

User's Guide



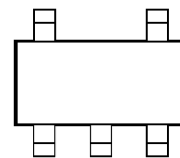
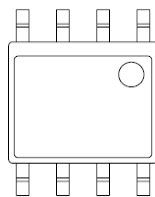
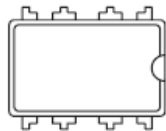
I²C Bus Serial EEPROM Programmer Software for MCC ASCII Interface I²C Bus Host Adapters

Version 2.2

10101000

I²C Bus

01001010



www.mcc-us.com

Introduction

The MCC iBurner™ I²C Bus Serial EEPROM Programmer Software provides a quick and easy way to program, read, and verify a wide variety of I²C Bus EEPROMS. iBurner is compatible with Windows 2000, XP, Vista, or higher, running .NET Version 2 or above. iBurner is also compatible with MCC ASCII interface based I²C Bus host adapters including the iPort/AI (#MIIC-202), iPort/AFM (#MIIC-203), and iPort/USB (#MIIC-204).

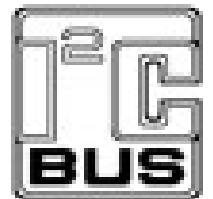
This user's guide describes the operation of the iBurner software.

Are you new to I²C? Want to know more? We suggest you review “What is I²C?” at www.mcc-us.com/I2CBusTechnicalOverview.pdf.

MCC products use Philips components and are licensed to use the I²C Bus.

“Purchase of Philips I²C components conveys a license under the Philips' I²C patent to use the components of the I²C system, provided the system conforms to the I²C specifications defined by Philips.”

I²C is a trademark of Philips Corporation.



09-DEC-08

Revision Report

09-DEC-08 Release Ver. 2.2

1. Enhance Script Engine validation, error handling, and logging.
2. Abort script processing immediately on command error detection. Quit command accepted but no longer required.
3. Correct Script Creator “Load” command offset generation.
4. Add script “*beep” command (beep computer speaker).
5. Add script “*settings” command (log current device settings).
6. Optimize Script Creator script generation.
7. Set initial Load/Save Buffer and Script folder.

10-JUL-08 Release Ver. 2.1

1. Correct EEPROM access algorithm.

02-JUL-08 Release Ver. 2.0

1. Easy-to-use scripting engine with Script Builder tool.
2. Intelligent iPort configuration determines optimum settings.
3. New checksum (CRC16) capability.
4. More I2C EEPROM device definitions.
5. More robust programming and logging facilities.

15-MAY-07 Release V1.1

1. Add support for Motorola S-Record file format.
2. Expand device library export function to support the exportation of one or more selected device definitions.

16-AUG-06 Initial Release V1.0

Copyright© 2008 by Micro Computer Control Corporation. All rights are reserved. No part of this publication may be reproduced by any means without the prior written permission of Micro Computer Control Corporation, PO Box 275, Hopewell, New Jersey 08525 USA.

DISCLAIMER: Micro Computer Control Corporation makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Further, Micro Computer Control Corporation reserves the right to revise the product described in this publication and to make changes from time to time in the content hereof without the obligation to notify any person of such revisions or changes.

WARNING - Life Support Applications: MCC products are not designed for use in life support appliances, devices, or systems where the malfunction of the product can reasonably be expected to result in a personal injury.

WARNING - Radio Frequency Emissions: This equipment can radiate levels of radio frequency energy that may cause interference to communications equipment. Operation of this equipment may cause interference with radio, television, or other communications equipment. The user is responsible for correcting such interference at the expense of the user.

WARNING - Electrostatic Discharge (ESD) Precautions: Any damage caused by Electrostatic Discharge (ESD) through inadequate earth grounding is NOT covered under the warranty of this product. See the “Electrostatic (ESD) Precautions” section of this guide for more information.

Printed in the United States of America

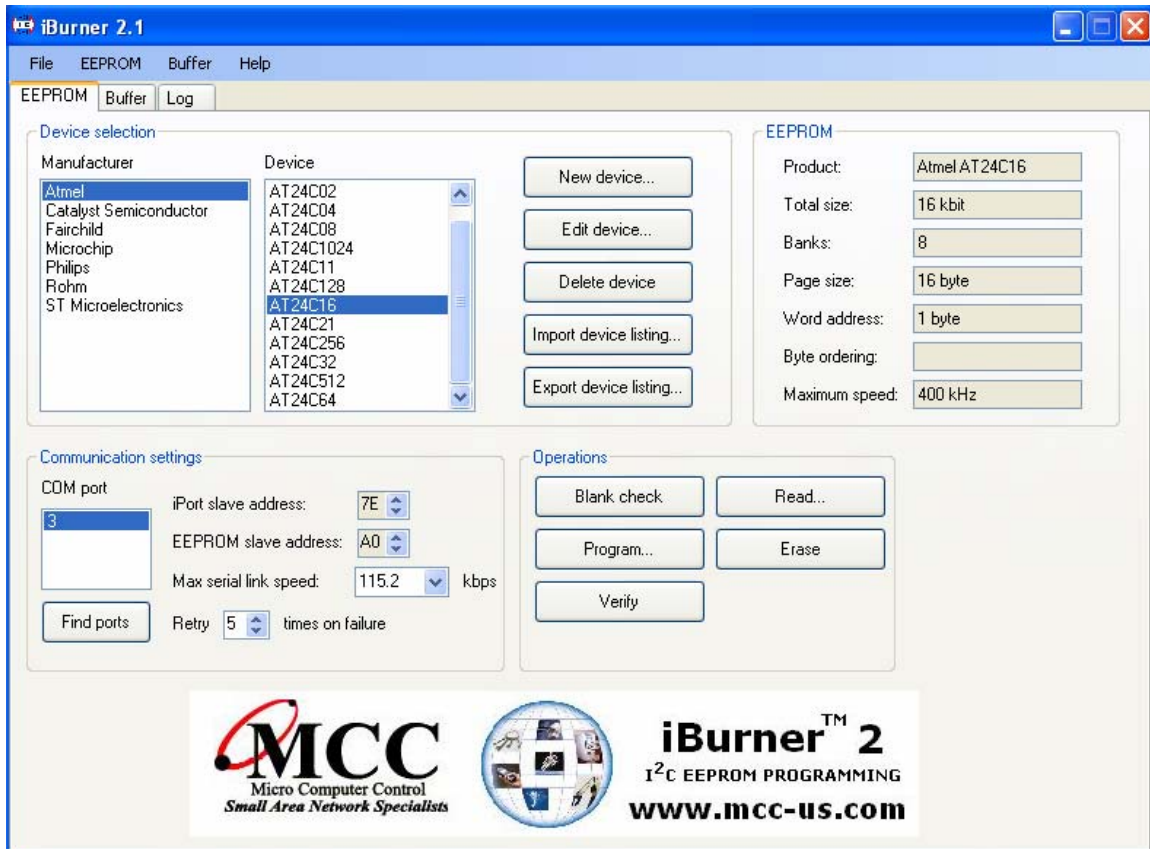
iBurner 2 User's Guide

© 2008 Micro Computer Control Corporation

Introduction to the iBurner software	2
New features and changes since iBurner 1.1	2
System requirements.....	2
Installation	3
I2C EEPROMs	4
Getting started	6
Selecting a device	6
Configuring the iPort	7
Using the buffer.....	8
Buffer operations	8
Loading files into the buffer	10
Saving the buffer to a file	11
Checksums.....	11
Manipulating EEPROMs.....	13
Blank check	13
Program	14
Verify.....	15
Read.....	15
Erase	16
Logs.....	17
The device library.....	18
Editing an existing device.....	18
Adding a new device	19
Removing an existing device.....	19
Exporting devices	20
Importing devices	20
Automation and scripting	22
Script Creator Tool	22
Configure	23
Load file.....	23
Program.....	24
Verify	24
Erase	24
Blank check.....	24
Print message	24
Pause	24
Running scripts from the command line.....	25
Script logging	25
Script language specification	26
IBS signature.....	26
The stack.....	26
Commands	27
Sample scripts.....	28

Introduction to the iBurner software

iBurner is an EEPROM programmer for I2C based products, developed to interface with the iPort line of I2C bus host adapters. It can program, read, erase, and verify data stored on most serial EEPROMs and includes a robust scripting engine for automation. The software is accompanied by a collection of EEPROM device definitions for the most popular devices from major manufacturers, and can be easily updated with new devices. iBurner provides a simple yet powerful interface to I2C serial EEPROMs and is available free of charge from Micro Computer Control Corporation.



New features and changes since iBurner 1.1

- Easy-to-use scripting engine with script builder
- Intelligent iPort configuration determines optimum settings
- New checksum (CRC16) capability
- More I2C EEPROM device definitions
- More robust programming and logging facilities

System requirements

In order to use the iBurner software you must satisfy the following system requirements.

1. Microsoft Windows operating system
 - Windows 98 SE
 - Windows 2000

- Windows XP
 - Windows Vista
2. Microsoft .Net Runtime Environment
 - Version 2 or newer
 3. I2C bus host adapter
 - iPort/AI (#MIIC-202)
 - iPort/AFM (#MIIC-203)
 - iPort/USB (#MIIC-204)

For more information on purchasing iPort host adapters, visit www.mcc-us.com.

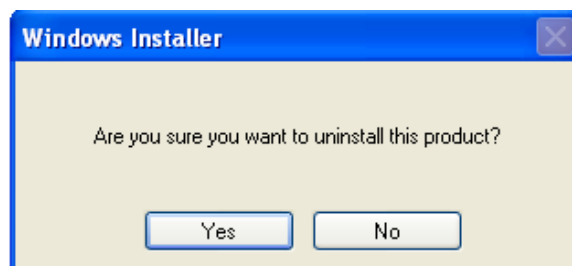
Installation and uninstallation

iBurner 2 is provided as a Microsoft Windows Installer (*.msi) can be downloaded free-of-charge from the MCC website: www.mcc-us.com. iBurner is available to all MCC customers who have purchased an iPort/AI, iPort/AFM, or iPort/USB bus host adapter.

You do not need to remove old versions of iBurner to install iBurner 2.x.

To install, download the setup file and run it. iBurner requires the Microsoft .Net 2.0 framework; if it is not installed, you will be presented with a link to MCC's website for instructions on downloading .Net. If your system already has .Net 2.0 or newer you will not see this message.

iBurner can be started from the Start menu, under the "iBurner 2.x" folder. Also provided is the iBurner script creator tool and the manual (this file) and license. To uninstall iBurner, click "Uninstall" from the "iBurner 2.x" folder on the Start menu.



I2C EEPROMs

I2C EEPROMs are serial EEPROMs that can be programmed or read over an I2C bus. The EEPROMs are I2C slaves and implement a simple protocol through which they can be accessed. In addition to the total size of the EEPROM, there are several important parameters that you should be familiar with.

The **page size** of an EEPROM is the number of bytes that can be programmed or read at a single time. This is usually a multiple of 16. After a page of data is transmitted to the device the EEPROM will commit the data to memory; thus the larger the page size the faster the chip can be programmed.

The **addressing bytes** of the EEPROM are the bytes that must be transmitted to select a particular location to read or write. Small EEPROMs require a single byte while larger devices require two. If two addressing bytes are required, the **byte ordering** (most-significant-byte to least-significant-byte or vice versa) must be known. The most common I2C EEPROMs from major manufacturers use MSB-to-LSB ordering.

Some EEPROMs allow for larger memories without resorting to a second address byte; this is accomplished through **banks**. A bank is a slave address at which the EEPROM will access a particular section of its memory. For example, a 512 byte EEPROM would typically require two addressing bytes, as a single addressing byte can only index 256 bytes of memory. Such an EEPROM might instead use two banks, or two I2C slave addresses at which it will respond; the first would access the lower 256 bytes, and the second would access the upper 256 bytes.

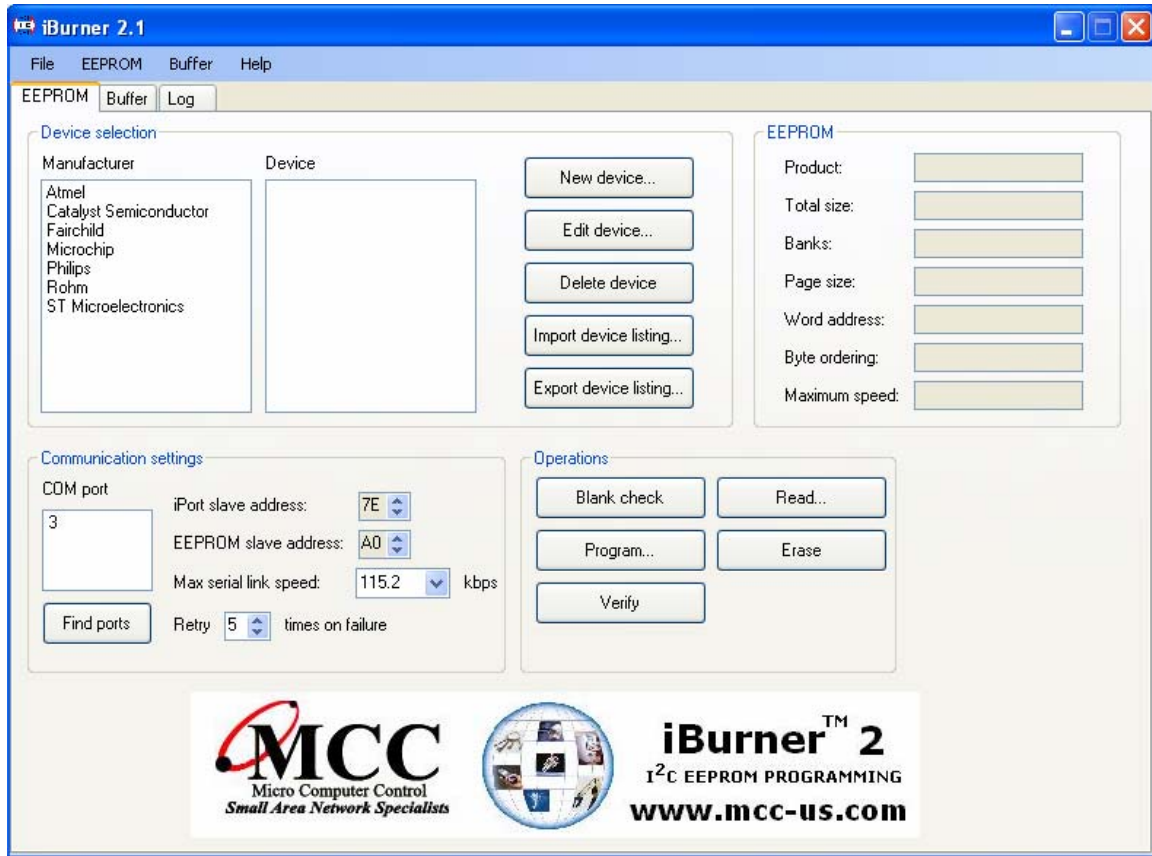
The I2C address of the first bank (the only bank for many EEPROMs) is referred to as the **slave address** of the device. On some EEPROMs, external pins are available to change this slave address. All EEPROM slave addresses are between A0 and AE, inclusive (and like all I2C addresses, include only the even numbers).

iBurner can program any I2C based EEPROM for which the slave address, total size, number of addressing bytes, byte ordering, and the number of banks is known. The slave address can be easily determined from the configuration of the EEPROM's external pins (if available; otherwise the address is fixed at A0) and the total size is readily available from the device documentation. The number of addressing bytes and banks can be determined from documentation, or from the following table, once the total size is known.

<i>Total size</i>	<i>Banks</i>	<i>Addressing bytes</i>	<i>Possible slave addresses (depending on pin configuration)</i>
16 bytes, 32 bytes, 64 bytes, 1 kbit, 2 kbit	1	1	A0, A2, A4, A6, A8, AA, AC, AE
4 kbit	2	1	A0, A4, A8, AC
8 kbit	4	1	A0, A8
16 kbit	8	1	A0
32 kbit, 64 kbit, 128 kbit, 256 kbit, 512 kbit	1	2	A0, A2, A4, A6, A8, AA, AC, AE
1 Mbit	2	2	A0, A4, A8, AC
2 Mbit	4	2	A0, A8
4 Mbit	8	2	A0

iBurner has a built-in database (to which new devices can be added or removed, and which can be exported for importation to other iBurner installations) with definitions for the most popular devices from major manufacturers.

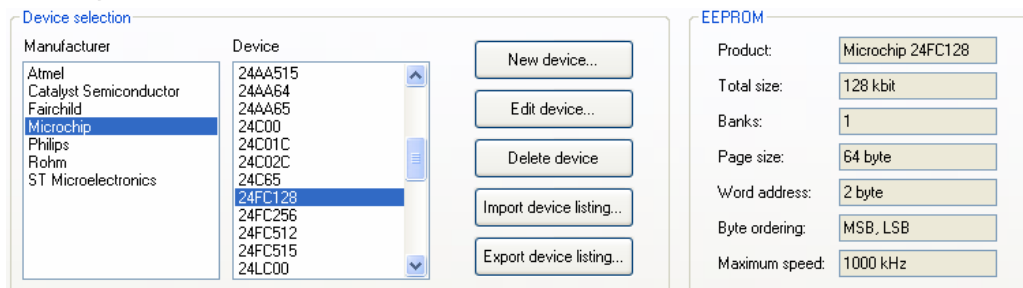
Getting started



When iBurner is first started, you will see a window like that above. There are three tabs; the *EEPROM* tab is used to select an EEPROM and provide iPort configuration settings, as well as accessing some common actions. The *Buffer* tab provides a display of the current memory buffer upon which work is being done. Reading the EEPROM will read data into this buffer, and programming the EEPROM will write data from this buffer. The *Log* tab displays information on current and past operations.

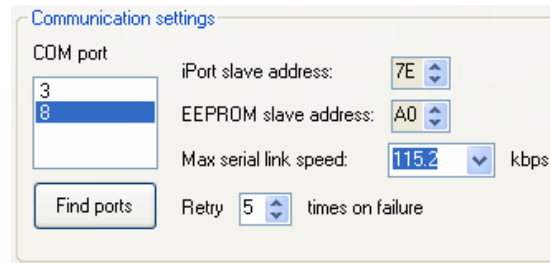
The first thing that should be done is configuration of the device and iPort. Once a device is selected the buffer will become available, and once the iPort is configured you will be able to perform operations.

Selecting a device



Use the *Manufacturer* list to select the manufacturer of your EEPROM. When you choose a manufacturer, the *Device* list will be populated with the available devices produced by that manufacturer. Selecting a device will populate the *EEPROM* fields with the stored device description.

Configuring the iPort



The screenshot shows a dialog box titled "Communication settings". It contains several fields and a button:

- COM port:** A list box with options 3 and 8. Option 8 is selected.
- Find ports:** A button located below the COM port list.
- iPort slave address:** A spin box set to 7E.
- EEPROM slave address:** A spin box set to A0.
- Max serial link speed:** A spin box set to 115.2, followed by the unit "kbps".
- Retry:** A spin box set to 5, followed by the text "times on failure".

Use the *Communication settings* fields to configure the iPort. If necessary, click the *Find ports* button to have iBurner redetect available serial ports (including virtual serial ports provided by iPort/USB devices). Select the serial port that corresponds to your device.

The iPort slave address can be left at the default 7E for most users; if you are programming an EEPROM on an active bus (a bus on which other I2C masters and slaves are operating at the same time as iBurner and the target EEPROM) you will want to verify that the iPort slave address is not shared with another device on the bus. You can select any valid I2C address for the iPort slave address.

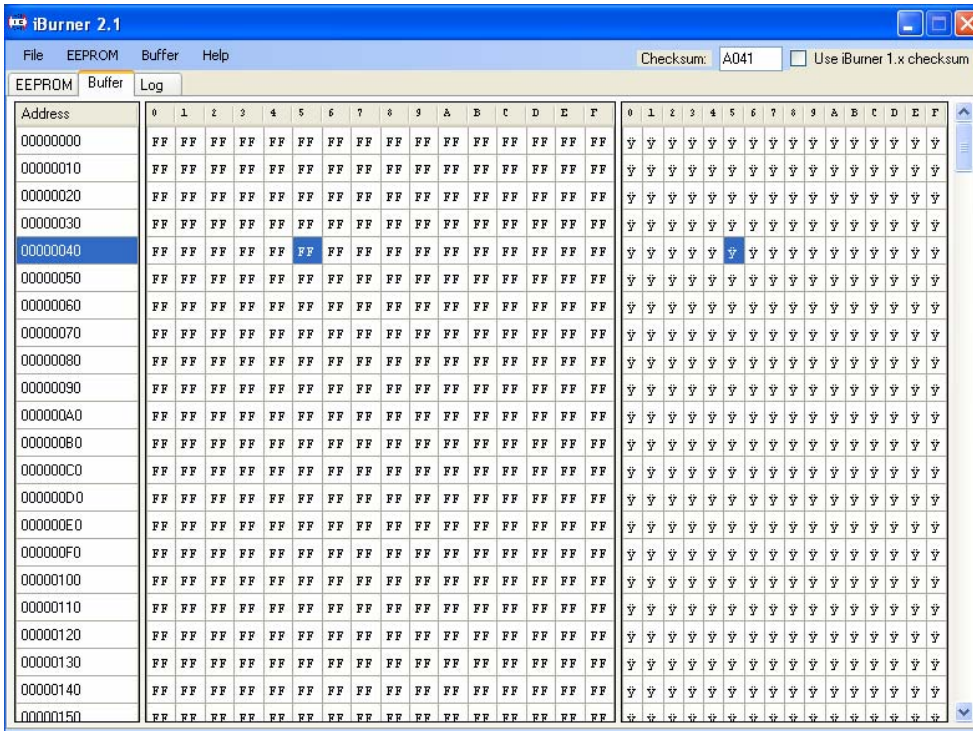
The EEPROM slave address can be set to any valid I2C address, but all manufacturer-produced EEPROMs will have slave addresses between A0 and AE, depending on available addresses and pin configuration (see the section “I2C EEPROMs”).

The maximum serial link speed should be left at the maximum value, 115.2 kbps, unless the user desires slower operation. iBurner will automatically adjust this value if the connected iPort cannot support the selected speed.

The retry count is used to repeat individual failed reads and writes in the context of an EEPROM operation. The default value of 5 will suffice for most users.

Once an EEPROM is selected and the desired iPort settings are chosen, buffer and EEPROM operations may be executed.

Using the buffer

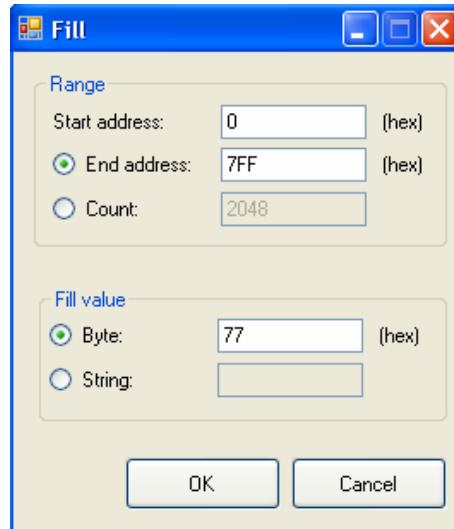


Buffer operations are executed from the *Buffer* tab and the *Buffer* menu. The *Buffer* tab has two main panels; the left panel displays the contents of the buffer in hexadecimal while the right displays the same information as printable ASCII characters (if possible). The contents of the buffer may be edited manually by double-clicking on any cell and entering a new value (either hex in the left panel or ASCII in the right panel).

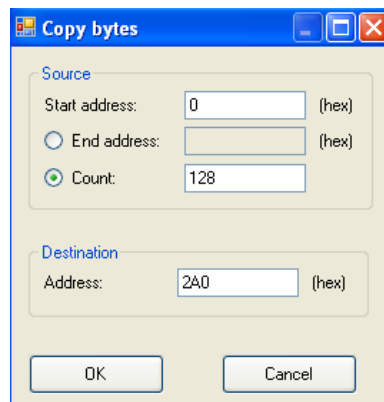
2	3	4	5	6	7	8	9	A	B	C
63	72	6F	20	43	6F	6D	70	75	74	65
74	72	6F	6C	20	43	6F	72	70	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

Buffer operations

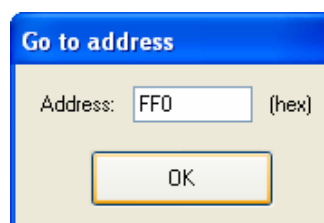
The *Buffer* menu provides three operations: *Fill*, *Copy*, and *Go to address*.



The *Fill* operation fills some portion of the buffer with data; either a repetition of a single byte value or a single instance of an ASCII string. The fill range can be described by a start address and either an end address or a count of bytes to be filled. The start address defaults to the address of the currently selected byte in the buffer and the end address defaults to the last address in the buffer (similarly, the count defaults to the total size of the buffer).

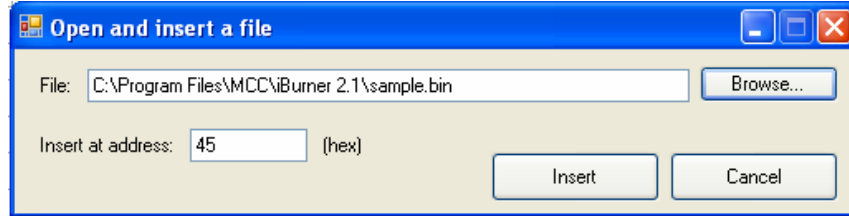


The *Copy* operation copies some portion of the buffer to a different location. The source and destination locations can overlap (eg. bytes 0 through 4 could be copied to bytes 1 through 5, if desired). The start address defaults to the address of the currently selected byte in the buffer; the end address (or count) and the destination address have no default value.

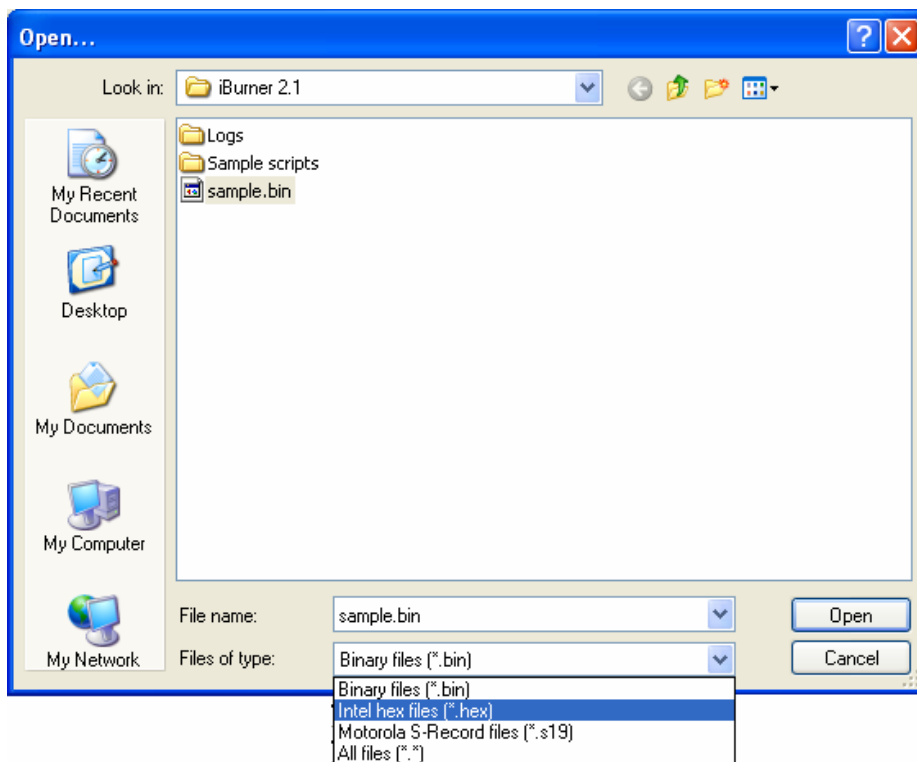


The *Go to address* operation simply moves the buffer view such that the user can see and edit the memory at the desired address.

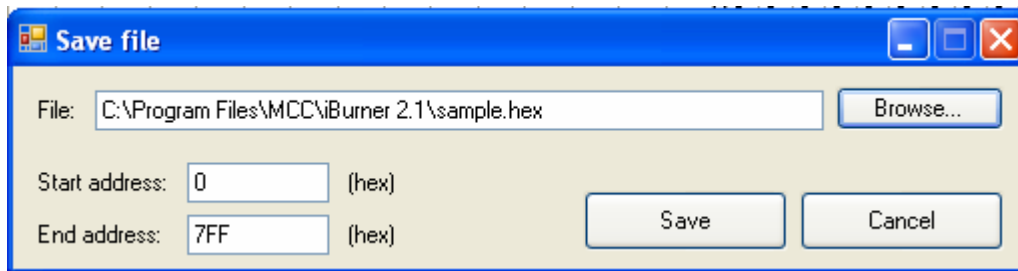
Loading files into the buffer



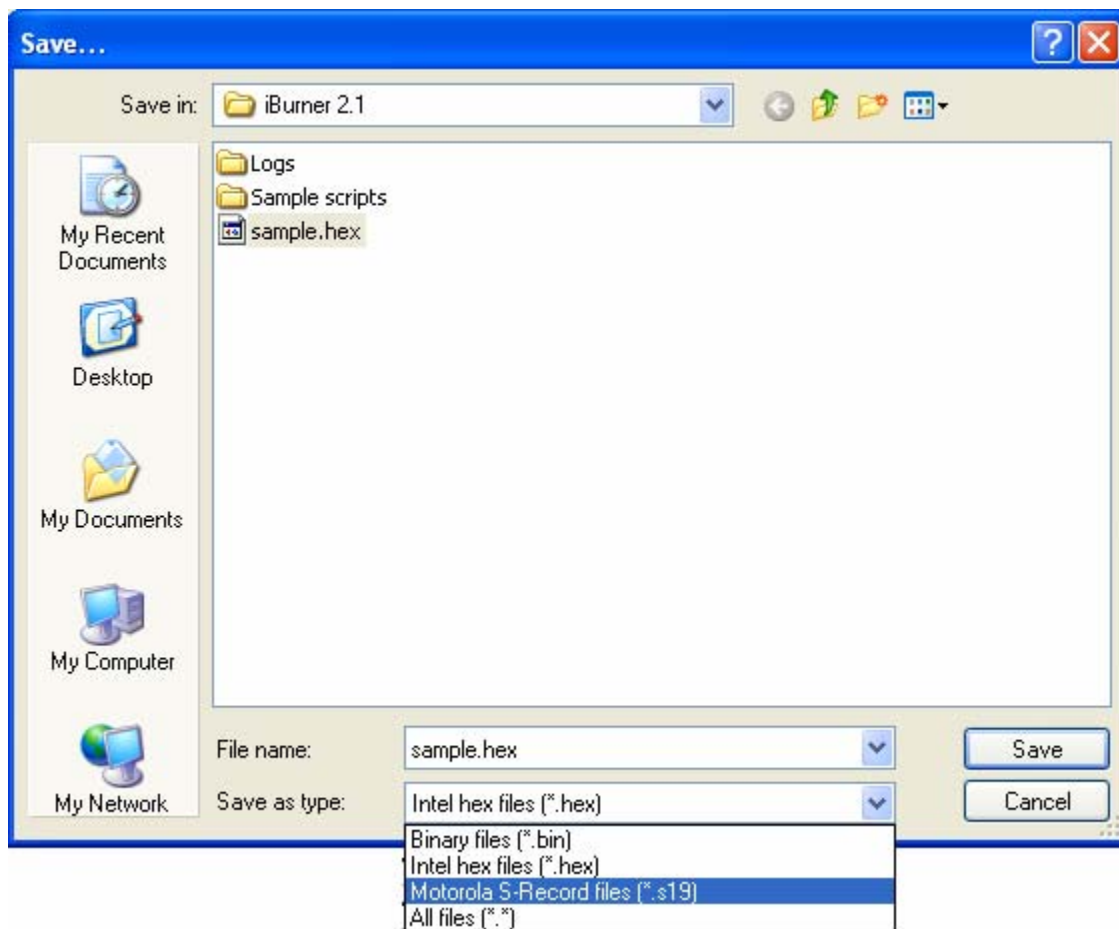
iBurner can load three types of files into the buffer: binary (or flat) files, Intel Hex files, and Motorola S-Record (S19) files. A binary file is any file that the user wishes to load byte-for-byte into the buffer. All files can be loaded with offsets: eg. a 128 byte binary file could be loaded into the buffer starting at address 0x70. The offset defaults to the address of the currently selected byte in the buffer.



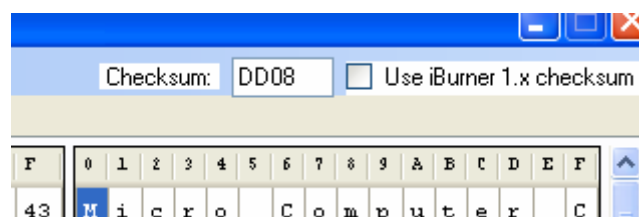
Saving the buffer to a file



The buffer, or some portion thereof, can be saved to a file. As with loading files, binary, Intel Hex, and Motorola S-Record files are supported. The selected range to save defaults to the entire buffer.

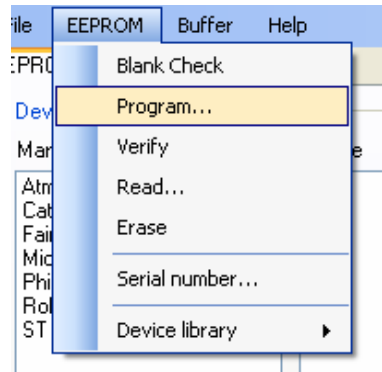


Checksums



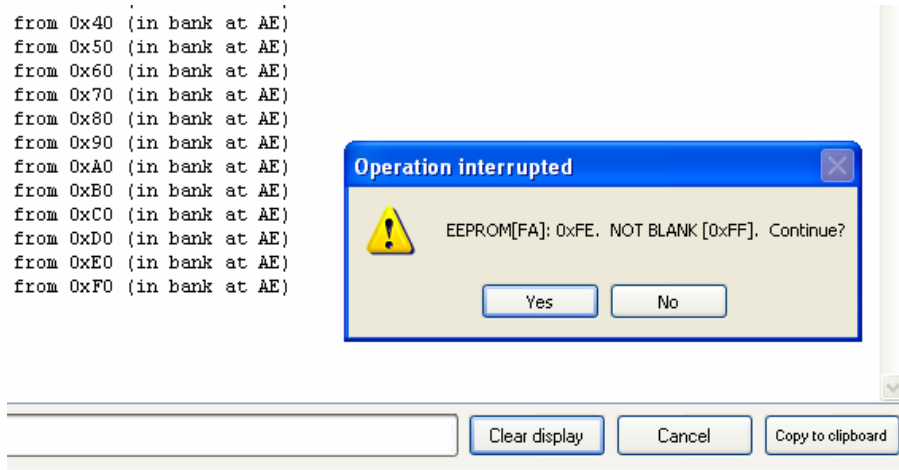
The *Buffer* tab displays the current checksum for the buffer in a text box at the top of the window. This checksum is a standard CRC16 computation (CRC polynomial A001) and can be used to quickly verify the contents of the buffer. To preserve compatibility with older versions of the iBurner software, the “Use iBurner 1.x checksum” checkbox may be checked; this is not recommended for use beyond backwards compatibility – the default CRC16 provides a more common and robust checksum.

Manipulating EEPROMs



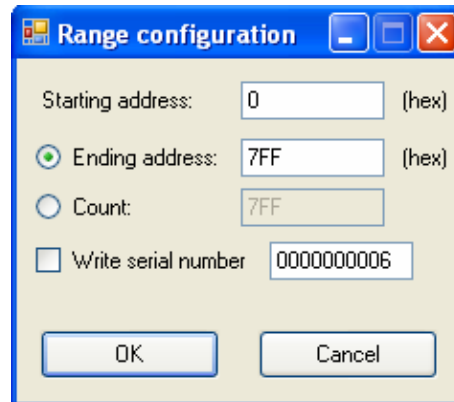
iBurner can perform five separate device operations, as detailed below.

Blank check



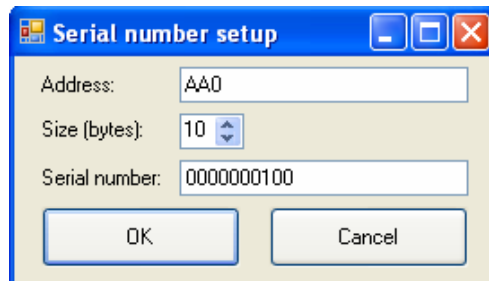
Clicking the *Blank check* button or *Blank check* from the *EEPROM* menu initiates a scan of the connected EEPROM to determine if the device contains any data (non-FF bytes). If such a byte is detected, the user is prompted to stop checking or continue. The information is logged as it is processed, including user decisions.

Program



Clicking the *Program...* button or *Program...* from the *EEPROM* menu initiates a full or partial program of the connected EEPROM with data from the buffer. The configuration window can be used to select the range to be programmed; the range defaults to the entire memory. Before a program operation begins the user is warned that the contents of the EEPROM will be changed and prompted to continue.

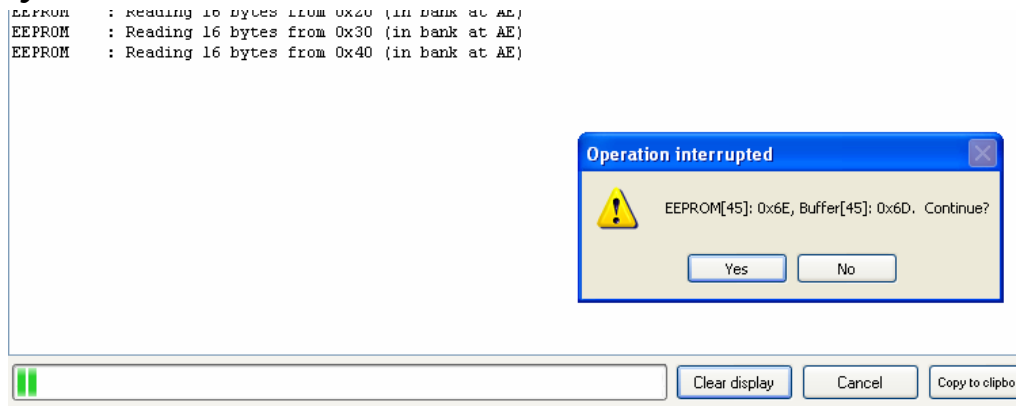
iBurner can insert a serial number string into the buffer prior to programming, and automatically increment this serial number upon successful completion of the program operation. To configure the serial number, click *Serial number...* from the *EEPROM* menu. A window such as that below will be displayed.



The address field is the hexadecimal address in the buffer where the serial number string should be inserted before each program operation. Enter the size of the serial number string, and the current value. You may now click the "Write serial number" checkbox in the program operation configuration dialog; the displayed serial number will be inserted into the buffer. Following the program operation the serial number will be incremented.

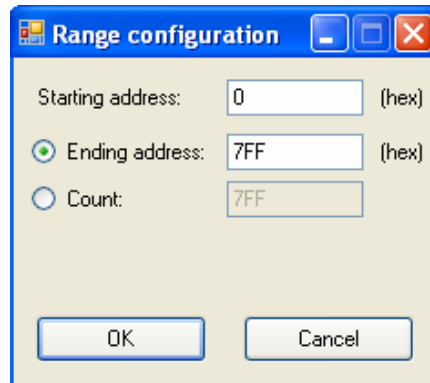
The serial number configuration is maintained between iBurner sessions.

Verify



Clicking the *Verify* button or *Verify* from the *EEPROM* menu initiates a comparison between the contents of the EEPROM and the buffer. If any bytes are found to be different, the user is prompted to stop checking or continue. The information is logged as it is processed, including user decisions.

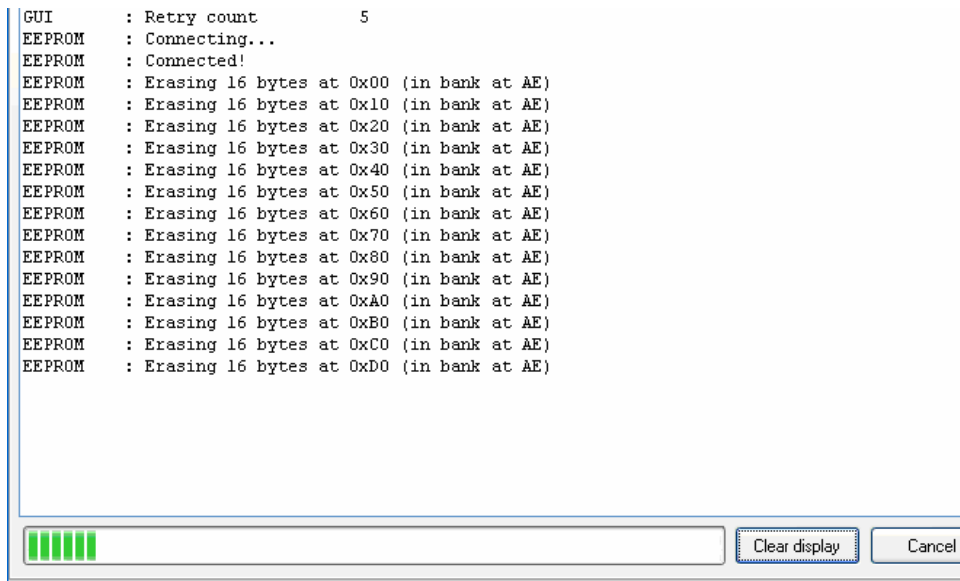
Read



Clicking the *Read...* button or *Read...* from the *EEPROM* menu initiates a full or partial loading of data from the EEPROM into the buffer. The configuration window can be used to select the range to be loaded; the range defaults to the entire memory. Before a read operation begins the user is warned that the contents of the buffer will be changed and prompted to continue.

Erase

```
GUI      : Retry count      5
EEPROM   : Connecting...
EEPROM   : Connected!
EEPROM   : Erasing 16 bytes at 0x00 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0x10 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0x20 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0x30 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0x40 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0x50 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0x60 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0x70 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0x80 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0x90 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0xA0 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0xB0 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0xC0 (in bank at AE)
EEPROM   : Erasing 16 bytes at 0xD0 (in bank at AE)
```



Clicking the *Erase* button or *Erase* from the *EEPROM* menu initiates a complete erasure of the EEPROM device by programming it with FF bytes. Before the erase operation begins the user is warned that the contents of the device will be changed and prompted to continue.

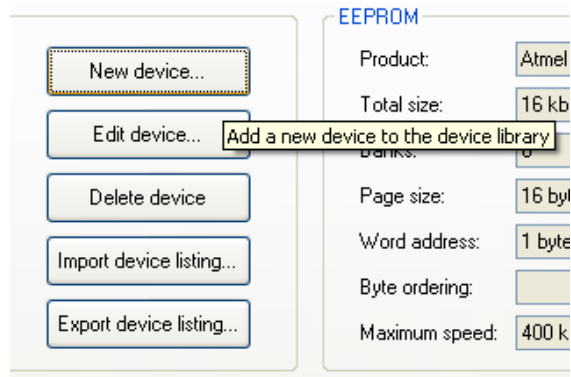
Logs

iBurner has two logging systems. While operations are in progress they write information to the *Log* tab indicating success or failure and progress. The *Log* tab also provides the *Cancel* button, which halts the current operation, and a *Copy to clipboard* button that will place the contents of the log on the clipboard for pasting into another application. The *Clear* button clears the on-screen display.

In addition to the *Log* tab, all iBurner content is logged to files. Files are named with the date and the “.log” extension, as in “yyyymmdd.log”. The log file to which data is being stored is indicated in the log itself and can be seen in the on-screen display. To view a log file, click *View log file...* in the *File* menu, and select the desired file. A *Copy to clipboard* option is provided in the resulting window to copy the contents of an entire day’s operations.

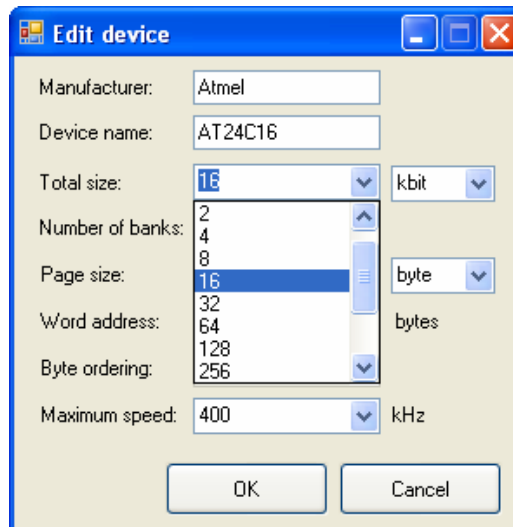
All operations that are processed in iBurner are logged, both those initiated with the GUI and the scripted interface. For more information on the logging of scripts, see the “Automation” section.

The device library



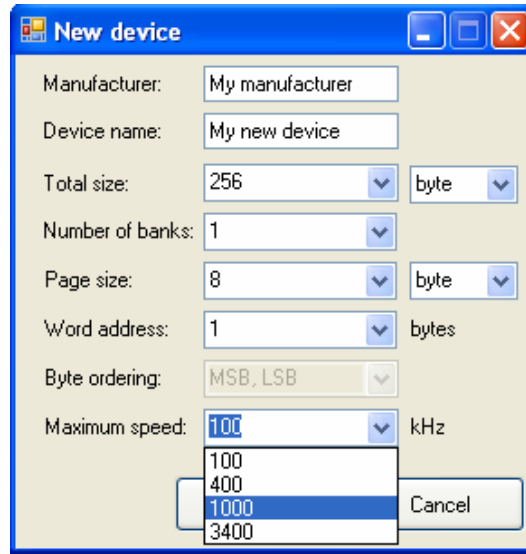
The device library is a collection of EEPROM devices, by manufacturer, for which iBurner contains predefined settings. The device library is loaded when iBurner starts and can be modified in a number of ways.

Editing an existing device



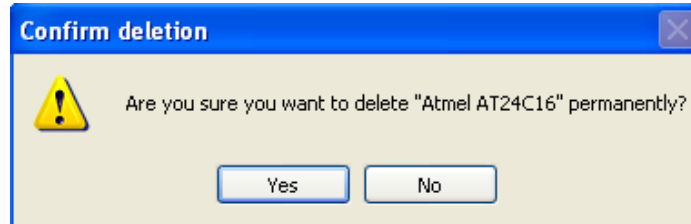
Existing devices can be edited by selecting the manufacturer and device and clicking the *Edit device...* button or *Edit device...* from the *Device Library* submenu of the *EEPROM* menu. This is useful for correcting improperly added devices or changing a device definition so it is used differently (with a smaller page size, with a lower bus speed, etc).

Adding a new device



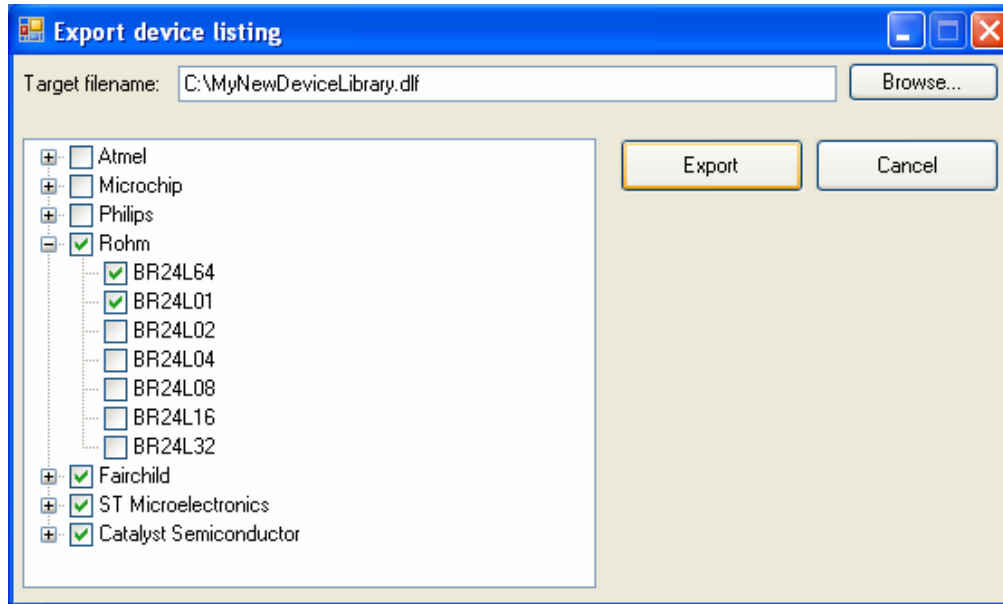
New device descriptions can be added to the device library by clicking the *New device...* button or *New device...* from the *Device Library* submenu of the *EEPROM* menu. See the "I2C EEPROMs" section for information on determining the proper parameters to describe a particular device.

Removing an existing device



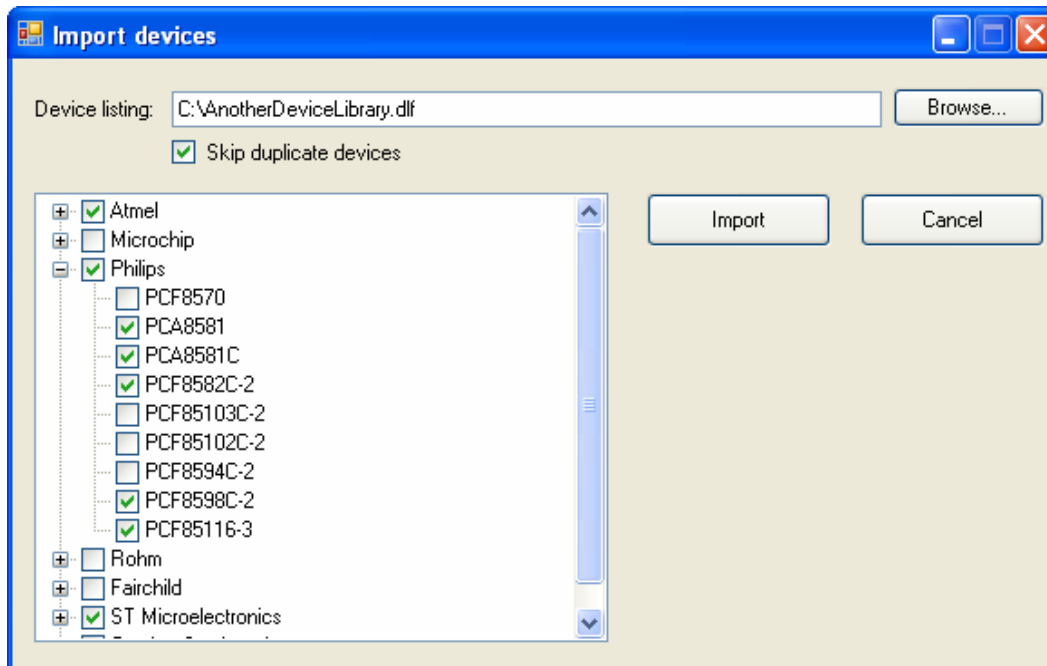
Devices can be removed from the device library by selecting the manufacturer and device and clicking the *Delete device* button or *Delete device* from the *Device Library* submenu of the *EEPROM* menu. A dialog box is displayed to confirm the removal.

Exporting devices



The complete device library or any selection of devices within it can be exported for use in another installation of iBurner by clicking the *Export device listing...* button or *Export device listing...* from the *Device Library* submenu of the *EEPROM* menu. Devices can be checked and unchecked to denote whether or not they will be exported. By default, no devices are checked. A device manufacturer can be checked or unchecked to check or uncheck all of the devices belonging to that manufacturer.

Importing devices



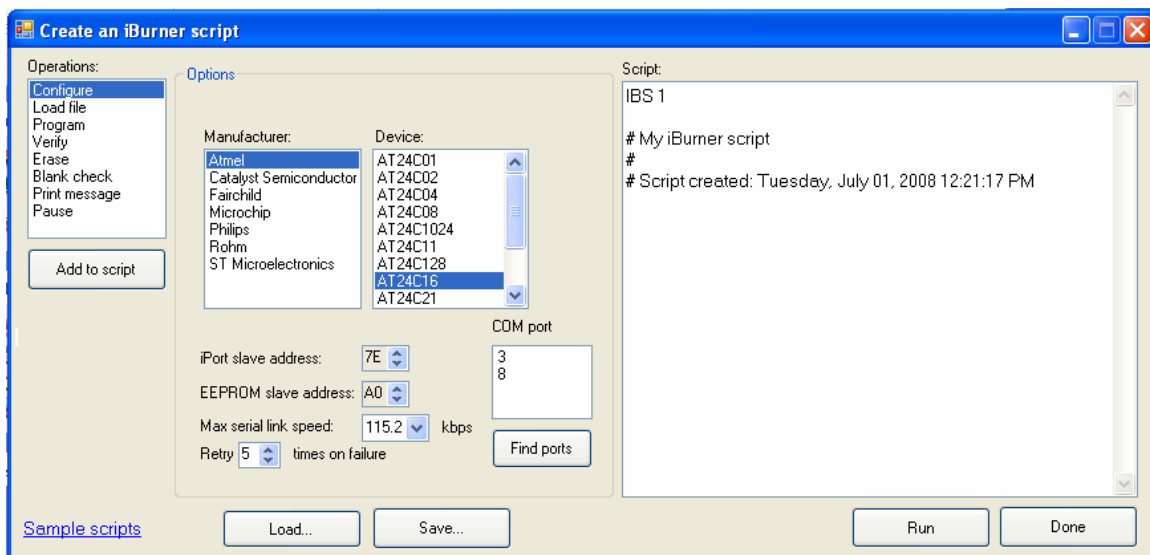
A complete device library that was previously exported or any selection of devices within it can be imported into iBurner by clicking the *Import device listing...* button or *Import device listing...* from the *Device Library* submenu of the *EEPROM* menu. Devices can be checked or unchecked to denote whether or not they will be imported. By default, no devices are checked. A device manufacturer can be checked or unchecked to check or uncheck all of the devices belonging to that manufacturer.

Automation and scripting

iBurner provides a powerful script language that you can use to automate many EEPROM operations. iBurner script (IBS) files can be written by hand, adapted from the provided samples, or constructed with the Script Creator tool. Scripts can be run from the Script Creator tool or from the command line, suitable for batch file or standalone use.

Script Creator Tool

The Script Creator tool provides a GUI for writing IBS files by allowing the user to select the desired operations, enter the required parameters, and automatically construct the script. The Script Creator can be started from the iBurner 2.1 folder on the Start menu.



To create a script, select operations in the list on the left, enter the proper options into the center panel, and click “Add to script” to add the necessary IBS code to the script file. Click “Run” to run the script; you will be prompted to save before the script is started. You can also edit the script manually in the on-screen display.

Configure

The screenshot shows the 'Configure' operation configuration window. The 'Operations' list on the left includes 'Configure', 'Load file', 'Program', 'Verify', 'Erase', 'Blank check', 'Print message', and 'Pause'. The 'Options' section on the right contains the following fields:

- Manufacturer:** A list box with 'Atmel' selected.
- Device:** A list box with 'AT24C16' selected.
- iPort slave address:** A spin box set to '7E'.
- EEPROM slave address:** A spin box set to 'A0'.
- Max serial link speed:** A spin box set to '115.2' kbps.
- Retry:** A spin box set to '5' times on failure.
- COM port:** A list box with '3' selected.

An 'Add to script' button is located below the 'Operations' list, and a 'Find ports' button is located below the 'COM port' list.

The “Configure” operation should be the first operation added to the script. Select the manufacturer and device name from the device library (if your device is not present you can add it in the main iBurner application as detailed earlier in this guide). Select the slave address of the EEPROM device, the COM port, and the maximum baud rate you wish the application to use, along with the number of times to retry on failure, and click “Add to script”. A “Configure” operation consists of a number of individual commands, which will be added automatically, along with comments and error detection.

Load file

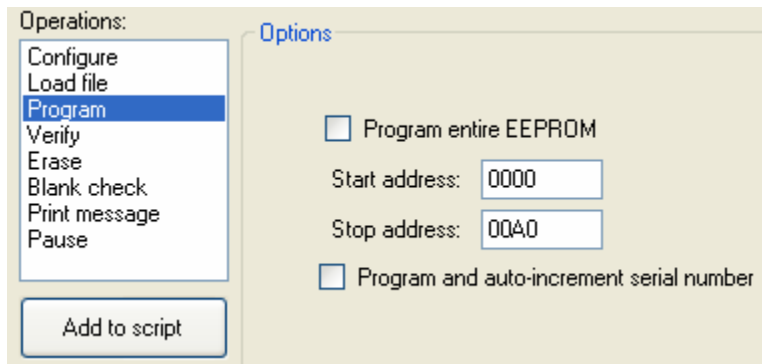
The screenshot shows the 'Load file' operation configuration window. The 'Operations' list on the left includes 'Configure', 'Load file', 'Program', 'Verify', 'Erase', 'Blank check', 'Print message', and 'Pause'. The 'Options' section on the right contains the following fields:

- File:** An empty text box with a 'Browse...' button to its right.
- Offset:** A text box containing '0000'.

A list box below the 'File' field shows the following file types: 'Binary', 'Intel Hex', and 'Motorola S-Record (S19)'. An 'Add to script' button is located below the 'Operations' list.

The “Load file” operation allows you to load a file into the buffer at a particular offset. The Script Creator supports the same files as the iBurner application. Click “Browse...” to select the file, choose the proper file type and offset, and click “Add to script”.

Program



The “Program” operation programs a selected region of the EEPROM with the buffer, and optionally inserts a serial number into the buffer (and auto-increments it upon successful programming). The serial number configuration (size, initial value, and offset within the buffer) can be set in the main iBurner application.

Verify

The “Verify” operation performs a byte-by-byte comparison of the EEPROM device and the buffer. If they do not match, or an error interrupts the process, the script terminates with a message; otherwise script operation continues. There are no configuration options for the “Verify” operation (you can edit the script manually if you want to change the message or operation behavior).

Erase

The “Erase” operation writes blank (0xFF) bytes to the EEPROM device. There are no configuration options for the “Erase” operation.

Blank check

The “Blank check” operation verifies that every byte in the EEPROM device is blank (0xFF). If a non-blank byte is found, or an error interrupts the process, the script terminates with a message; otherwise script operation continues. There are no configuration options for the “Blank check” operation (you can edit the script manually if you want to change the message or operation behavior).

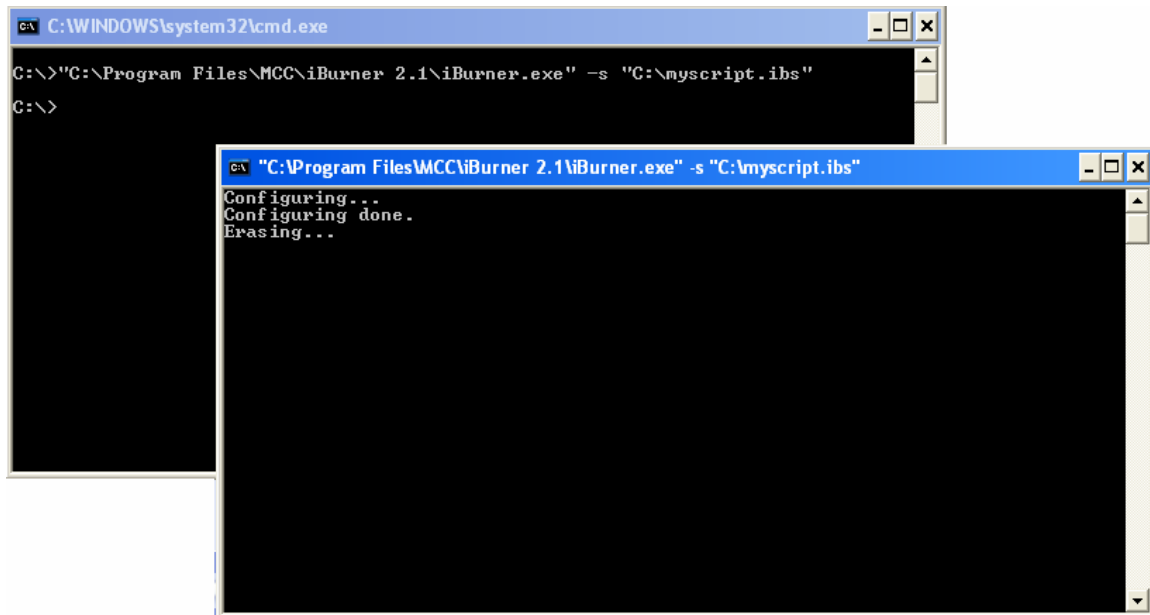
Print message

The “Print message” operation displays a message when executed. Enter the desired message and click “Add to script”.

Pause

The “Pause” operation pauses script execution until the user presses a key. If the user presses the escape key, script operation terminates (without a message).

Running scripts from the command line



IBS scripts can be run from the command line, from shortcuts, or from batch files, with the following command.

```
"C:\Program Files\MCC\iBurner 2.1\iBurner.exe" -s  
"C:\The\Path\To\My\Script.ibs"
```

The “-s” option indicates that the program should run in scripted mode. Alternatively you could specify “-m” which will interpret a script in “managed” mode, meaning that the script will wait for the user to press a key before exiting. This is useful for temporary command line windows, such as that created when running scripts with Windows shortcuts.

You can start the Script Creator itself from the command line, as well.

```
"C:\Program Files\MCC\iBurner 2.1\iBurner.exe" -c
```

Script logging

Scripts log the same information as operations performed manually in the main iBurner application, but this information is not displayed to the screen (only script outputs are displayed on the command line). The logged information is stored to the same log files as used in the main iBurner application, and can be retrieved by clicking “File”, “View log files...” or by navigating to the “Logs” subdirectory of the program installation directory.

Script language specification

The Script Creator provides a handy way to generate scripts; if you require advanced control over the script or wish to edit an existing script, this section will document the IBS format in its entirety.

This is version 1 of the IBS file format.

IBS signature

Every IBS file begins with an IBS signature, a single line containing “IBS X”, where X is the version of the IBS format. Attempting to execute scripts of newer versions than that supported by the iBurner software will yield an error, causing script termination.

The stack

iBurner scripts operate on a stack (LIFO structure). The IBS file is processed line-by-line, with the first character on the line determining how the contents of the line interact with the stack. Blank lines are ignored.

Any line that begins with # is a comment. The line is ignored and script execution moves on to the next line.

Any line that begins with * is a command. There is a fixed set of available commands, documented below. Commands pop some number of arguments off the stack, operate on them, and then push some number of return arguments onto the stack.

All other lines are assumed to be data. Data is pushed onto the stack.

The following script illustrates the difference between comments, data, and commands, and how commands operate on the stack.

```
IBS 1

# The "5" below is neither a comment nor a command,
# so it is pushed onto the stack.
5

# The "port" command below pops one argument off the
# stack and if it is a valid number sets it as the
# COM port on which further operations will occur.
*port

# The "port" command pushes true or false, indicating
# whether or not the provided number was valid. The
# "quit" command will display a message and quit if it
# sees a false on the stack, so we can quit if the
# "port" command failed. First we push the message...
The port number was invalid!
```

```

#...now we issue the "quit" command.
*quit

# If the port number was bad, the script won't get
# this far, it will display our message and quit.
# Otherwise the script will just keep running.

```

Commands

The following lists all commands available to iBurner scripts. The arguments are listed in the order they should be pushed.

Command	Arguments	Return arguments	Description
msg	<i>text</i>	none	Displays <i>text</i> to the screen.
pause	none	none	Waits for the user to press a key. If the key is ESC, the script exits immediately.
quit	<i>flag, text</i>	none	If <i>flag</i> is true, displays <i>text</i> to the screen and terminates script execution. Otherwise does nothing (falls through)
port	<i>port</i>	<i>flag</i>	<i>port</i> is established as the desired COM port. <i>flag</i> indicates success or failure. <i>port</i> should be a decimal number ("1", "14", etc).
baud	<i>rate</i>	<i>flag</i>	<i>rate</i> is established as the serial link speed. <i>flag</i> indicates success or failure. <i>rate</i> should be "19.2", "57.6", or "115.2".
haddress	<i>addr</i>	<i>flag</i>	<i>addr</i> is established as the host adapter I2C address. <i>flag</i> indicates success or failure. <i>addr</i> should be a two digit hexadecimal value ("A0", "AE", etc).
eaddress	<i>addr</i>	<i>flag</i>	<i>addr</i> is established as the target EEPROM I2C address. <i>flag</i> indicates success or failure. <i>addr</i> should be a two digit hexadecimal value ("A0", "AE", etc).
retry	<i>count</i>	<i>flag</i>	<i>count</i> is established as the retry count. <i>flag</i> indicates success or failure. <i>count</i> should be a decimal number ("0", "5", etc).
eprom	<i>mfg, prod</i>	<i>flag</i>	The EEPROM with manufacturer <i>mfg</i> and product name <i>prod</i> is established as the target EEPROM. <i>flag</i> indicates success or failure.
load	<i>fname, form, offset</i>	<i>flag</i>	The file <i>fname</i> (of format <i>form</i>) is loaded into memory at <i>offset</i> . <i>flag</i> indicates success or failure. <i>form</i> should be "bin", "hex", or "s19". <i>offset</i> should be a hexadecimal number ("00A5", "FF0", etc).
program	<i>saddr, eaddr</i>	<i>flag</i>	The previously configured EEPROM is programmed from address <i>saddr</i> to <i>eaddr</i> . <i>flag</i> indicates success or failure. <i>saddr</i> and <i>eaddr</i> should be hexadecimal numbers ("00A5", "FF0", etc).
programwsn	<i>saddr, eaddr</i>	<i>flag</i>	Identical to <i>program</i> , but writes the configured serial number to the buffer (configured in the GUI) and auto-increments it after a successful program.
verify	none	<i>flag</i>	Verifies that the buffer and device are equivalent. <i>flag</i> indicates this.
erase	none	<i>flag</i>	Erases the device. <i>flag</i> indicates success or failure.
blank	none	<i>flag</i>	Verifies whether or not the device is blank. <i>flag</i> indicates this.

Sample scripts

A number of sample scripts that can be modified as needed are available in the “Sample scripts” subdirectory of the program installation directory.