

Report No:

AN110

Title:

ASCII Text Control (ATC) Protocol for Remote Control of Equinox Programmers

Author:

John Marriott

Date:

27th March 2009

Version Number:

1.06

Abstract:

This application note describes the 'ASCII Text Control (ATC)' Protocol for Remote Control of Equinox programmers. This protocol can be used to control an Equinox programmer from a Remote Application or Remote System using a simple set of ASCII commands. This protocol is ideal for controlling Equinox programmers from any Remote System which features an RS-232 serial port and which can operate at a fixed baud rate of 38,400.

Please note:

This protocol only allows ONE Equinox programmer to be controlled at a time and only allows the remote starting and status polling of 'Programming Projects'. The protocol does NOT support direct reading / writing of data from Target Devices or direct control of the actual programming functions.

All rights are reserved. Reproduction in whole or in part is prohibited without the prior written consent of the copyright owner. The information presented in this document does not form part of any quotation or contract, is believed to be accurate and reliable and may be changed without prior notice. No liability will be accepted by the publisher for any consequence of its use. Publication thereof does not convey nor imply any license under patent or other industrial or intellectual property rights

Contents

1.0 Introduction	3
1.1 Features of the ASCII Protocol	3
1.2 Limitations of the ASCII protocol.....	4
1.3 Programmers supported	4
1.4 PC control of programmer via RS-232	5
1.5 Remote System control of programmer via RS-232	5
2.0 Setting 'ASCII Text Mode' protocol	6
2.1 Overview	6
2.2 Is my programmer ENABLED for 'ASCII Text Mode' protocol ?	6
2.3 Upgrading your programmer for 'ASCII Text Mode' Protocol'	7
3.0 Standalone Programming Projects	8
3.1 Overview	8
3.2 Standalone Programming Projects definition	8
3.3 Creating a Standalone Programming Project.....	9
3.4 Testing a Programming Project in EDS (Development Mode).....	9
3.5 Choosing a unique Project Name	10
4.0 Uploading Projects to the Programmer	11
4.1 Overview	11
4.2 Uploading Projects using the EQTools - Upload Wizard utility	11
4.3 Uploading Projects using the Standalone Upload Wizard utility	11
4.4 Uploading Projects using the ConsoleEDS utility.....	13
5.0 Switching a programmer to ASCII Mode	14
5.1 Overview	14
5.2 Switching to the 'ASCII Text Communications' protocol	14
5.3 Switching back to the 'Binary Communications' protocol.....	15
6.0 Using the Terminal Emulation utility	17
6.1 Overview	17
6.2 Launching the EQTools – Terminal Emulator utility	17
6.3 Using the Terminal Emulator.....	17
6.4 Returning to EQTools – Binary Mode	18
7.0 ATC Protocol Specification	20
7.1 Communications Settings	20
7.2 ASCII Command Format.....	20
7.3 ASCII Command List	21
7.3.1 RESET command.....	21
7.3.2 PROG VERSION command	21
7.3.3 READ ALL command	21
7.3.4 PROGRAM command	22
7.3.5 FPROGRAM (FAST PROGRAM) command.....	22
7.3.6 INFO command	23
7.3.7 GO BINARY command.....	24
7.4 Error Codes / Messages	25
8.0 Typical control session	26
Appendix 1 – Document Change History	27

1.0 Introduction

The 'ASCII Text Control' communications protocol or 'ATC' is designed as a simple way for a Remote Test System to control the basic programming operations of an Equinox programmer via an RS-232 serial link. It is an 'ASCII' protocol with simple commands to initiate programming of pre-loaded 'Programming Projects' which are stored in the on-board 'FLASH Memory Store' of the programmer. The protocol supports up to 64 independent Programming Projects per programmer, but is limited to one programmer per PC COM (serial) port.

Implementation of this protocol is straightforward due to the limited number of commands and restricted functionality. As this is an ASCII protocol, it can be quickly evaluated using any Terminal Emulation Software Utility such as Hyperterminal. The speed of programming of a Target IC is very fast as the programmer is accessing all Project Data from on-board FLASH memory.

1.1 Features of the ASCII Protocol

This protocol has been designed as a simple method to allow a Remote Application such as an In-Circuit Tester (ICT) to control the basic operations of an Equinox ISP programmer. The programmer only supports the programming of entire 'Standalone Programming Projects' which are already uploaded to the programmer on-board FLASH memory.

Features:

- Simple ASCII protocol
- Supports control of any Equinox ISP programmer via a simple ASCII command set
- Supports execution of 'Standalone Programming Projects' via a single 'PROGRAM' command
- Programmer can be polled from the Remote Application to check for PASS / FAIL information
- Provides a diagnostic 'Error Code' if a project fails for any reason
- Straightforward to integrate into any Remote Application which can support RS232 Serial Communications
- Straightforward to integrate into any In-Circuit Tester (ICT) which can support RS232 Serial Communications
- Works with any PC Operating System
- Compatible with PCs running DOS (requires support for communications at 38,400 baud)
- The same control program will work with any Equinox ISP programmer as the ASCII command set is identical for all programmers

1.2 Limitations of the ASCII protocol

The limitations of the ASCII protocol are as follows:

- The protocol only supports programming of Programming Projects which are pre-loaded into the Programmer on-board FLASH Memory Store.
- The protocol does not support writing / reading of individual bytes or blocks of data to / from a Target Device.
- The protocol does not support uploading of 'Programming Projects' to the programmer. This must be performed using the EQTools – "Project Upload Wizard" or with the "ConsoleEDS /UPLOAD" command.
- Only very limited diagnostics are available using this protocol i.e. FAIL + Error Number.

1.3 Programmers supported

The Equinox programmers which are capable of supporting 'ASCII Text Mode' control are listed in the table below.

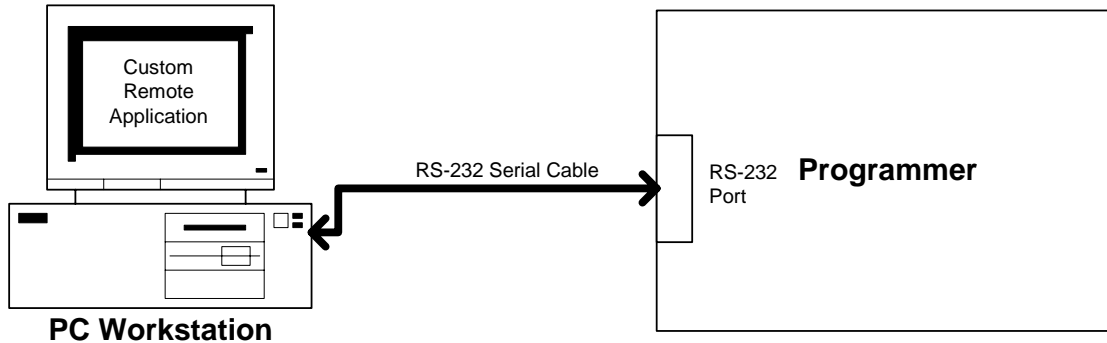
Programmer	ASCII Text Mode Control	Order Code
EPSILON5	License upgrade required	EPSILON5A1-UPG6
FS2000A	License upgrade required	FS2000A-UPG7
FS2003	License upgrade required	FS2003-UPG8
FS2009	License upgrade required	FS2009-UPG8
PPM3 MK2	Enabled as standard	-
PPM4 MK1	Enabled as standard	-
ISPnano	Enabled as standard	-

Please note:

- The PPM3 / PPM4 and ISPnano range of Production ISP Programmers are enabled for 'ASCII Text Mode' control as standard. The 'Communications Node Address' must be set to address '0'.
- A license upgrade is required for all other programmers to enable them for 'ASCII Text Mode'.

1.4 PC control of programmer via RS-232

An Equinox programmer can be remote controlled from a so-called 'Remote Application' running on a PC Workstation as detailed in the diagram below.



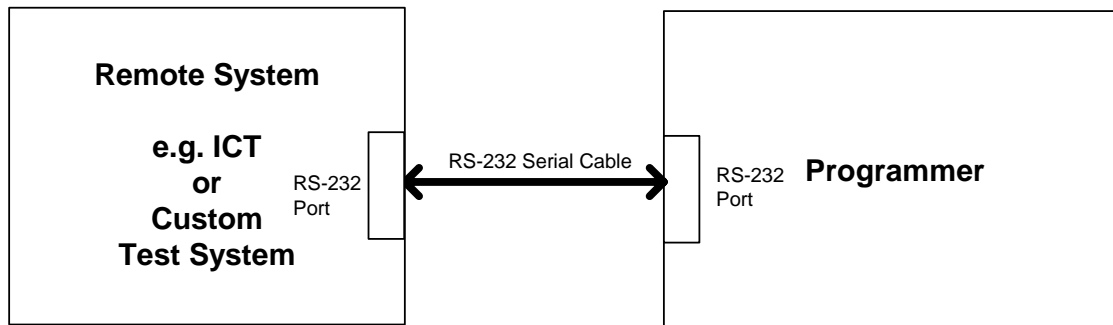
The 'Remote Application' is defined as custom software application written by the user which sends the required commands in the 'ASCII Text Mode' protocol to the programmer via the RS-232 link.

Please note:

- The protocol does not cater for uploading of the Programming Projects to the programmer in the first place. This task must be performed on a one-time-only basis using the 'EQTools – Project Upload Wizard'.
- A spare PC COM port is required to communicate with the programmer.

1.5 Remote System control of programmer via RS-232

An Equinox programmer can be remote controlled from any 'Remote System' such as an In-circuit Tester (ICT) or Custom Test System (CTS) via RS-232 using the 'ASCII Text Mode Protocol'.



The Remote System requires the following functionality:

- Spare RS-232 Serial Port
- Ability to generate and receive ASCII data at 38,400 BAUD
- Sufficient RAM to buffer communications messages

Please note:

The protocol does not cater for uploading of the Programming Projects to the programmer in the first place. This task must be performed on a one-time-only basis using the 'EQTools – Project Upload Wizard'.

2.0 Setting 'ASCII Text Mode' protocol

2.1 Overview

In order to use the '**ASCII Text Communications**' protocol to remote control an Equinox programmer, it is necessary to set up the programmer as detailed below:

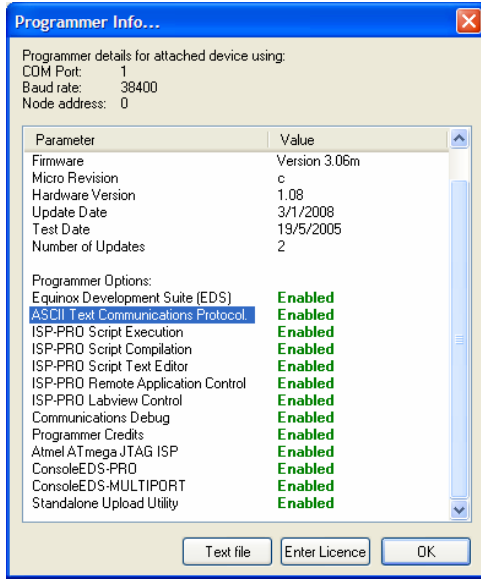
1. Verify that your programmer is ENABLED for '**ASCII Text Communications**' using EQTools
 - See section 2.2 for instructions
2. Upload your pre-compiled '**Standalone Programming Project(s)**' to the programmer using the EQTools – '**Upload Wizard**' utility.
 - The '**Standalone Project(s)**' must have already been compiled using EQTools.
 - These projects are then uploaded to FLASH memory on the programmer.
3. Switch on '**ASCII Text Communications Mode**' using EQTools for the Target Programmer
 - The programmer will now only respond to ASCII commands.
 - It will no longer respond to EQTools.
 - See section 5 for instructions.
4. Use '**Terminal Emulation Package**' or your '**Remote Application**' to communicate to the Target Programmer using the '**ASCII Text Mode**' protocol.
 - The '**Terminal Emulation Package**' is built in to EQTools and allows you to experiment with the '**ASCII Text Communications Mode**' without having to use any additional software.

2.2 Is my programmer ENABLED for 'ASCII Text Mode' protocol ?

The PPM3-MK2, PPM4 MK1 and ISPnano programmers are ENABLED for '**ASCII Text Communications**' protocol as standard. All other programmers require a chargeable license upgrade to enable this feature.

To check whether your programmer is ENABLED for '**ASCII Text Mode**' protocol, please follow the steps below:

- Connect the programmer to a PC spare COM port and apply power to the programmer.
- For the PPM3 / PPM4 / ISPnano programmers only, make sure that the '**Communication Node Address**' is set to address '0' and that '**RS-232 Mode**' is selected.
- Launch EQTools
- From the top menu bar, select **<Programmer><Programmer Info>**
- EQTools will now communicate with your programmer and display the '**Programmer Info**' screen:



- From the displayed list of options, check that '**ASCII Text Communications Protocol**' is ENABLED
- If '**ASCII Text Communications Protocol**' is DISABLED, then your programmer requires a 'License Upgrade' to allow this feature. Please consult the Equinox web site or contact Equinox via e-mail at support@equinox-tech.com to request the current price of this upgrade.

2.3 Upgrading your programmer for 'ASCII Text Mode' Protocol'

To upgrade your programmer to support 'ASCII Text Mode' Protocol', please follow the steps below:

1. Purchase the relevant upgrade from Equinox or your local distributor
2. If you have received an upgrade pack from a distributor, please email the serial number of your programmer and the License String on the enclosed form to support@equinox-tech.com. Equinox will then email you the relevant Upgrade String for your programmer.
3. To enter the License Upgrade Code
 - Launch EQTools
 - Select **<Programmer><Programmer Info>**
 - Select **<Enter License>**
 - Paste the 'License String' which has been emailed to you from Equinox (not the string printed on the sheet which came in the Upgrade Pack)
 - Click **<OK>**

→ The attached programmer should now be enabled for 'ASCII Text Communications Protocol'.

3.0 Standalone Programming Projects

3.1 Overview

The 'ATC' protocol supports execution of pre-compiled '**Standalone Programming Projects**' under the control of a Remote Application. This '**Standalone Project**' is a single file which contains all programming actions plus data including Programming Flags, FLASH CODE File, EEPROM DATA File, Fuse File settings and Security Fuse Settings. The Project File is generated using either the '**EQTools – Project Builder**' or the '**EQTools – EDS**' utilities.

For most programmers it is possible to have up to 64 '**Programming Projects**' resident in the programmer memory at any point in time. However, the lower cost Epsilon5 programmer only supports storing of a single project at any point in time.

The table below shows the number of '**Standalone Programming Projects**' which can be stored in each programmer type:

Programmer	Number of Standalone Programming Projects stored in programmer FLASH Memory
EPSILON5	1
FS2000A	64
FS2003	64
FS2009	64
PPM3 MK2	64
PPM4 MK1	64
ISPnano	64

Please note:

The '**Standalone Programming Projects**' must have already been uploaded to the programmer using the EQTools – '**Upload Wizard**' utility before the 'ATC' protocol can be used. Once the projects have been uploaded to the programmer, they can then be executed by either using the programmer keypad to select the relevant project or by sending the '**PROGRAM**' command to the programmer via the ATC protocol.

3.2 Standalone Programming Projects definition

A 'Standalone Programming Project' in a single project file (*.prj) which contains all the information required to program the Target IC.

The Programming Project contains the following information:

- Project Unique ID
- Target IC (Chip) to be programmed
- Target System Voltage (and current parameters – PPM3-MK2 only)

- Programming Flags
- FLASH CODE File (this is the firmware for the Target IC)
- EEPROM DATA File (this is the data file which is to be programmed into the EEPROM area of the Target IC)
- Fuse File settings
- Security Fuse Settings

This Project File (*.prj) file is then added to a '**Project Collection**' and finally uploaded to the programmer 'FLASH Memory Store'.

3.3 Creating a Standalone Programming Project

There are two ways of creating a 'Standalone Programming Project' as follows:

- Using EQTools – Project Builder
- Using EQTools – EDS (Development Mode) and then compiling the project using Project Builder

The instructions for creating a project are different depending on the Target IC which you are programming. For detailed instructions, please consult the relevant application note for the IC which you need to program.

When controlling a programmer using the 'ATC' protocol, the project must have been set up as a '**Standalone**' project. This basically means that when the project has finished executing, it will tri-state all I/O pins, power down the Target System (PPM3 MK2 only) and then wait for the next project to be selected.

To set up a 'Standalone Project':

- Launch EQTools
- Open your Programming Project using Project Builder or EDS (Development Mode)
- Select the <Programmer and Project type> tab
- Select 'Standalone – keypad control' from the drop-down menu
- Compile the project

3.4 Testing a Programming Project in EDS (Development Mode)

It is very difficult to test the correct execution of a 'Standalone Programming Project' once it has been uploaded to a programmer. It is therefore much better to perform any project testing using the 'EDS (Development Mode)' before uploading the project to the programmer.

To test a 'Standalone Programming Project' in EDS (Development Mode)

- Compile the Programming Project
- Add the project to a Project Collection → The Project Manager window is displayed
- Highlight the project you want to test in EDS mode
- Click the <Test in EDS> button → The selected project will now be launched into the EDS utility.
- It is now possible to manually test the programming of the FLASH, EEPROM, Fuses and Lock bits of a Target Device under PC control.

- Once you are happy that the project executes properly in EDS Mode, then you can test in Standalone Mode (uploaded to the programmer).

3.5 Choosing a unique Project Name

Each project must have a unique 'Project Name' or 'Project ID' which can be used to identify it via the 'ATC' protocol. It is recommended that 'version control' is enabled so that there is no chance of the incorrect project being executed by mistake.

The format of the Unique ID is as follows:

<Project_name> <Date><Version>

Example: **myproject1_150103_2.1.2.22**

Project Name

The Project Name is limited to 16 ASCII characters when version control is enabled.

Only alphanumeric characters should be used as far as possible.

The '_' (underscore) character is not allowed as this is used as a delimiter. The '-' (dash) character can be used instead.

Date

The date is usually the last compilation date of the project.

Version

The version information can be used for whatever version convention is required in your company. Many customers use the first three parts of the version as the current firmware version and the last part as the build number for the project.

For the example above:

- The project is called '**myproject1**'.
- It was compiled on the 15/01/03.
- The firmware version of the HEX file contained within the project is 2.1.2.
- The project is currently on build 22.

4.0 Uploading Projects to the Programmer

4.1 Overview

The '**ASCII Text Communications**' protocol does **not** support uploading of the actual '**Standalone Programming Projects**' contained within a 'Project Collection' to a target programmer. This means that all the required projects must be uploaded to the programmer using EQTools **BEFORE** the Remote Application starts to communicate with the programmer using the 'ASCII Text Communications' protocol.

This task must be performed using either the "**EQTools – Upload Wizard**" (free) or the "**ConsoleEDS UPLOAD**" command (chargeable upgrade).

4.2 Uploading Projects using the EQTools - Upload Wizard utility

The '**Upload Wizard**' utility provides a simple method of manually uploading the '**Standalone Programming Projects**' to a programmer. This utility is automatically installed as default during the EQTools software installation.

To upload your project(s) to the programmer from EQTools:

- Connect the programmer to a PC spare COM port and apply power to the programmer.
- Launch EQTools
- Make sure the programmer is configured for 'BINARY Communications' mode. (EQTools cannot communicate to the programmer if it is in ASCII mode)
- From the left-hand pane, click **<Project Manager>** and then select **<Open Collection>**
- Browse to and open the Project Collection which you wish to upload to the programmer.
- Click the **<Upload>** icon → Upload Wizard will launch
- Follow the on-screen instructions to upload your Project Collection to the programmer.

Please note:

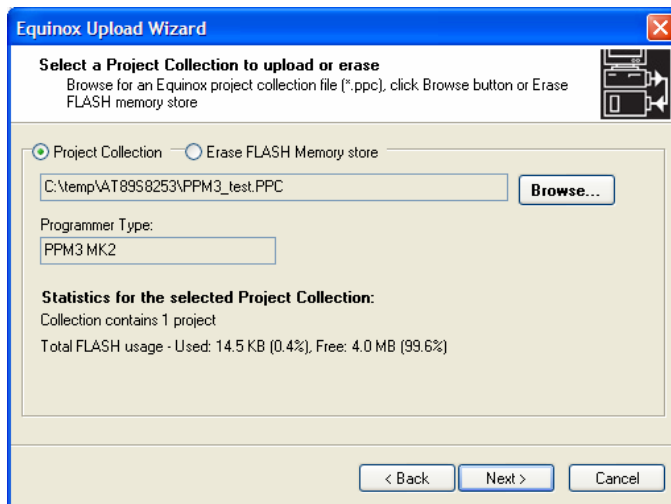
- It is also possible to install just the '**Upload Wizard**' utility without installing the main EQTools application.
- The '**Upload Wizard**' utility can be launched automatically from a Remote Application. This functionality requires a License Upgrade.

4.3 Uploading Projects using the Standalone Upload Wizard utility

The '**Upload Wizard**' utility can be installed as a standalone application i.e. without EQTools by running the EQTools installation and then opting for a 'Custom' install with only 'Upload Wizard' selected. This is useful for production applications where the minimal software is to be installed on the production PC. The 'Upload Wizard' utility provides a simple method of manually uploading the 'Standalone' Programming Projects' to a programmer without the complexities of EQTools.

To upload your project(s) to the programmer using the Standalone Upload Wizard utility:

- Connect the programmer to a PC spare COM port and apply power to the programmer.
- Launch Upload Wizard directly (The application can be found *in program files\equinox\UploadWizard.*)
- Make sure the programmer is configured for 'BINARY Communications' mode. (Upload Wizard cannot communicate to the programmer if it is in ASCII mode)
- Follow the on-screen instructions to detect the attached programmer and then click the **<Next>** button
- The 'Select a Project Collection' screen will now be displayed.
- Click the **<Browse>** button and then browse to and open the Project Collection (*.prj) file which you wish to upload to the programmer.



- Click the **<Next>** button
- Follow the on-screen instructions to upload your Project Collection to the programmer.
- At the end of the Upload Wizard, the application will exit.
- The Programming Projects are now uploaded in the programmer.
- The programmer is still in 'BINARY' communications mode.

4.4 Uploading Projects using the ConsoleEDS utility

The 'ConsoleEDS' utility can be used to upload 'Standalone Programming Projects' to an attached programmer. This utility can be launched and controlled from a Remote Application making it possible to completely automate the uploading of projects to a programmer. The use of ConsoleEDS requires the purchase of a ConsoleEDS license. Please see 'Application Note – AN111 – ConsoleEDS Manual' for full instructions.

The following simple commands can be used to upload multiple projects in a Project Collection to a programmer:

ConsoleEDS /UPLOAD=ProjectCollection.ppc

Where:

- ***ProjectCollection.ppc*** is the Project Collection file which contains the Programming Projects to be uploaded to the programmer.

ConsoleEDS will now start to upload the Programming Projects in the Project Collection. The Projects are uploaded and then a separate Verify Pass of each project is performed.

To upload a single project from a Project Collection to an attached programmer, use the following command:

ConsoleEDS /UPLOAD=ProjectCollection.ppc,ProjectName

Where:

- ***ProjectCollection.ppc*** is the Project Collection file which contains the Programming Projects to be uploaded to the programmer.
- ***ProjectName*** is the name (Unique ID) of the Programming Project to be uploaded.

ConsoleEDS will now start to upload the specified Programming Project in the Project Collection. Once the Project has been uploaded, a separate Verify Pass is performed.

5.0 Switching a programmer to ASCII Mode

5.1 Overview

Any Equinox programmer is configured to be in 'Binary Communications Mode' as standard. This allows it to communicate with EQTools. Once you have uploaded your projects to the programmer using EQTools, it is then necessary to configure the programmer to start communicating using the ASCII protocol. This will allow you to send ASCII commands to the programmer from any Remote Application capable of sending serial commands via ASCII.

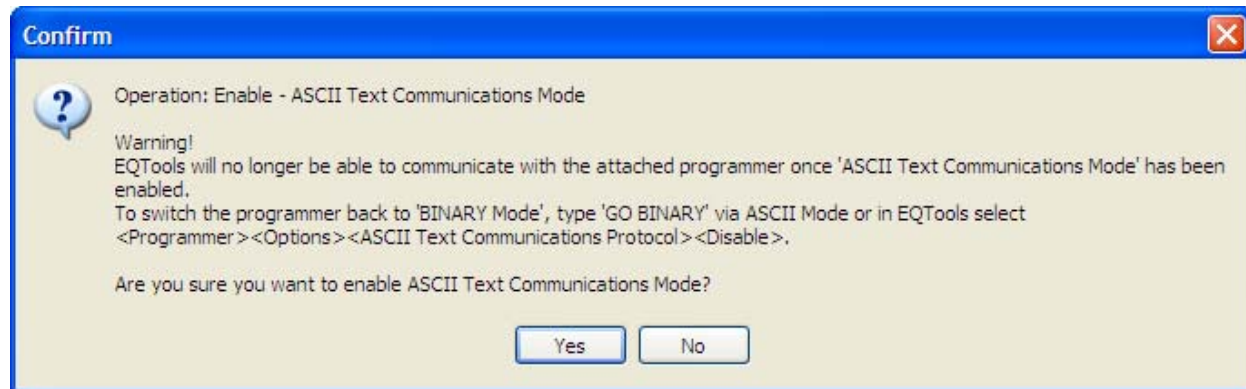
Please note:

Once the programmer is operating in 'ASCII Mode', EQTools will no longer be able to communicate with the programmer.

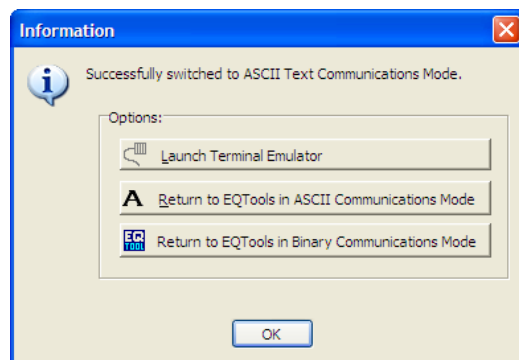
5.2 Switching to the 'ASCII Text Communications' protocol

To switch to the 'ASCII Text Communications' protocol:

- Launch EQTools
- From the top menu bar, select **<Programmer><Options><Text Communications>**
- Select **<Enabled>** → EQTools will confirm that the programmer is now running in ASCII protocol mode.



- Click **<Yes>** → the following menu is now displayed....



- Click **<Launch Terminal Emulator>** to test the programming using the 'Terminal Emulator' built in to EQTools.
- If you want to use your own third party 'Terminal Emulator', click the **<Return to EQTools in ASCII Communications Mode>** button and then close down EQTools to release the COM port for use by the Remote Application (if required).

Please note:

Once the programmer is enabled for the ATC protocol, the programmer will no longer communicate with the main EQTools application. You can now only communicate with the attached programmer using the 'Terminal Emulator' which comes as part of EQTools or using your own Remote Application or Terminal Emulator package eg. Hyperterminal. To get the programmer to work with EQTools again, it is necessary to switch back the EQTools 'Binary' protocol.

5.3 Switching back to the 'Binary Communications' protocol

EQTools uses a 'Binary Communications' protocol to communicate with the Target Programmer. Once the programmer is enabled for the ATC protocol, the programmer will no longer communicate with EQTools. To get the programmer to work with EQTools again, it is necessary to switch back to the EQTools 'Binary' protocol.

To switch back to the EQTools - 'Binary Communications' protocol, please use one of methods detailed below:

1. From EQTools - Terminal Emulator utility:

- From within EQTools, select **<Programmer><Terminal Emulator>**
- Click the **<Go Binary>** button or type **GO BINARY<CR>** in the window
 - Programmer switches back to the 'Binary Communications' protocol.
 - EQTools will now be able to communicate with the programmer as normal.

2. From EQTools – Options menu

- From the top menu bar, select **<Programmer><Options><Text Communications>**
- Select **<Disabled>** → EQTools will send **'GO BINARY<CR>'** to the target programmer.
- Programmer switches back to the 'Binary Communications' protocol.

3. From your own Terminal Emulation Package:

- Launch the Terminal Emulator package eg. Hyperterminal
- Select the correct COM port to which the programmer is attached
- Set the communications parameters to Baud rate: 38,4000, 8 data bits, No parity
- In the emulator message window, type **'GO BINARY<CR>'**
- Programmer switches back to the 'Binary Communications' protocol.

4. From your Remote Application

- Select the correct COM port to which the programmer is attached
- Set the communications parameters to Baud rate: 38,4000, 8 data bits, No parity
- Send the **'GO BINARY<CR>'** command

- Programmer switches back to the 'Binary Communications' protocol.
- The programmer will no longer respond to the Remote Application via the 'ACT' protocol.

6.0 Using the Terminal Emulation utility

6.1 Overview

EQTools features a simple 'Terminal Emulation' utility which can be used to test out any programmer which is enabled for 'ASCII Text Communications (ATC) Protocol'. This utility is already set up for the correct communication parameters to communicate with the programmer and is therefore probably easier than using a third party Terminal Emulator. It is also very simple to swap between 'ASCII' and 'Binary' communications mode directly from the Terminal Emulator saving valuable time during testing.

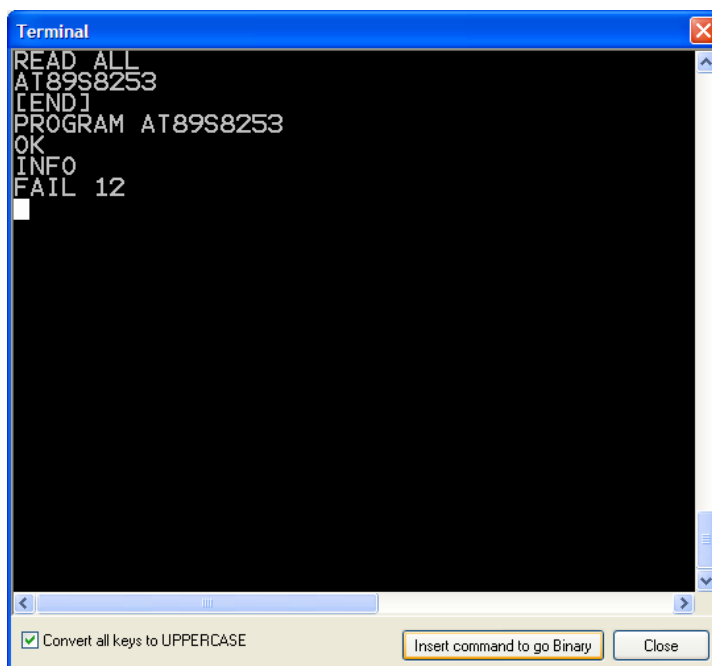
6.2 Launching the EQTools – Terminal Emulator utility

The EQTools – Terminal Emulator can be launched as follows:

1. Enable the attached programmer for 'ASCII Text Communications (ATC) Protocol'.
- see section 5.
or
2. From within EQTools, select **<Programmer> <Terminal Emulator>**
→ Terminal Emulator utility will now launch
→ The utility automatically defaults to the correct baud rate etc.

6.3 Using the Terminal Emulator

When the Terminal Emulator is launched, a new window called 'Terminal' will be visible. It is now possible to type ASCII commands directly into this window. To send a command to the programmer, simply press the **<CR>** (Carriage Return) key.



In the above example, the programmer already contains a single project called 'AT89S8253'.

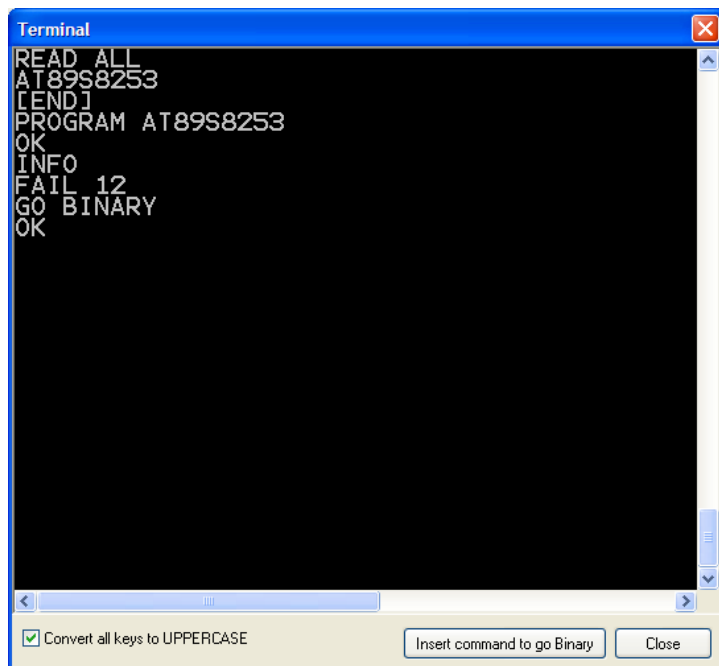
The table below details the ASCII commands sent to the programmer and the responses back from the programmer.

PC command sent	Response from programmer	Comment
READ ALL	AT89S8253 [END]	This command returns a list of the 'Programming Projects' in the programmer.
PROGRAM AT89S8253	OK	This command executes the specified 'Programming Project'. The programmer will send the response 'OK' immediately after receiving the command. The 'INFO' command should then be sent to poll the programmer to see if the project has finished execution.
INFO	PASS or FAIL xx	The 'INFO' command is then used to poll the programmer to see if the 'PROGRAM' command has completed.

6.4 Returning to EQTools – Binary Mode

Once you have finished experimenting with the programmer in 'ASCII Mode' you can return to EQTools programmer control mode by performing the following actions:

- Click the **<Insert the command to Go Binary>** button or type out the command '**GO BINARY**' manually
- Hit the **<CR>** (Carriage Return) button on the PC keyboard



- Click the **<Close>** button to return to the main EQTools window
- The programmer is now operating in 'BINARY' Communications Mode
- EQTools now has full control of the programmer

7.0 ATC Protocol Specification

7.1 Communications Settings

When operating using the 'ACT' protocol, the programmer uses the communication settings as detailed in the table below.

Parameter	Setting
Baud Rate	38,400 (fixed)
Data Bits	8
Stop Bits	1
Parity	None
Hardware Handshaking	None
ASCII Settings	<ul style="list-style-type: none"> - Send line end with line feeds - Echo typed characters locally - Append line feeds to incoming line ends

If you are using a Terminal Emulation Utility such as Hyperterminal, it is necessary to set up the 'ASCII Settings' as detailed in the table so that the types text is echoed to the screen and line feeds are processed correctly.

7.2 ASCII Command Format

This section details the ASCII commands which the programmer recognises when operating with the 'ACT' protocol. It also details the possible responses which the programmer will give to each command.

Please note:

- All commands must be in CAPITAL letters.
- A <CR> (carriage return character) must be sent after each command.
- The use of '<...>' is for illustration use in the manual only. These characters should not be sent to the target programmer.
- The programmer will recognise commands with leading spaces.
- The programmer will send the message 'BAD COMMAND' if it does not recognise a command sent from the PC.

7.3 ASCII Command List

This section details all the commands available in ASCII mode.

7.3.1 RESET command

The 'RESET' command initiates a 'Soft Reset' of the programmer. If the command is executed during a programming operation, the programmer will quit out of the current operation and reset to the state as if it had just been powered on.

PC command sent	Response from programmer	Comment
RESET	OK	The programmer will quit out of any programming operation and reboot.

7.3.2 PROG VERSION command

The '**PROG VERSION**' command returns the programmer firmware version, serial number, hardware version and last firmware update date.

PC command sent	Response from programmer	Comment
PROG VERSION	CODE VERSION 3.06 SERIAL NUMBER 840 HARDWARE VERSION 1.08 UPDATE DATE 3-1-2008 [END]	The list is terminated by the text '[END]'.

7.3.3 READ ALL command

The '**READ ALL**' command returns a full list of all Programming Projects which are currently resident in the programmer 'FLASH Memory Store'. The project 'Unique ID' is returned which features the 'Project name' plus version control characters.

PC command sent	Response from programmer	Comment
READ ALL	ProjectName1 ProjectName2 ProjectNamen [END]	This command returns a list of the 'Programming Projects' in the programmer. The list is terminated by the text '[END]'.

7.3.4 PROGRAM command

The '**PROGRAM**' command starts the execution of a specified 'Standalone Programming Project'.

PC command sent	Response from programmer	Comment
PROGRAM PROJECTNAME	<p>If the specified project is found in the programmer, it responds with: OK</p> <p>If the specified project is not found in the programmer, it responds with: BAD IMAGE</p>	<ul style="list-style-type: none"> • A 'Project Image CRC Check' is performed • The programmer will then send the response 'OK' immediately after receiving the command and the 'BUSY' LED will illuminate on the programmer. • The 'INFO' command should then be sent to poll the programmer to see if the project has finished execution.

Where:

- **PROJECTNAME** is the name of the Programming Project to be executed

7.3.5 FPROGRAM (FAST PROGRAM) command

The '**FPROGRAM**' command is identical in functionality to the '**PROGRAM**' command except that it does not perform a 'Project Image CRC Check' before starting executing of the specified 'Standalone Programming Project'. This can speed up the execution of a project significantly as the 'Image Check' can be time consuming for large FLASH / EEPROM area devices.

PC command sent	Response from programmer	Comment
FPROGRAM PROJECTNAME	<p>If the specified project is found in the programmer, it responds with: OK</p> <p>If the specified project is not found in the programmer, it responds with: BAD IMAGE</p>	<ul style="list-style-type: none"> • A 'Project Image CRC Check' is not performed • The programmer will send the response 'OK' immediately after receiving the 'FPROGRAM' command and the 'BUSY' LED will illuminate on the programmer. • The 'INFO' command should then be sent to poll the programmer to see if the project has finished execution.

Where:

- **PROJECTNAME** is the name of the Programming Project to be executed

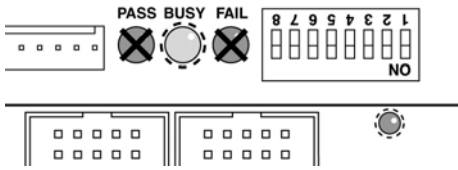
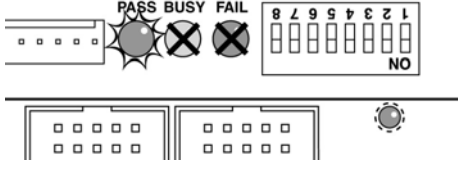
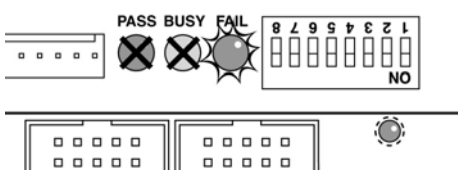
Please note:

This command requires that the programmer is running firmware 3.07 or above.

7.3.6 INFO command

The **'INFO'** command polls the programmer for the status of the project which is currently executing on the programmer. The programmer will return **'BUSY'** during normal execution of the project and the **'BUSY'** LED will illuminate constantly on the programmer. If the project finishes with no programming errors, then the programmer will return **'PASS'** and the **'PASS'** LED will flash constantly. If programming error occurs, then the programmer will return **'FAIL xx'** where 'xx' is the associated error code and the **'FAIL'** LED will flash constantly.

Here is a summary of the possible responses:

PC command sent	Response from programmer	Programmer Status LEDs
INFO	i. BUSY <ul style="list-style-type: none"> If the programmer is still executing the project. The 'BUSY' LED will illuminate constantly. 	
	ii. PASS <ul style="list-style-type: none"> If the programmer has executed the project and the result was a PASS. The 'PASS' LED will flash constantly. 	
	iii. FAIL xx <ul style="list-style-type: none"> If the programmer has executed the project and the result was a failure condition. The 'xx' is the error code for the fail condition. The 'FAIL' LED will flash constantly. 	

Please note:

- The programmer can be polled using the **INFO** command at any time during the execution of the project.
- When the project has finished executing, the **'PASS'** or **'FAIL'** LED will flash continuously. This status condition is automatically cleared when the next project is executed.
- The **'Error Codes'** can be looked up in the **'Error Messages Manual'**.

7.3.7 GO BINARY command

The '**GO BINARY**' command forces the programmer back to 'Binary Communications Mode' which is compatible with the Equinox EQTools and ISP-PRO software.

PC command sent	Response from programmer	Comment
GO BINARY	OK	<ul style="list-style-type: none">The programmer will no-longer respond to ASCII commands after this command has been executed.

Please note:

- The programmer will no-longer respond to ASCII commands after this command has been executed.
- This command should be executed in order for EQTools to communicate with the programmer again.

7.4 Error Codes / Messages

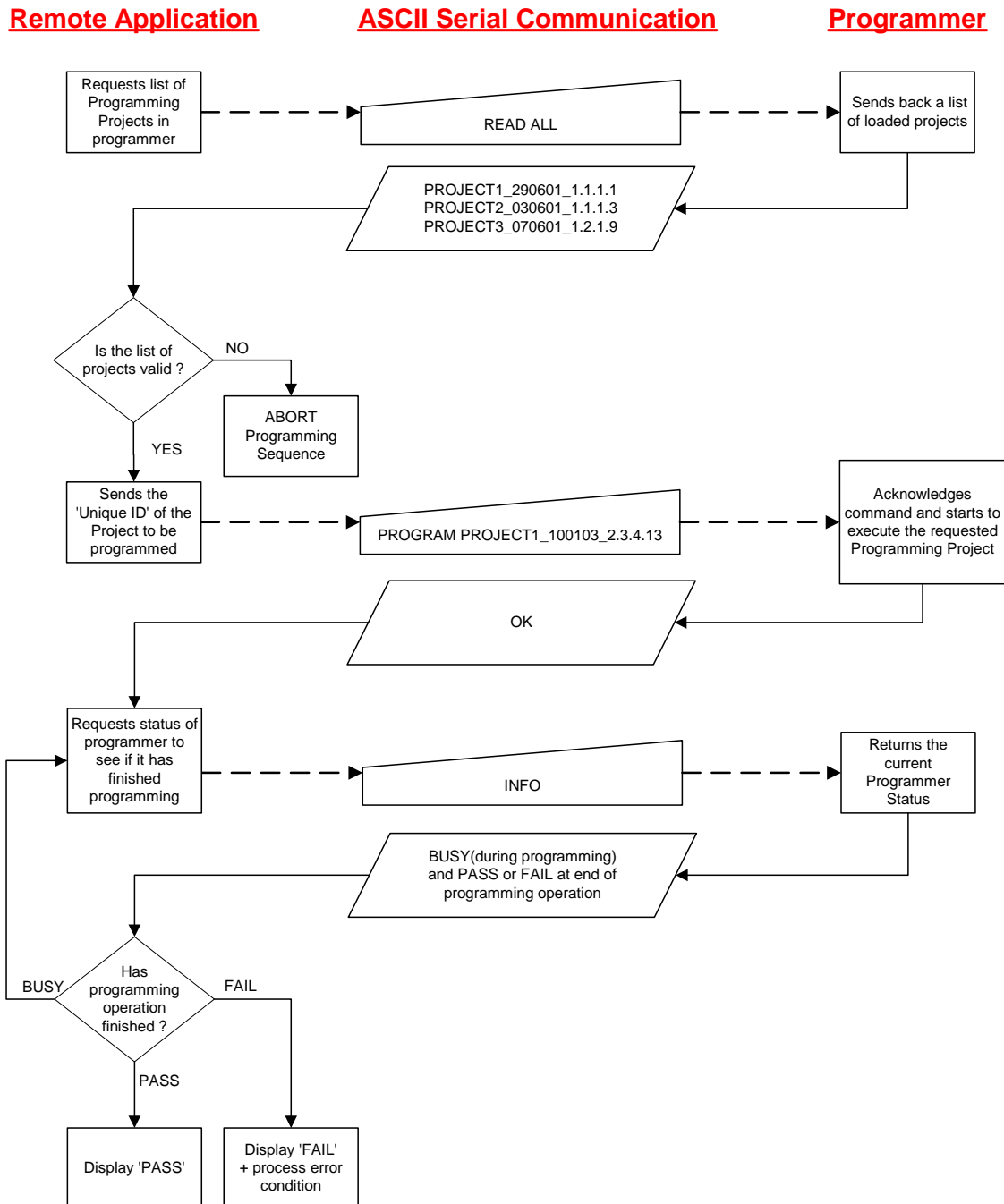
When the **PROGRAM**, **FPROGRAM** or **INFO** commands are executed, it is possible that the programmer will return '**FAIL**' plus an '**Error Code**'. This '**Error Code**' usually represents the failure mode and can be used to diagnose what went wrong. A full explanation of each 'Error Code' can be found in the '**Programmer Error Messages**' application note which is available for download from the Equinox web site.

It is very difficult to diagnose why a '**Standalone Programming Project**' has failed as there is very little diagnostic information available. If the programmer has an LCD display, then there may be further diagnostics available on this display.

The best way to diagnose programming failures is to launch the '**Standalone Programming Project**' in the EDS (Development Mode) within EQTools. This allows you to fully experiment and test the project under PC control. Any programming failures can usually be found by using this EDS mode.

8.0 Typical control session

The diagram below details a typical example of a control flow for a 'Remote Application' controlling an Equinox programmer. The 'Remote Application' sends serial commands (see 'ASCII Serial Communication' column) to the 'Programmer' which in turn responds with the relevant strings.



Appendix 1 – Document Change History

Version 1.06 – 27th March 2009

- Added support for the PPM4-MK1 programmer
- Added support for the ISPnano programmer

Version 1.04 – 29th January 2008

- Added new '**FPROGRAM**' command.
- Added '**Standalone Project**' instructions
- Updated '**Uploading Projects to programmer**' instructions
- Added '**Terminal Emulator**' instructions
- Added '**Error Messages / Codes**' section