Application Note

Report No:

AN118

Title:

In-System Programming (ISP) of I2C Serial EEPROM Memory Devices using Equinox ISP Programmers

Author:	Date:	Version Number:
John Marriott	27th Mar 09	1.02

Abstract:

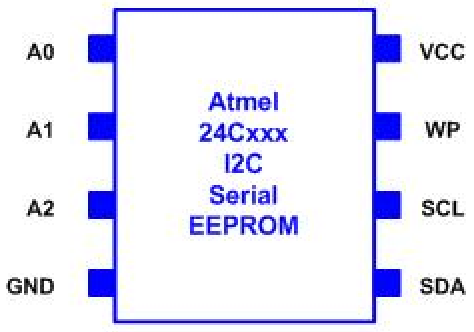
This Application Note describes how to implement In-System Programming (ISP) support for I2C Serial EEPROM Memory devices using an Equinox ISP Programmer. This support will allow the devices to be programmed in-situ on a Target Board using the I2C 2-wire programming interface. Multiple I2C devices can be programmed on the same Target Support by simply selecting a different I2C address in the Programming Project. The fastest possible programming speeds are guaranteed as the EEPROM data is stored locally within the programmer. Standalone operation of the programmer (no PC) via single auto-program key is also possible making this solution ideal for high-volume production applications.

Contents

1.0 Introduction

1.1 Overview

This Application Note describes how to implement In-System Programming (ISP) support for I2C Serial EEPROM Memory devices using Equinox ISP Programmers. This support will allow I2C EEPROM memory devices to be programmed in-situ on a Target Board using the I2C 2-wire programming interface. Multiple I2C devices can be programmed on the same Target Support by simply selecting a different I2C address in the Programming Project. The fastest possible programming speeds are guaranteed as the EEPROM data is stored locally within the programmer. Standalone operation of the programmer (no PC) via single auto-program key is also possible making this solution ideal for high-volume production applications.



Features

- In-System Programming (ISP) support for I2C Serial EEPROMs
- Programmer uses the I2C 2-wire interface to program the device in-situ on a Target Board
- Supports both 100 kHz and 400 kHz I2C operating frequencies
- Supports 'Standard' and 'Extended' I2C operating modes
- Multiple I2C Memories on the same I2C bus can be programmed
- Fast programming times due to local storage of data to be programmed (Standalone Mode)
- Supports Program/Verify per page or Program entire device and then verify
- Supports 3.0 to 5.0V Target Voltage operation
- Standalone programmer operation (no PC required) is possible
- PC software utility supports automatic serial number (up to 64-bit) generation and programming (chargeable upgrade)

1.2 Programmers supporting I2C Programming

The Equinox programmers which are capable of supporting programming of I2C Serial EEPROM devices are listed in the table below.

Programmer	Firmware required	Upgrade License code
EPSILON5	3.07 or above	EPSILON5-UPG16
FS2000A	Not supported	Not supported
FS2003	3.07 or above	FS2003-UPG16
FS2009	3.07 or above	FS2003-UPG16
PPM3 MK1	Not supported	Not supported
PPM3 MK2	3.07 or above	PPM3A1-UPG16
PPM4 MK1	3.07 or above	PPM4MK1-UPG16
ISPNano	To be advised	ISPNano-UPG16

Please note:

- The I2C Algorithms were implemented in ***Firmware version 3.07***. Please upgrade your programmer to this version or any version higher than this in order to use the I2C algorithms.
- This support requires that ***EQTools build 914*** or above is installed.
- A chargeable 'License upgrade' is required for all programmers to enable them to support programming of I2C Serial EEPROMs.

1.3 Programmer Firmware upgrade

The I2C programming algorithms were implemented in Firmware version 3.07.
Please upgrade your programmer to this version or any version higher than this in order to use the I2C algorithms.

For detailed instructions on how to upgrade your programmer firmware, please refer to:
"Application Note – AN112 – Firmware upgrade instructions for Equinox ISP programmers."

1.4 Device Support

For the latest Device Support – please refer to website.

1.5 Minimum hardware / software requirements

The ***'24xxx Serial EEPROM Memory'*** I2C algorithms require the following minimum versions of hardware and software:

1. Programmer hardware: Epsilon5, FS2003, FS2009, PPM3-MK2, PPM4-MK1, ISPNano

2. EQTools Software: build 914 or above
- Please download the latest EQTools software version from our website.
- For detailed instructions on how to upgrade your programmer firmware, please refer to:
"Application Note – AN112 – Firmware upgrade instructions for Equinox ISP programmers."

3. Programmer firmware: 3.07 or above
- Please download the latest firmware version from our website.

1.6 Upgrading your Equinox Programmer to support I2C

The ***'I2C Serial EEPROM Memory'*** algorithms are not supported as standard on Equinox programmers (but please see below for exceptions). It is necessary to purchase a 'License Upgrade' from Equinox to support these devices. Equinox will then send you an ***'Upgrade License String'*** which will upgrade your programmer to support I2C programming.

Please note: If you have purchased the ''EPSILON5(E2) - Portable ISP Programmer for Serial EEPROM Memory Devices'' or EPS-E2-BUNDLE the programmer is pre-loaded with this license and these steps do not need to be performed.

1.6.1 Purchasing a '24xxx Serial EEPROM Memory' License

All Equinox ISP programmers require the purchase of a 'License Upgrade' to enable ***'24xxx Serial EEPROM Memory'*** programming support.
Please see the table in section 1.2 for the relevant upgrade for your programmer.

1.6.2 How do I enable the programmer for '24xxx Serial EEPROM Memory' programming?

To enable your programmer to support ***'24xxx Serial EEPROM Memory'*** I2C algorithms, please purchase the relevant upgrade from Equinox or an Equinox distributor:

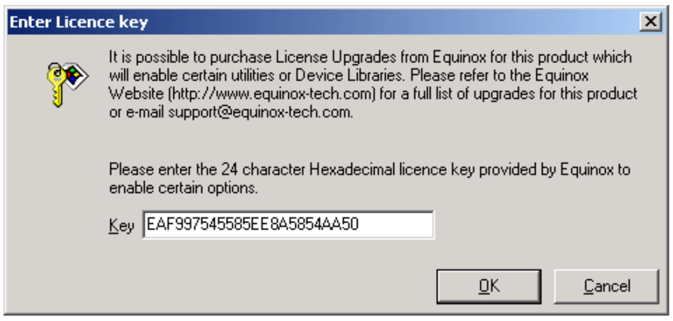
1. **If you purchase the upgrade directly from Equinox**
- Equinox will email you a 'License String'.
- This string can be entered directly into the ***<Enter License>*** screen in EQTools.

2. **If you purchase the upgrade from a distributor**
- The distributor will send you the 'Upgrade Pack' by courier.
- Within the 'Upgrade Pack' you will find an Upgrade Form with a Code String on it.
- Email this Code String plus your programmer Serial Number to support@equinox-tech.com
- Equinox will then send you a 'License String' which is keyed to your programmer Serial Number.
- This string can be entered directly into the ***<Enter License>*** screen in EQTools.

1.6.3 Entering the License String to upgrade your programmer

Once you have received the License Strings from Equinox, please follow the steps below to apply the upgrade to your programmer:

- Launch EQTools → The EQTools ***'Welcome Screen'*** is displayed.
- Close down the EQTools ***'Welcome Screen'***
- From the top menu bar, select ***<Programmer><Programmer Info>***
  → the Programmer Information screen is displayed
- Click the ***<Enter License>*** button
  → The ***<Enter License Key>*** screen is displayed.


Enter Licence key

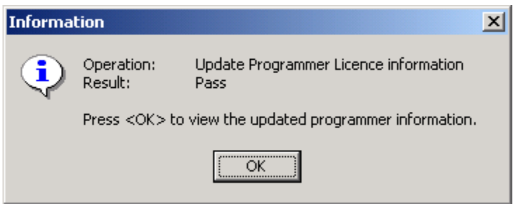
It is possible to purchase License Upgrades from Equinox for this product which will enable certain utilities or Device Libraries. Please refer to the Equinox Website (http://www.equinox-tech.com) for a full list of upgrades for this product or e-mail support@equinox-tech.com.

Please enter the 24 character Hexadecimal licence key provided by Equinox to enable certain options.

Key EAF997545585EE8A5854AA50

- Enter the License String you were sent by Equinox
- Click ***<OK>***

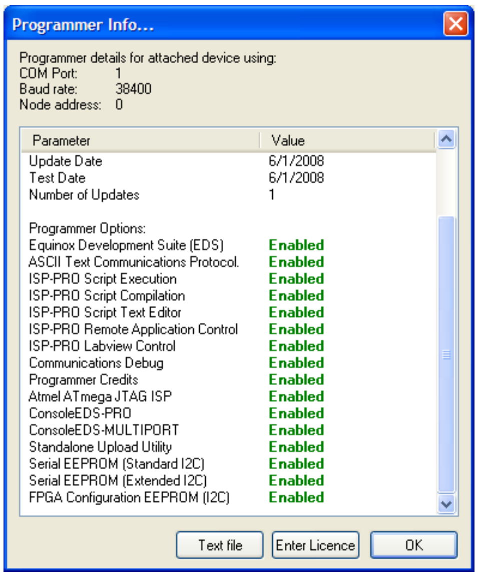
  → EQTools should acknowledge that the attached programmer has been upgraded.


Information

Operation: Update Programmer Licence information
Result: Pass

Press <OK> to view the updated programmer information.

- Click ***<OK>***
- If you now check the Programmer Info screen, you should find that the following entries are now ENABLED: ***Serial EEPROM (Standard I2C)*** and ***Serial EEPROM (Extended I2C)***


Programmer Info...

Programmer details for attached device using:
COM Port: 1
Baud rate: 38400
Node address: 0

Parameter	Value
Update Date	6/1/2008
Test Date	6/1/2008
Number of Updates	1
Programmer Options:	
Equinox Development Suite (EDS)	Enabled
ASCII Text Communications Protocol.	Enabled
ISP-PRO Script Execution	Enabled
ISP-PRO Script Compilation	Enabled
ISP-PRO Script Text Editor	Enabled
ISP-PRO Remote Application Control	Enabled
ISP-PRO Labview Control	Enabled
Communications Debug	Enabled
Programmer Credits	Enabled
Atmel ATmega JTAG ISP	Enabled
ConsoleEDS-PRO	Enabled
ConsoleEDS-MULTIPORT	Enabled
Standalone Upload Utility	Enabled
Serial EEPROM (Standard I2C)	Enabled
Serial EEPROM (Extended I2C)	Enabled
FPGA Configuration EEPROM (I2C)	Enabled

1.7 Technical Support

If you have any questions about using an Equinox ISP Programmer, please refer to the list of further information sources detailed below.

1. User Manual for your Programmer

2. On-line help
Press ***<F1>*** for help at any time when running EQTools or ISP-PRO.

The help system is context-sensitive. Simply press ***<F1>*** on any error message and the possible causes of the error should be listed. This help system is updated on a regular basis. Please see software update details for information on keeping up-to-date with software revisions.

3. Equinox Web Site
The support page for all Equinox ISP Programmers can be found at:
http://www.equinox-tech.com/products/downloadsearch.asp

4. E-mail
Please e-mail any technical support questions about this product to:
support@equinox-tech.com

5. Telephone
It is usually easier to answer Technical Support questions via e-mail.
However, if you wish to ask a question in person please call: **+44 (0) 1942 841975**

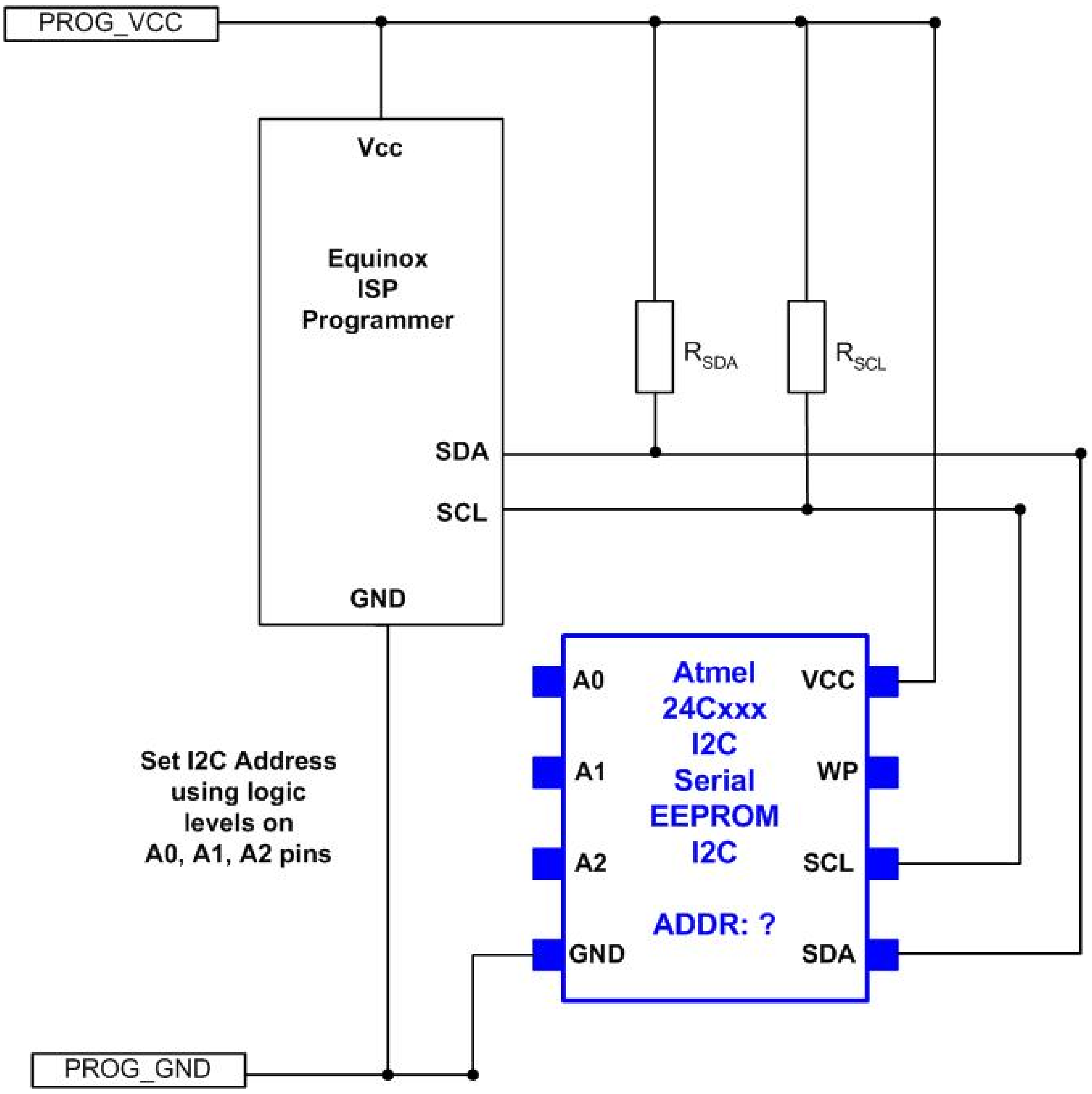
6. Fax
Please fax any technical support questions about this product to**: +44 (0) 1942 844181**

Equinox will try our best to answer your questions about this product as quickly as possible. However, we cannot promise an immediate reply. Please consult our web site for new software updates as the problem that you are enquiring about may have already been fixed in a new version.

2.0 I2C Hardware Connections

2.1 Overview

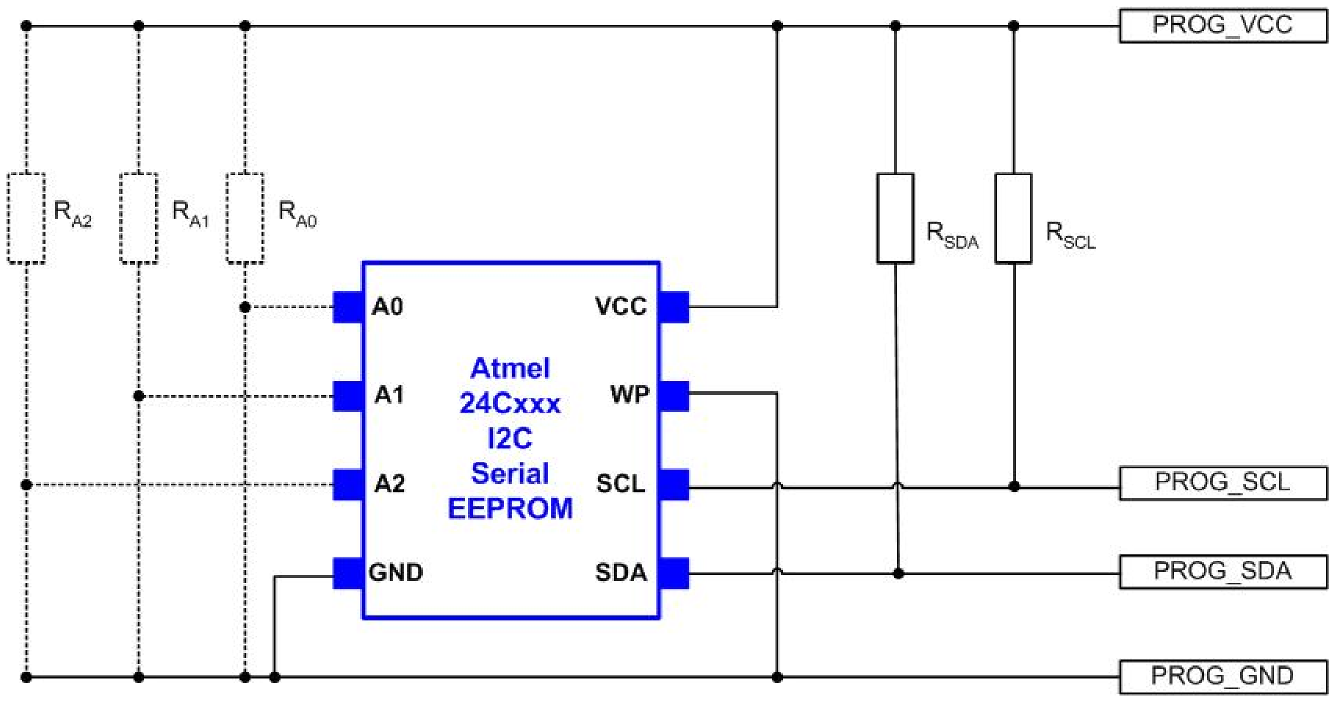
The 24xxx Serial EEPROM Memories use the 2-wire ***'I2C interface'*** for programming / reading the Target Device. This interface is made up a bi-directional ***'data'*** signal called ***SDA*** and a clock signal called ***'SCL'***. The programmer is always the ***'I2C Bus Master'*** during the programming process and so will always drive the clock line to the slave I2C device(s) on the I2C bus.


PROG_VCC
Vcc
Equinox ISP Programmer
SDA
SCL
GND
R_SDA
R_SCL
A0
A1
A2
GND
Atmel 24Cxxx I2C Serial EEPROM I2C
ADDR: ?
VCC
WP
SCL
SDA
Set I2C Address using logic levels on A0, A1, A2 pins
PROG_GND

It is also possible to have multiple I2C Devices on the I2C bus. The I2C connections to the programmer are identical in this scenario, but each I2C device then must have a different I2C Node Address.

2.2 Single I2C Device - Signal Connections

The schematic below shows the connections required to In-System Program (ISP) a single 24xxx Serial EEPROM Memory device using an Equinox ISP programmer.



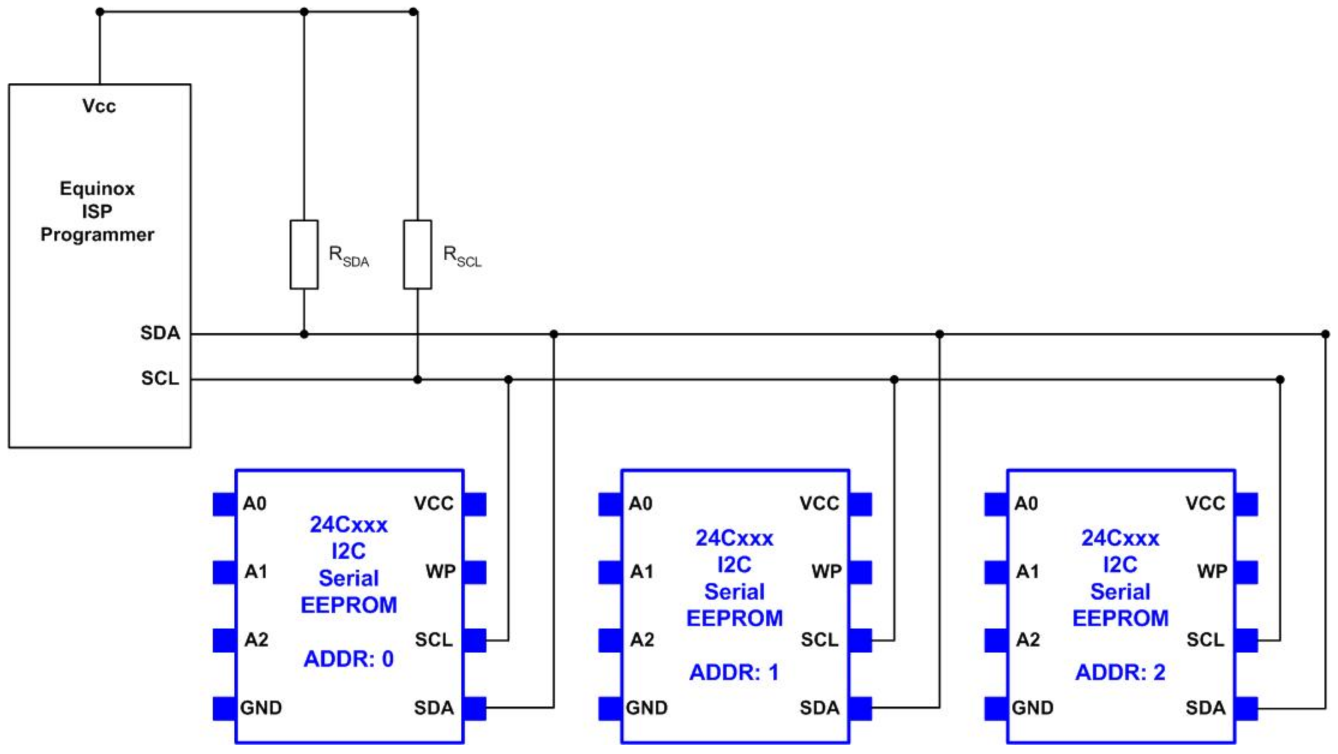
The connections to each pin on the 24xxx Serial I2C Memory device are detailed in the table below.

Signal Name	Signal description	Signal direction (from Programmer)	Equivalent SPI pin on ISP Header connector
PROG_SDA	I2C Serial Data	Bi-directional	PROG_SDA (MOSI)
PROG_SCL	I2C Serial Clock	Output	PROG_SCL (SCK)
A0..A1..A2	I2C Node Address	No connection to programmer	Not applicable
WP	Write Protect	No connection to programmer	Not applicable
Vcc	Vcc Voltage	Passive	PROG_VCC
GND / Vss	Ground / 0V connection	Passive	PROG_GND

Please refer to sections 2.4 for the values of the R_{SDA}, R_{SCL} pull-up resistors and section 2.5 for an explanation of the R_{A0}, R_{A1}, R_{A2} resistors.

2.3 Multiple I2C Device - Signal Connections

The schematic below shows the connections required to In-System Program (ISP) multiple 24xxx Serial EEPROM Memory devices on the same I2C bus using an Equinox ISP programmer.



- Each I2C Device must have a different ***'I2C Address'*** which is set using the A0, A1, A2 address pins.
- The programmer can then read / write the required device by specifying the I2C address. For clarity, the address line connections are not shown on this diagram.

The connections to each pin on the 24xxx Serial I2C Memory device are detailed in the table below.

Signal Name	**Signal description**	**Signal direction (from Programmer)**	**Equivalent SPI pin on ISP Header connector**
PROG_SDA	I2C Serial Data	Bi-directional	PROG_SDA (MOSI)
PROG_SCL	I2C Serial Clock	Output	PROG_SCL (SCK)
A0..A1..A2	I2C Node Address	No connection to programmer	Not applicable
WP	Write Protect	No connection to programmer	Not applicable
Vcc	Vcc Voltage	Passive	PROG_VCC
GND / Vss	Ground / 0V connection	Passive	PROG_GND

Please refer to sections 2.4 for the values of R_{SDA}, R_{SCL} pull-up resistors.

2.4 I2C Signals – pull-up resistors

In order to implement I2C programming using an Equinox ISP Programmer, it is necessary to add a pull-up resistor on both the I2C ***'SDA'*** and ***'SCL'*** signal lines – see R_{SDA} and R_{SCL} in the schematics shown in sections 2.2 and 2.3. The recommended values for these resistors are shown in the table below. The exact choice of resistor value is dependent on the Target Voltage and number of I2C devices on the I2C bus.

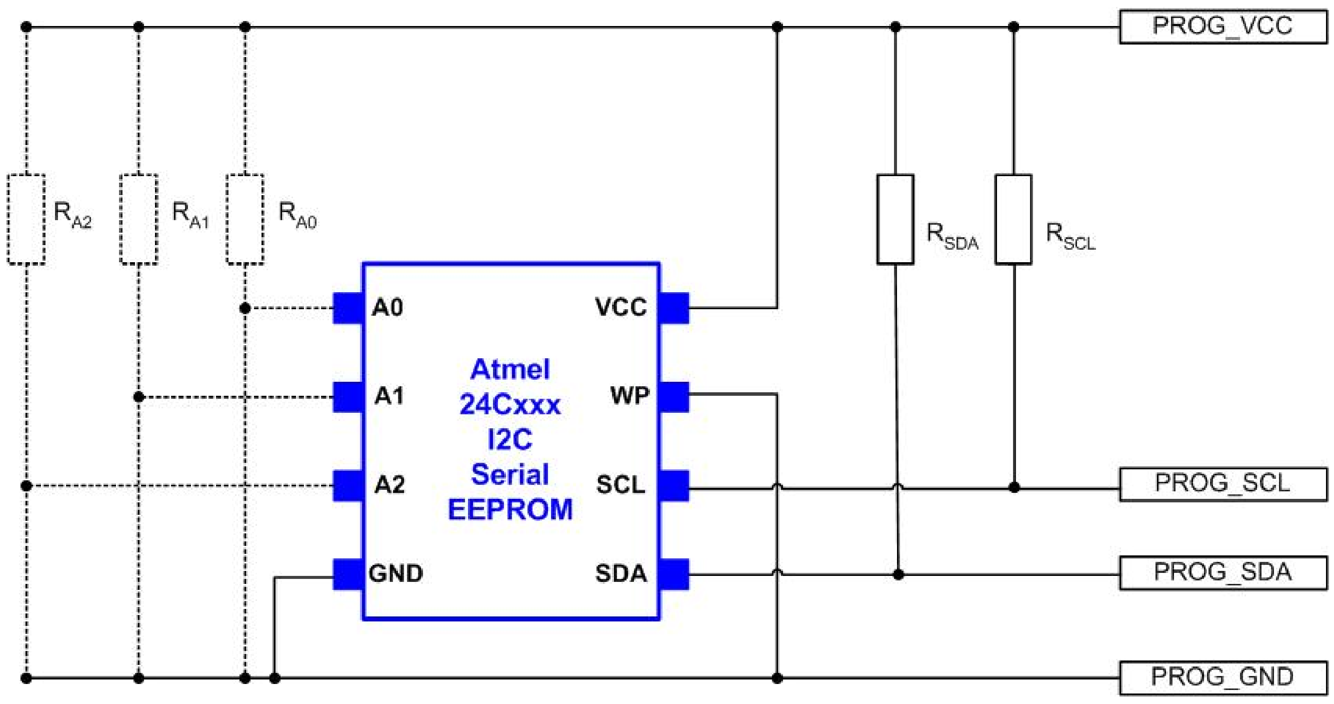
Pull-up resistor	3.3V	5.0V	Units
R_{SDA}	4.7k	4.7k	Ohms
R_{SCL}	4.7k	4.7k	Ohms

Please note:

- The pull-up resistors (R_{SDA} and R_{SCL}) may already by present on the Target System. If not, they could be added as part of the ISP Cable assembly.
- The pull-up resistors should be connected from the ***'SDA'*** and ***'SCL'*** signal lines to the ***'Target Vcc'*** line. This may not be the same Vcc that the programmer is sourcing its power from.

2.5 I2C Node Address

Each device on the I2C Bus must have a unique node address so that the programmer / application controller can communicate to each individual device by specifying this address. For 24xxx Serial EEPROM Memory devices, the ***'I2C Address'*** is selected by using the ***A0, A1, A2*** address pins. Many designers use pull-up resistors to ***Vcc*** on the ***A0, A1, A2*** pins to set a logic '1' or simply connect the pins to ***GROUND*** to set a logic '0'. Please refer to the schematic below – the resistors and R_{A0}, R_{A1}, R_{A2} are used to set the I2C address. The typical value of each resistor is 4k7 ohms.



Please note:

- The actual I2C address selected is dependent on the memory architecture of the device being programmed and the voltages on the ***A0, A1, A2*** address pins.
- Some devices do not use all of the address pins to set the I2C address. Please refer to the datasheet for the individual device for further information.
- The Equinox Programmer Software – EQTools – will display all the available I2C addresses for the selected device. It is then possible to select the relevant address which matches the address settings on your Target Device(s).
- The Equinox Programmer Software – EQTools – supports communication with any device on the I2C bus by simply changing the 'Device Signature' to the address of the I2C device which you want to program.

2.6 Target Voltage range

Equinox programmers support programming of I2C devices when the Target Voltage is between the voltages shown in the table below:

Parameter	Voltage limits
Minimum Voltage	3.0V
Maximum Voltage	5.0V

Each programmer features a ***'Line Driver Circuit'*** which interfaces the logic levels of the programmer with the logic levels of the Target System. To prevent damage to either the programmer or Target System, it is important to match the ***'Line Driver Voltage'*** to the ***'Target System Voltage'***.

Example 1 – Epsilon5 @ 3.3V
If the Target System is running at 3.3V, it is possible to power the programmer from the Target System by inserting the ***'Vcc Jumper Link' (J9)*** on the programmer. It is also possible to power the programmer at 3.3V through the Jack Socket and this will then power the Target System at 3.3V.

Example 2 - Epsilon5 @ 5.0V
If the Target System is running at 5.0V, it is possible to power the programmer from the Target System by inserting the ***'Vcc Jumper Link' (J9)*** on the programmer. It is also possible to power the programmer at 5.0V through the Jack Socket and this will then power the Target System at 5.0V.

Example 3 – FS2003 @ 3.3V
If the Target System is running at 3.3V, it is possible to power the programmer from the Target System by inserting the ***'Vcc Jumper Link' (J9)*** on the programmer.
Warning! – do not apply +9V to the DC Jack Socket when in this configuration because the programmer will then generate +5V and try to feed it to the Target System.

Example 4 – FS2003 @ 5.0V If the Target System is running at 5.0V, it is possible to power the programmer from the Target System by inserting the ***'Vcc Jumper Link' (J9)*** on the programmer. It is also possible to power the programmer at 9.0V through the Jack Socket and this will then power the Target System at 5.0V.

Please refer to the User Manual for each programmer for details of how to power the programmer and Target System.

2.7 ISP Header Connectors for I2C

The I2C algorithms on Equinox ISP programmers re-use the SPI ***'MOSI'*** and ***'SCK'*** signals for the I2C ***'SDA'*** and ***'SCL'*** signals respectively. It is therefore possible to use any of the ***'ISP Connectors'*** on an Equinox programmer to connect to an I2C Target System.

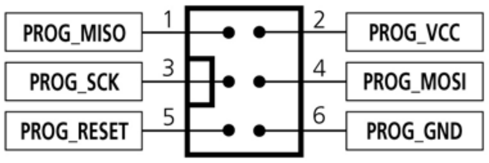
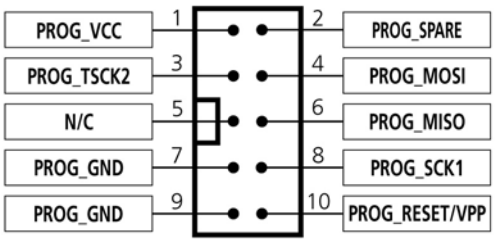
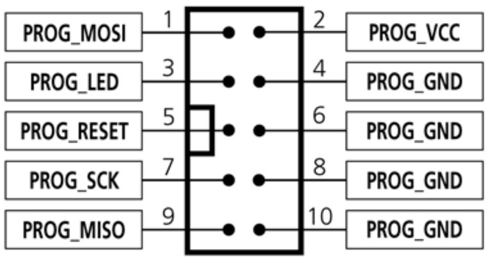
The I2C pin assignments to connect to a Target I2C Device are as follows:

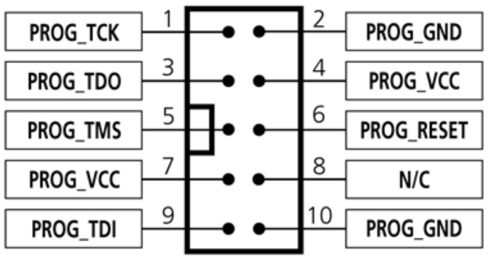
- The I2C ***'SDA'*** (data) signal corresponds to the SPI – ***'MOSI'*** signal.
- The I2C ***'SCL'*** (clock) signal corresponds to the SPI – ***'SCK'*** signal.

There is no recommended connector for the I2C interface, so the choice of connector often depends on the signal interface required on the Target System and also the physical amount of space available for this connector.

Please note:

- The I2C – ***'SDA'*** and ***'SCL'*** signals are available on each of these connectors as detailed in the table below.
- The smallest form-factor connector on an Equinox ISP Programmer is the ***'Atmel 6-way'*** header. This connector features the I2C SDA and SCL signals together with Vcc and Ground.

#	**ISP Header**	**Description / Function**	**ISP Header Pin-out**
1	J3	**Atmel 6-way ISP Header** • SDA - PROG_MOSI • SCL – PROG_SCK	 PROG_MISO 1 2 PROG_VCC PROG_SCK 3 4 PROG_MOSI PROG_RESET 5 6 PROG_GND
2	J6	**Equinox 10-way Header(a)** • SDA - PROG_MOSI • SCL – PROG_SCK	 PROG_VCC 1 2 PROG_SPARE PROG_TSCK2 3 4 PROG_MOSI N/C 5 6 PROG_MISO PROG_GND 7 8 PROG_SCK1 PROG_GND 9 10 PROG_RESET/VPP
3	J7	**Atmel 10-way Header** • SDA - PROG_MOSI • SCL – PROG_SCK	 PROG_MOSI 1 2 PROG_VCC PROG_LED 3 4 PROG_GND PROG_RESET 5 6 PROG_GND PROG_SCK 7 8 PROG_GND PROG_MISO 9 10 PROG_GND

4	J8	**Atmel 10-way JTAG Header** • SDA - PROG_TDO • SCL – PROG_TCK	 PROG_TCK 1 2 PROG_GND PROG_TDO 3 4 PROG_VCC PROG_TMS 5 6 PROG_RESET PROG_VCC 7 8 N/C PROG_TDI 9 10 PROG_GND

Please note:

- ***PROG_VCC*** is the voltage at which the programmer Line Driver Circuitry is powered. This is usually connected to the 'Target VCC'.
- ***PROG_GND*** is the 0V / GROUND connection for both the programmer and Target System.

3.0 Creating an I2C Programming Project

3.1 Overview

This section describes how to create a ***'Programming Project'*** for any 24xxx Serial EEPROM using the ***'I2C – Serial Programming Algorithm'***. During development, the EDS utility can be used to interactively test your project under PC control. Once the project is fully tested, it can then be exported as a 'Standalone Project' and then uploaded to a programmer. This allows the programmer to be used in 'Standalone Mode' (no PC required).

The programming of 24xxx Serial EEPROM memories requires that ***EQTools build 914*** or above is installed on your PC and that ***firmware 3.07*** is installed in the attached programmer(s).

3.2 Information required to create an I2C Project

The following information about the Target Device(s) to be programmed and the Target System is required in order to create an I2C Programming Project:

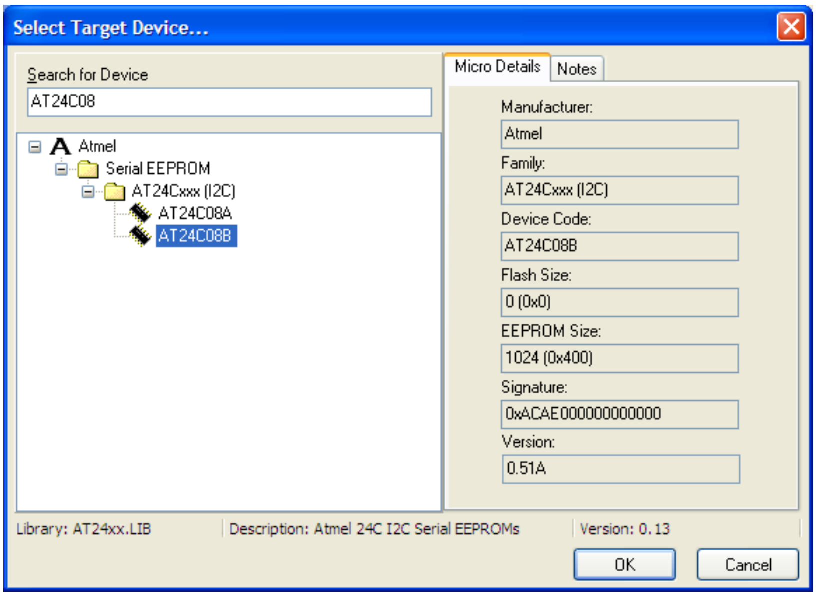
#	Information / data required	Example
1	Serial EEPROM Part Code	AT24C08
2	I2C connections / connector on Target board	4-pin I2C connector
3	I2C Programming configuration	i. Single I2C device or ii. Multiple I2C devices each with a different address on the I2C bus.
4	I2C Device address	The 'I2C Address' of each device on the I2C bus is required.
6	I2C Pull-up resistor value on SDA and SCL lines	e.g. 4k7 ohms
6	Target System Vcc voltage	e.g. 3.3V
7	Target System maximum current consumption	e.g. 100mA
8	EEPROM area 'Data File'	Binary (*.bin) or Intel Hex (*.hex)

3.3 Creating an EDS (Development project)

The simplest way to create a Programming Project for an I2C device is to use the EDS (Development Mode) Wizard as follows:

3.3.1 Launching EDS and selecting a Target Device

- Launch EQTools
- Select **<Create a new Development (EDS) Project**> → the EDS (Development) Wizard will launch
- Click **<Next>** → the **<Select Target Device>** screen will be displayed.



Search for Device: AT24C08

Atmel
Serial EEPROM
AT24Cxxx (I2C)
AT24C08A
AT24C08B

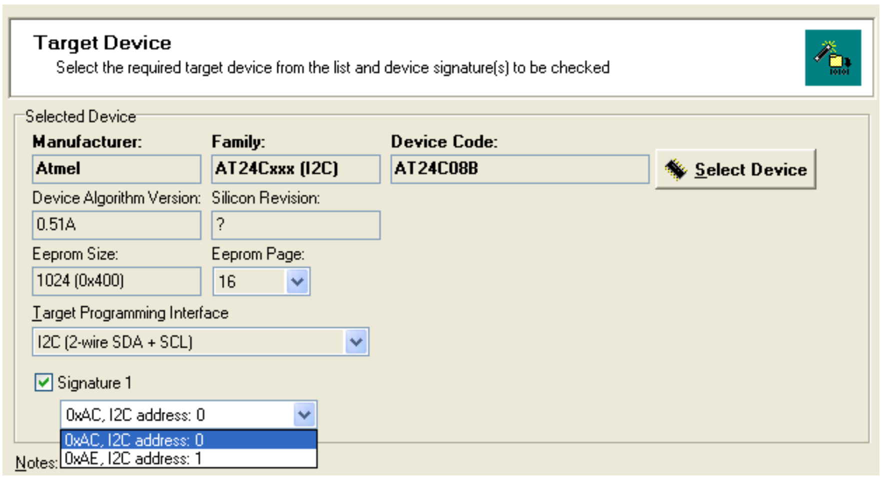
Manufacturer: Atmel
Family: AT24Cxxx (I2C)
Device Code: AT24C08B
Flash Size: 0 (0x0)
EEPROM Size: 1024 (0x400)
Signature: 0xACAE000000000000
Version: 0.51A

Library: AT24xx.LIB | Description: Atmel 24C I2C Serial EEPROMs | Version: 0.13

- Type the required device into the ***'Search for Device'*** field → a list of matching devices are displayed in the box below.
- Select the required device from the list and then click **<OK>** → the device is now selected.
- On the next screen, check that the device selection and all other device parameters are correct
- Click **<Next>** to advance to the next screen

3.3.2 Changing the I2C Device Address

Once the Target Device has been selected, it is then possible to set up the ***'I2C Address'*** of the device on the I2C bus. Each Serial EEPROM device has a range of possible I2C Addresses which depends on the memory size of the device. The possible addresses for the selected I2C device are show in the list box ***'Signature 1'*** – see screenshot below.



Target Device

Select the required target device from the list and device signature(s) to be checked

Selected Device

Manufacturer: Atmel

Family: AT24Cxxx (I2C)

Device Code: AT24C08B

Device Algorithm Version: 0.51A

Silicon Revision: ?

Eeprom Size: 1024 (0x400)

Eeprom Page: 16

Target Programming Interface: I2C (2-wire SDA + SCL)

☑ Signature 1: 0xAC, I2C address: 0

0xAC, I2C address: 0

0xAE, I2C address: 1

For the Atmel 24C08B device, there are two possible I2C addresses as follows:

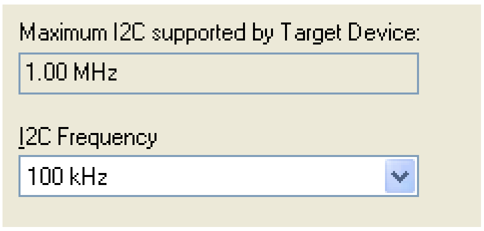
- 0xAC, I2C address: 0
- 0xAE, I2C address: 1

The actual address is dictated by the hardware setting of the address pins on the 24C08B device on the actual Target System. Please consult the circuit diagram of your Target System in order to work out the correct I2C address.

- Select the correct I2C address
- Click ***<Next>*** to move onto the next screen

3.3.3 Selecting the I2C Frequency

It is possible to select the ***'I2C Frequency'*** on this screen.
Some I2C EEPROMs only operate at 100 kHz, but many newer devices now support operation at up to 400 kHz. EQTools will automatically display the maximum I2C frequency supported by the Target Device. The slowest I2C frequency will be selected by default.



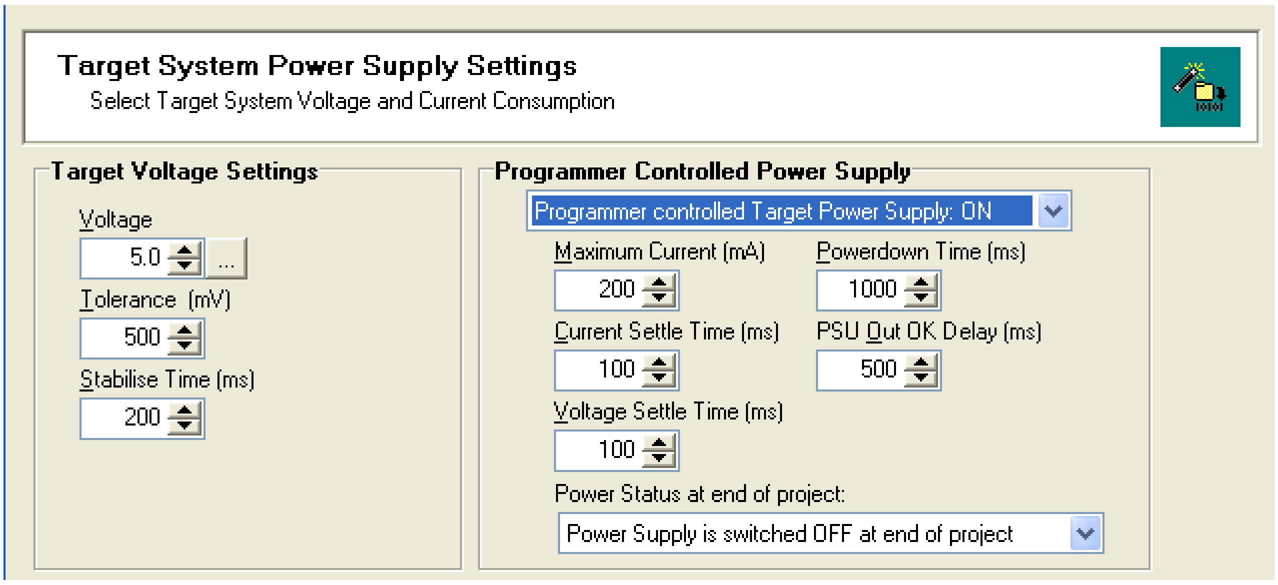
Maximum I2C supported by Target Device: 1.00 MHz

I2C Frequency: 100 kHz

For development purposes, select the slowest I2C frequency to ensure reliable programming. The frequency can be increased at a later stage once reliable operation has been proven.

3.3.4 Target System – Power Supply Settings

This screen allows you to set up the Power Supply characteristics of your Target System.



Target System Power Supply Settings
Select Target System Voltage and Current Consumption

Target Voltage Settings
Voltage: 5.0
Tolerance (mV): 500
Stabilise Time (ms): 200

Programmer Controlled Power Supply
Programmer controlled Target Power Supply: ON
Maximum Current (mA): 200
Powerdown Time (ms): 1000
Current Settle Time (ms): 100
PSU Out OK Delay (ms): 500
Voltage Settle Time (ms): 100
Power Status at end of project: Power Supply is switched OFF at end of project

i. Select the Target Voltage

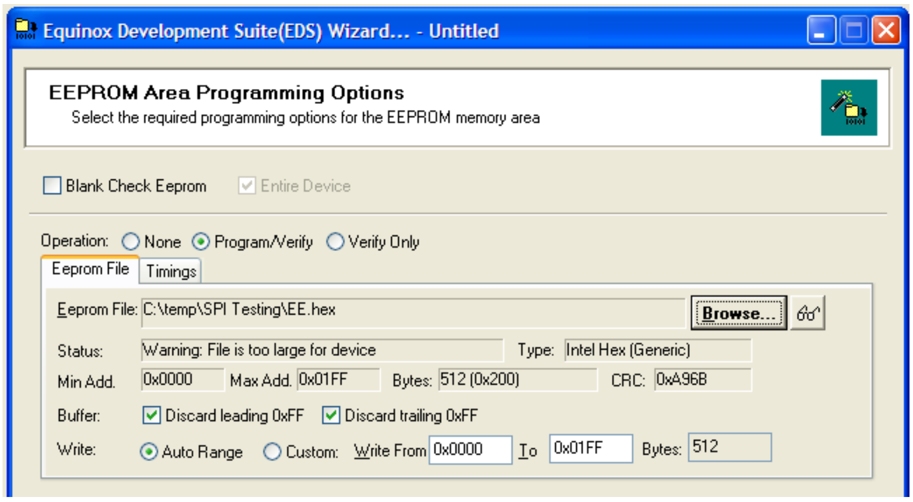
- This should be the voltage at which the Target System is being powered during the programming operation.
- Set the 'Voltage Tolerance' to be as wide as possible e.g. 500mV to allow for power supply variations. If the programmer is powering the Target System, this will also give a faster power-up time.
- It may be possible to power just the Target Microcontroller rather than the entire Target System.

ii. Set up the Target Powering and current parameters

- This option is only available for the PPM3-MK2 programmer.
- If the programmer is to power the Target System, select ***<Programmer controlled Target Power Supply: ON>***
- Set the ***'Maximum Current'*** to the maximum possible current which the Target System could draw from the programmer.
- Leave all other settings as default.

3.3.5 Specifying the EEPROM (Data) File

This screen allows you to specify the EEPROM (data) file which is to be programmed into the EEPROM area of the Target Device. This is an optional step – you can also specify the file once you are in the Development Suite (EDS).



Equinox Development Suite(EDS) Wizard... - Untitled

EEPROM Area Programming Options
Select the required programming options for the EEPROM memory area

☐ Blank Check Eeprom ✓ Entire Device

Operation: ○ None [✓ Program/Verify] ○ Verify Only

Eeprom File | Timings

Eeprom File: C:\temp\SPI Testing\EE.hex [Browse...]

Status: Warning: File is too large for device Type: Intel Hex (Generic)

Min Add. 0x0000 Max Add. 0x01FF Bytes: 512 (0x200) CRC: 0xA96B

Buffer: ✓ Discard leading 0xFF ✓ Discard trailing 0xFF

Write: [✓ Auto Range] ○ Custom: Write From 0x0000 To 0x01FF Bytes: 512

i. Blank Check the EEPROM

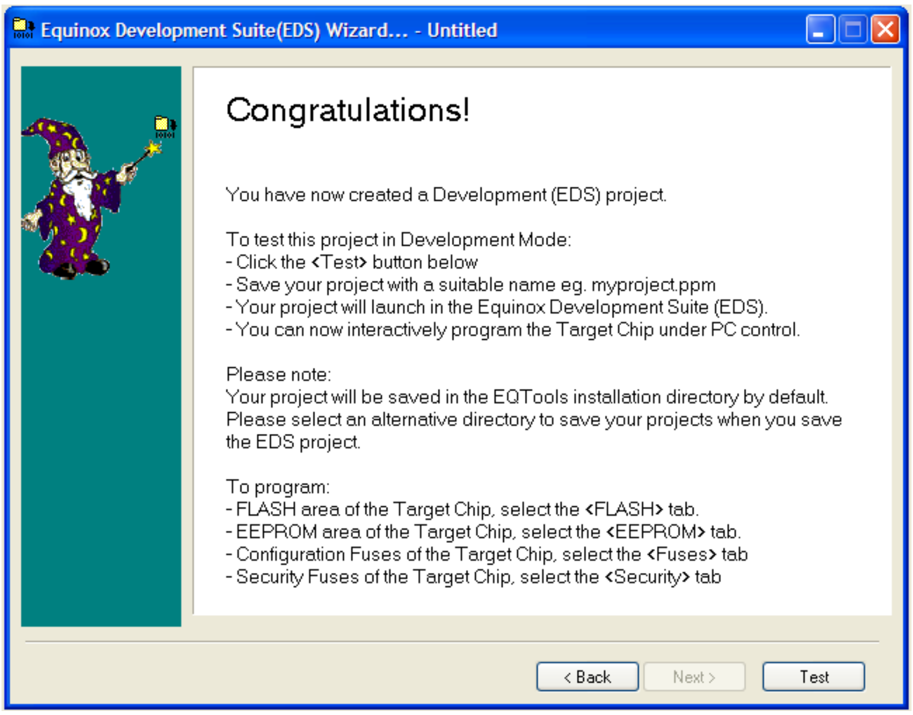
- Most Serial EEPROM devices feature an automatic erase of each BYTE or PAGE during the programming process. It is therefore not necessary to perform a ‘Blank Check’ operation.
- If you want to be absolutely sure the EEPROM is blank, you can enable the ‘Blank Check EEPROM’ option. This will perform a full Blank Check of the EEPROM area to check that all locations are set to 0xFF.
- Warning – this check can be time-consuming and will increase the overall programming time!

ii. Selecting the EEPROM File

- Click the **<Browse>** button
- Browse to and select the file you wish to load and then select **<OK>**
- If the input file is a BINARY file then the wizard will load the data in from file starting at address 0x0000 and continuing contiguously to the end of the file.
- If the input file is an INTEL HEX file then the wizard will load in from file from the start address specified in the file to end address specified in the file.
- In the example above, the EEPROM file contains more data than the physical EEPROM size of the Target Chip. The input data will therefore only read the data in to the address range of the EEPROM.
- The granularity of the EEPROM Memory depends on the device being programmed and can be 1, 4, 8, 16, 32, 64, 128 or 256 bytes. This is known as the ‘EEPROM Page Size’. This means that the programmer will always program in pages of the relevant number of bytes. Your input file will therefore be rounded up to the nearest ‘Page Size’.

3.3.6 Launching EDS at the end of the EDS Wizard

Once you reach the end of the EDS Wizard, click the **<Test>** button to launch the project in the Equinox Development Suite (EDS).



Congratulations!

You have now created a Development (EDS) project.

To test this project in Development Mode:
- Click the <Test> button below
- Save your project with a suitable name eg. myproject.ppm
- Your project will launch in the Equinox Development Suite (EDS).
- You can now interactively program the Target Chip under PC control.

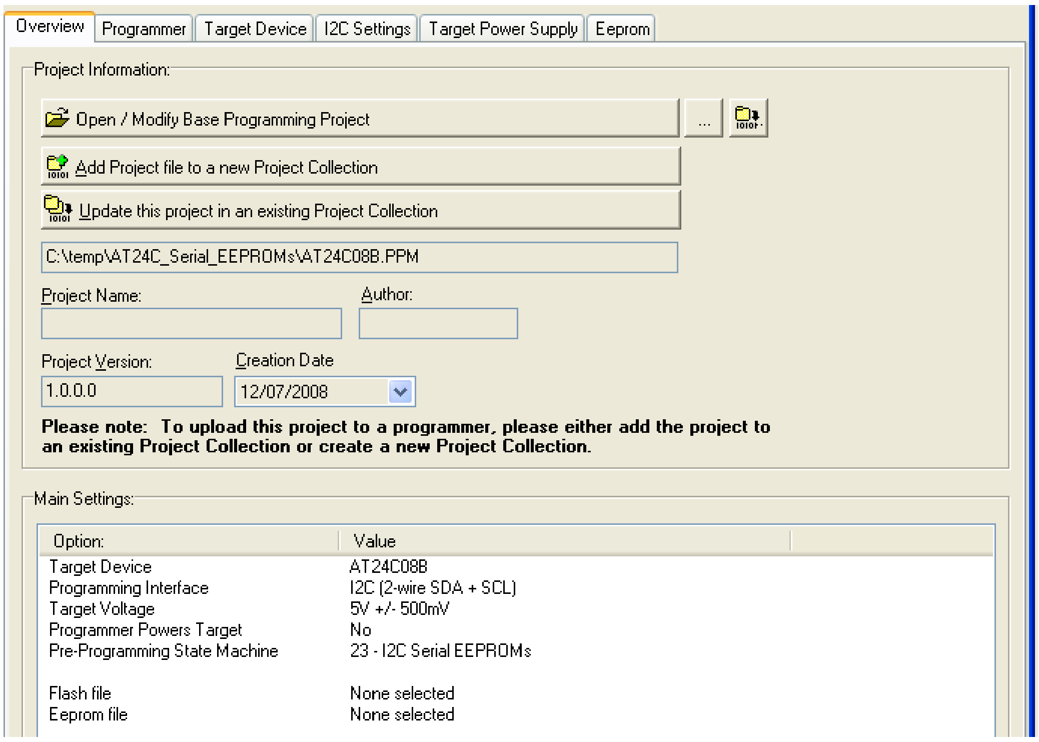
Please note:
Your project will be saved in the EQTools installation directory by default.
Please select an alternative directory to save your projects when you save the EDS project.

To program:
- FLASH area of the Target Chip, select the <FLASH> tab.
- EEPROM area of the Target Chip, select the <EEPROM> tab.
- Configuration Fuses of the Target Chip, select the <Fuses> tab
- Security Fuses of the Target Chip, select the <Security> tab

Enter a name for the EDS project e.g. ***AT24C08B*** and click **<Save>**
→ Your project will now launch in EDS Mode.

3.4 Testing an I2C Project in Development (EDS) Mode

If you have clicked the **<Test>** button at the end of the EDS Wizard, then an EDS (Development Mode) session will now launch.


Overview
Programmer
Target Device
I2C Settings
Target Power Supply
Eeprom
Open / Modify Base Programming Project
Add Project file to a new Project Collection
Update this project in an existing Project Collection

Project Information:

C:\temp\AT24C_Serial_EEPROMs\AT24C08B.PPM

Project Name:
Author:
Project Version: 1.0.0.0
Creation Date: 12/07/2008

Please note: To upload this project to a programmer, please either add the project to an existing Project Collection or create a new Project Collection.

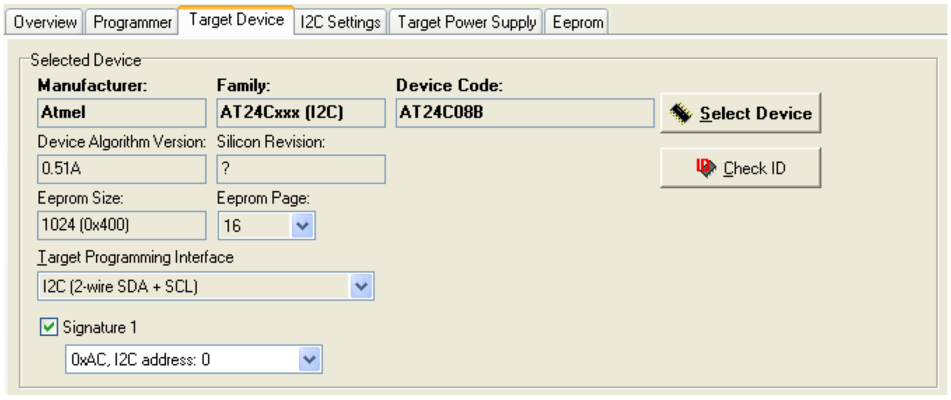
Main Settings:

Option:	Value
Target Device	AT24C08B
Programming Interface	I2C (2-wire SDA + SCL)
Target Voltage	5V +/- 500mV
Programmer Powers Target	No
Pre-Programming State Machine	23 - I2C Serial EEPROMs
Flash file	None selected
Eeprom file	None selected

3.5 Testing I2C communication with the Target IC

To make sure that the programmer can communicate to the Target I2C device, try reading back the Device Signature as follows:

- Select the **<Target Device>** tab



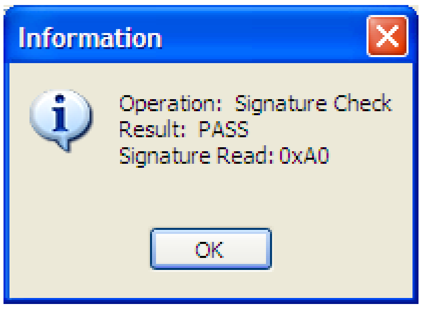
Selected Device

Manufacturer: Atmel
Family: AT24Cxxx (I2C)
Device Code: AT24C08B
Device Algorithm Version: 0.51A
Silicon Revision: ?
Eeprom Size: 1024 (0x400)
Eeprom Page: 16
Target Programming Interface: I2C (2-wire SDA + SCL)
☑ Signature 1: 0xAC, I2C address: 0

- Check that the ***'Signature 1'*** field is enabled and the correct ***'I2C Address'*** is selected for the I2C Device which you want to program.
- Click the **<Check ID>** button on the right-hand side of the screen.

→ The programmer will now try to communicate with the Target Device via the I2C Interface.

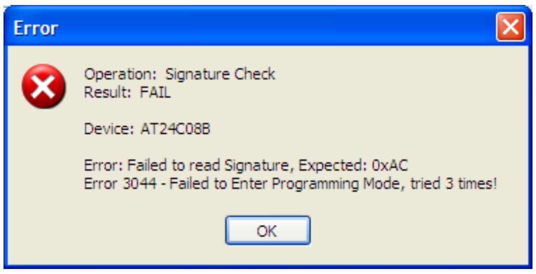
→ If the Target IC responds correctly, then EDS will report ***'Signature Read: Pass'***



Operation: Signature Check
Result: PASS
Signature Read: 0xA0

→ If the Target IC does not respond, then EDS will report either:

i. Cannot enter programming mode



Operation: Signature Check
Result: FAIL

Device: AT24C08B

Error: Failed to read Signature, Expected: 0xAC
Error 3044 - Failed to Enter Programming Mode, tried 3 times!

If you receive this error, it means that the programmer could not establish communications with the specified I2C device.

Please check the following:

- The I2C connections between the programmer and the Target System are correct.
- There is definitely power applied to the Target System and to all the I2C devices.
- The ***'I2C Frequency'*** settings are correct for the Target Device being programmed.

ii. 'Signature Read: Fail'.

If you receive this error, please check the following:

- The correct ***'I2C address'*** for the Target IC has been selected.
- Try changing the ***'I2C address'*** to an alternative address and then click the **<Check ID>** button again.

3.6 Programming multiple I2C Memories on same I2C bus

It is possible to have many 24xxx Serial EEPROM Memory devices on the same I2C Bus. EDS can only communicate with a single I2C Device at a specified address at any given time. However, it is possible to communicate with any I2C memory on the I2C Bus at any address by simply changing the ***'Target Device'*** and ***'I2C address / Signature'*** in EDS. The simplest method to cope with multiple I2C Memories on the same I2C bus is to make a separate Programming Project for each device.

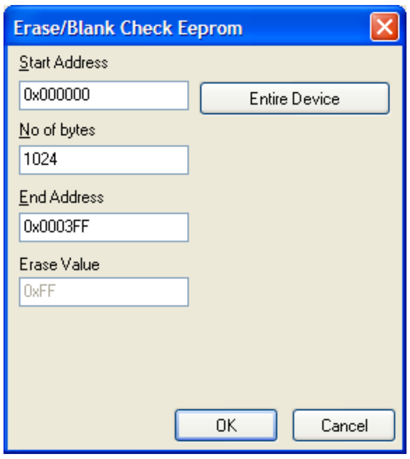
3.7 Erase / Blank Check of the EEPROM Area

The 24xxx Serial EEPROM Memory devices do not feature a global ***'Chip Erase'*** command. Instead each byte is automatically set to 0xFF by the device before being programmed to a new value. This effectively means that it is not necessary to erase an area of Serial EEPROM Memory before programming it as the new data will automatically erase and over-program the old data.

If you do wish to erase all or a block of EEPROM memory, the programmer must write the value 0xFF into the required address range. This will have the same effect as performing an Erase operation. In EDS, the ***'Erase'*** and ***'Blank Check'*** operations have been combined into a single operation called ***'Erase / Blank Check'***.

To perform an ***'Erase / Blank Check'*** operation:

- Select the <EEPROM> tab
- Click the ***<Erase / Check>*** button
  → The ***'Erase / Blank Check'*** dialog box is displayed.



Erase/Blank Check Eeprom

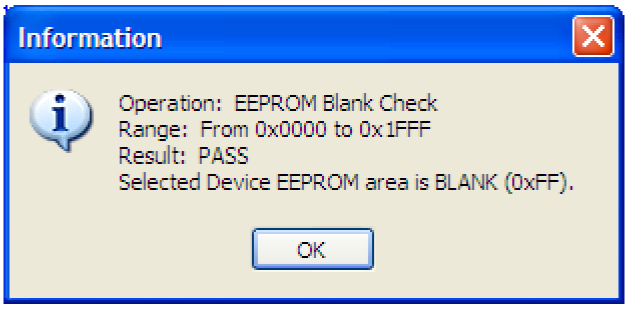
Start Address: 0x000000

No of bytes: 1024

End Address: 0x0003FF

Erase Value: 0xFF

Select the address range you wish to Erase / Blank Check (Click <Entire Device> to select the complete address range of the selected IC)
Click <OK> to perform the ***'Erase / Blank Check'*** operation
→ The BUSY light should now illuminate on the programmer.
→ The programmer will report either a PASS or FAIL.



Information

Operation: EEPROM Blank Check
Range: From 0x0000 to 0x1FFF
Result: PASS
Selected Device EEPROM area is BLANK (0xFF).

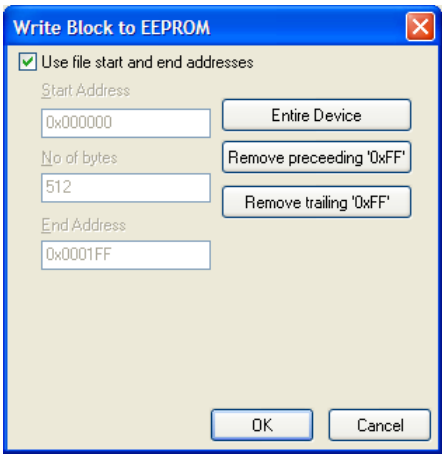
Please note:
Performing the the ***'Erase / Blank Check'*** operation can be time-consuming for larger EEPROM size devices so this method of erasure is not recommended for production users.

3.8 Programming the EEPROM Area

The following instructions describe how to program the contents of a file into the EEPROM area of the Target IC:

To program the EEPROM area:

- Select the **<EEPROM>** tab
- If you have not already selected a data file to program, click the ***'Edit buffer'*** check box and then click the **<Load>** button to select a suitable Binary of Intel Hex file.
- The contents of the specified file should now be displayed in the Buffer Window.
- Click the **<Write>** button


Write Block to EEPROM

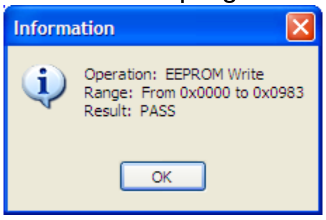
☑ Use file start and end addresses

Start Address: 0x000000

No of bytes: 512

End Address: 0x0001FF

- Select the address range you wish to program
- EDS will automatically use the ***'Start'*** and ***'End'*** address of the EEPROM input file unless otherwise specified. This reduces the total data actually programmed to the number of bytes in the input file rounded to the end of the nearest EEPROM Page.
- If you want to program the entire EEPROM range, click the **<Entire Device>** button.
- Click **<OK>** to program the EEPROM of the Target Chip.
- The programmer should now start to program the chip.
- The ***BUSY*** LED will illuminate on the programmer.
- The programmer will program the contents of the Buffer Window into the EEPROM area of the Target Device. The EDS progress bar should now move from left to right to indicate that the Target IC is being programmed.
- Once the programming operation is complete, EDS will display the following message:


Information

Operation: EEPROM Write
Range: From 0x0000 to 0x0983
Result: PASS

- The EEPROM data is programmed in pages and then verified so if a failure occurs it will be notified immediately.

If EDS reports an 'EEPROM programming error', please check the following:

- The ***'I2C Frequency'*** settings are correct for the Target Device being programmed.
- The correct value of pull-up resistor has been used on the I2C ***SDA*** and ***SCL*** signal lines.

4.0 Using Standalone Programming Projects

4.1 Overview

Once you have fully tested your project in ***EDS*** (Development Mode), it is then possible to make this project into a ***'Standalone Project'*** which can be uploaded to a programmer. This single standalone project file (*.prj) will contain all the information required to program the Target Device including the EEPROM file etc.

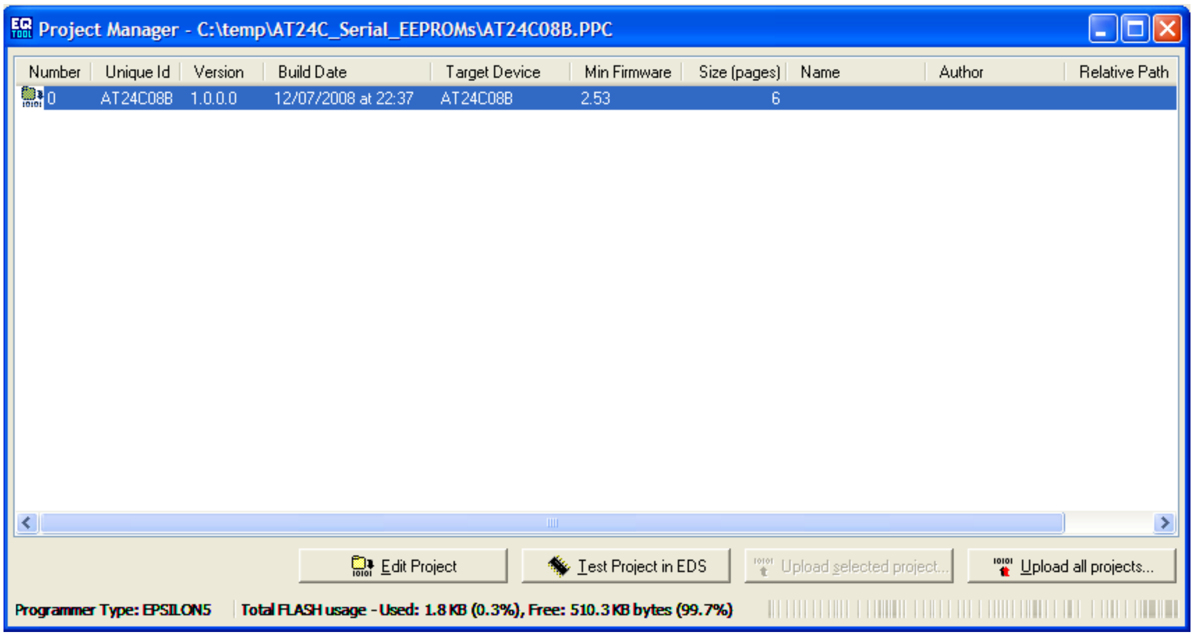
4.2 Creating a Standalone Project from EDS (Development Mode)

- In EDS (Development Mode), select the ***<Overview>*** tab
- Click the ***<Add Project File to a new Project Collection>*** button

→ The Project File is added to a new Project Collection

- Save the Project Collection with a suitable name e.g. ***AT24C08.PPC***

→ Project Manager will now launch


Project Manager - C:\temp\AT24C_Serial_EEPROMs\AT24C08B.PPC
Edit Project
Test Project in EDS
Upload selected project...
Upload all projects...

Number	Unique Id	Version	Build Date	Target Device	Min Firmware	Size (pages)	Name	Author	Relative Path
0	AT24C08B	1.0.0.0	12/07/2008 at 22:37	AT24C08B	2.53	6			

Programmer Type: EPSILON5 Total FLASH usage - Used: 1.8 KB (0.3%), Free: 510.3 KB bytes (99.7%)

The project which you have just added should be displayed as the first and only project in the Project Manager window.

4.3 Uploading a Standalone Project to a programmer

To upload the project to the programmer:

- Click the ***<Upload all projects>*** button
  → uploads all the projects in the collection to the programmer.

or

- Click once on the project you wish to upload in the ***Project Manager*** window and then click the ***<Upload selected project>*** button
  → uploads only the selected project in the collection to the programmer.

Follow the on-screen ***Upload Wizard*** instructions to complete the uploading of the project(s) to the programmer(s).

4.4 Re-testing a Project in EDS (Development mode)

If you want to re-test a Programming Project in EDS (Development Mode), the simplest method to do this is as follows:

- Use ***Project Manager*** to open your ***Project Collection (*.ppc file)***
- Click once on the project which you wish to test in EDS mode. This will select the project.
- Click the ***<Test in EDS>*** button
  → the selected project will now be opened in EDS (Development Mode).
- You can now test your project in EDS (Development Mode).

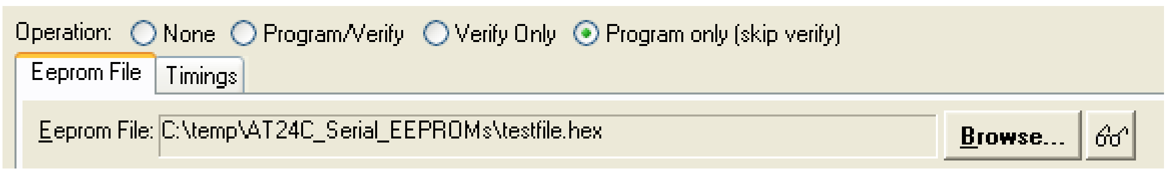
4.5 Optimising a Standalone Project for speed

When a Serial EEPROM project is first created, the programming mode defaults to ***'Program / Verify'*** mode where the programmer programs a page and then verifies each page of the device until the specified address range has been programmed. This method works very well but can be slow for larger memory size devices as the programmer has to constantly reset the EEPROM address pointer.

The optimum programming speed can be obtained by setting the programmer to ***'Program only'***. In this mode, the programmer will program the entire device address range first without performing a read-back / verify per page. A second Standalone Project set to ***'Verify Only'*** is then used to verify that all the data has been programmed correctly. The two projects can be 'chained' together so that they automatically execute one after the other.

To create a set of Chained Projects to program an EEPROM device:

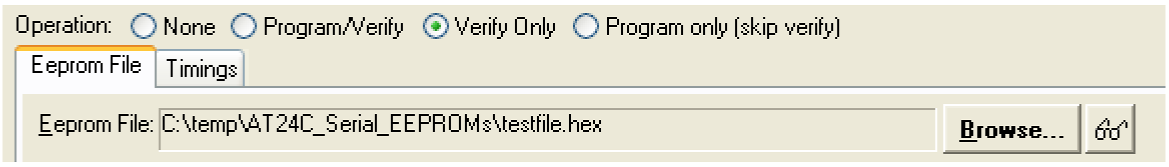
- Open your project in Project Builder
- Go to the ***<EEPROM>*** tab
- Set the ***"Operation"*** to ***"Program only (skip verify)"***



Operation: None Program/Verify Verify Only [✓ Program only (skip verify)]

Eeprom File: C:\temp\AT24C_Serial_EEPROMs\testfile.hex

- Compile the project and then select ***<Update project in new Project Collection>***
- Now save this project under a different name e.g. ***"AT24C08-VER.ppm"*** by opening the project and then selecting ***<File><Save as>.***
- Go to the ***<EEPROM>*** tab
- Set the ***"Operation***" to ***"Verify only"***

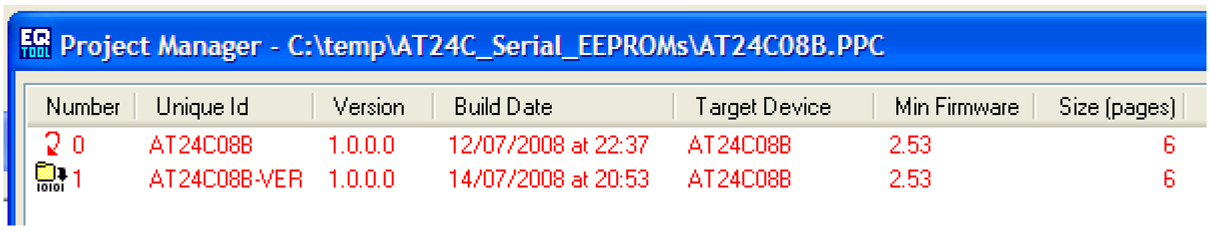


Operation: ○ None ○ Program/Verify [✓ Verify Only] ○ Program only (skip verify)

Eeprom File | Timings

Eeprom File: C:\temp\AT24C_Serial_EEPROMs\testfile.hex

- Compile the project and update in your Project Collection
- You should now have two projects in your Project Collection e.g. ***"AT24C08"*** and ***"AT24C08-VER"***.
- Click the ***'Chain Project'*** icon on the top icon bar → this will chain the two projects together so the second project will automatically execute after the first project.



Project Manager - C:\temp\AT24C_Serial_EEPROMs\AT24C08B.PPC

Number	Unique Id	Version	Build Date	Target Device	Min Firmware	Size (pages)
0	AT24C08B	1.0.0.0	12/07/2008 at 22:37	AT24C08B	2.53	6
1	AT24C08B-VER	1.0.0.0	14/07/2008 at 20:53	AT24C08B	2.53	6

- Upload the Project Collection to the programmer
- The Chained Projects will now show as a single project in the programmer.
- Click the <Yes> button on the programmer to execute the projects:
  → Project1 executes → This project only programs the data into the EEPROM area.
  → Project2 executes → This project performs a full verify pass of the data programmed into the Target Device by Project1.

This method offers significant time savings when programming large EEPROM devices.