Project.pep"      ;Loads the program settings
mode program                        ;Switches to "Programming" mode
run                                  ;Starts the process
savelog                             ;Saves the log data to the file log.txt
```

Then the special software monitors the log.txt file, evaluates it and then continues on to the next device.