

Report No:

AN123

Title:

Controlling an Equinox ISP Programmer from a Remote System via the programmer 4-wire TTL Remote Control Port

Author:

John Marriott

Date:

8th April 2009

Version Number:

1.04

Abstract:

This application note describes how to control an Equinox ISP programmer from a Remote System using the 4-wire TTL 'Remote Control' port. This allows the programmer to be controlled by any Remote System using 4 signal lines.

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 Overview.....	3
1.1 Introducing Remote TTL I/O Control	3
1.2 Features	4
1.3 Advantages and Disadvantages	4
1.4 Programmers Supported.....	5
2.0 Remote TTL I/O Control signalling sequence.....	6
2.1 Overview of control sequence.....	6
2.2 Remote Control Sequence diagram.....	7
2.3 Remote Control sequence explanation	8
3.0 Remote TTL Control Interface Connector	9
3.1 Overview	9
3.2 PPM3-MK2 and PPM4-MK1 – Remote Control Port.....	9
3.2.1 TTL Control Port on the IO-CON-1 or IO-CON-3 modules.....	10
3.2.2 TTL Control Port on the IO-CON-2 module	11
3.3 ISPnano programmer – Remote Control Port	12
3.3.1 Overview.....	12
3.3.2 ISPnano - Remote Control Port – pin-out.....	12
4.0 Standalone Programming Projects.....	13
4.1 Overview	13
4.2 Creating a Standalone Programming Project.....	13
4.3 Forcing the Project to auto-start.....	14
4.4 Uploading a single project to a programmer	15
4.5 Testing a single project on the programmer.....	15
4.6 Uploading multiple projects to the programmer	16
4.6.1 Overview.....	16
4.6.2 Instructions	16
4.6.3 Unlocking the programmer keypad.....	18
4.6.4 Selecting a new project via the keypad (unlocked mode)	19

1.0 Overview

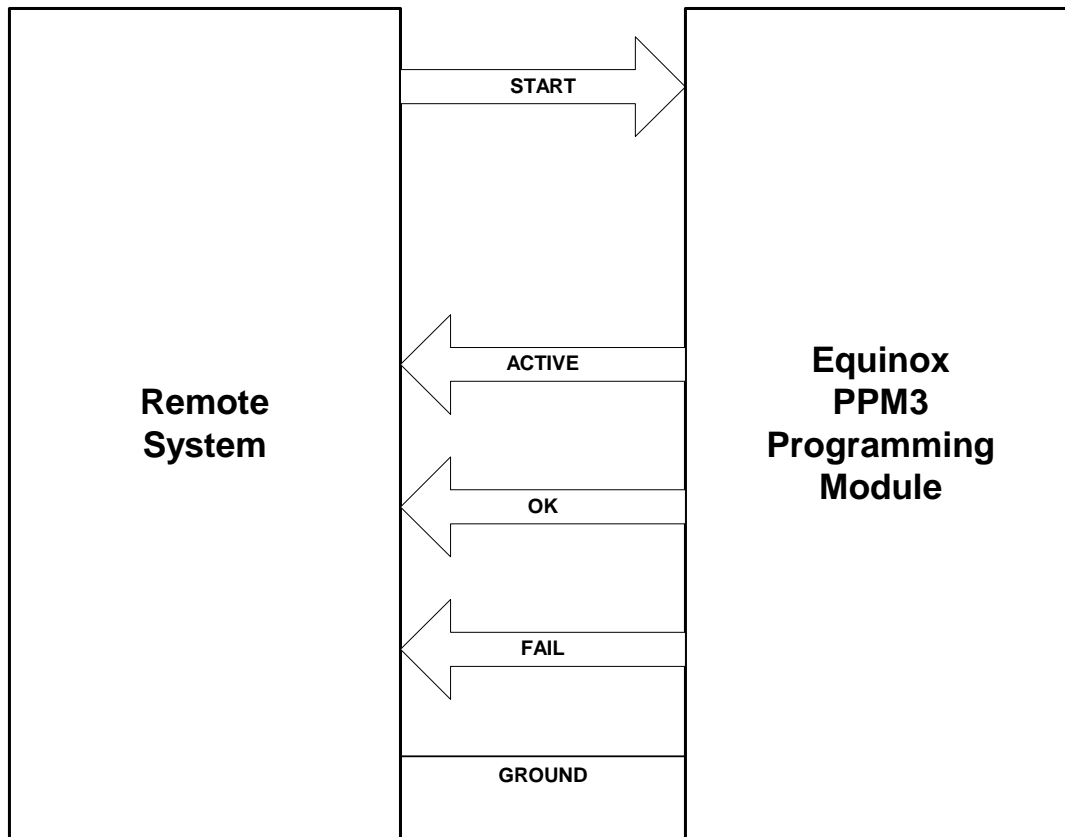
1.1 Introducing Remote TTL I/O Control

This application note describes how to control an Equinox ISP programmer operating in '**Standalone Mode**' from a '**Remote System**' using the 4-wire '**Remote System - TTL Control Port**' on the programmer. This control method is ideal for interfacing the programmer to **In-Circuit Testers (ICT's)** or other production equipment which does not have an RS-232 port.

The '**Remote System**' controls the programmer using 4-TTL signal lines:

- Start, Active, OK and FAIL – see diagram below.

Fig. 1.0 Remote System Control of Programmer



The definitions of the '**Remote System**' and the '**Equinox Programmer**' are as follows:

Remote System

The '**Remote System**' is the remote hardware / control mechanism which is used to control the programmer. This could be another microcontroller based system, a PLC or any other system which has TTL I/O capability.

Equinox Programmer

This is the Equinox Programmer which is to be controlled by the '**Remote System**'.

1.2 Features

The '**Remote TTL I/O Control**' method is a very simple and yet powerful way of controlling an Equinox programmer.

The main features of this control method are as follows:

- Allows an Equinox ISP Programmer to be controlled using any '**Remote System**'.
- Programmer control uses only 4 x TTL signal lines
- Ideal for interfacing a programmer to an In-Circuit Tester (ICT) which only has TTL control outputs available.
- Ideal for interfacing a programmer to a PLC (Programmable Logic Controller)
- Ideal for interfacing a programmer to any Remote System which does not have a serial port
- Remote System only needs to have the ability to assert / monitor 4 x TTL I/O lines in order to control the programmer.
- Supports automatically starting a single pre-loaded '**Standalone Programming Project**'
- Progress of the programmer can be monitored using the ACTIVE signal line
- The programmer reports a PASS or FAIL result via the OK and FAIL signal lines

1.3 Advantages and Disadvantages

The relative advantages and disadvantages of the '**Remote TTL I/O Control**' method are as follows:

Advantages:

- Very simple method to control the operation of an Equinox Programmer.
- It is compatible with any Remote System which can sequence 4 x TTL I/O lines.
- Very simple to implement – no specialist programming knowledge required.

Disadvantages

- Only supports control of a single pre-loaded '**Standalone Programming Project**'
- It is **NOT** possible to control multiple '**Standalone Programming Projects**' as there is no mechanism to select the correct project via TTL port.
- It does **NOT** allow control of individual programming actions e.g. Erase, Program FLASH, Program Fuses etc.
- No diagnostic information available if the programming operation fails for any reason.

1.4 Programmers Supported

The following programmers support '**Remote TTL I/O Control**':

Programmer	Programmer Description	Remote TTL I/O Control
PPM3 MK2	Single channel Production ISP Programmer	Enabled as standard
PPM4 MK1	Single channel Production ISP Programmer	Enabled as standard
ISPnano	Single channel Production ISP Programmer	Enabled as standard

Please note:

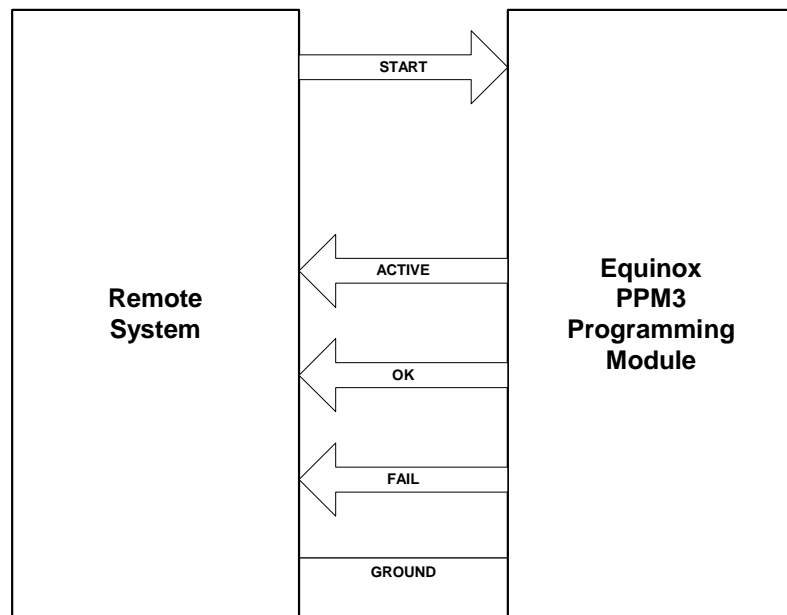
The '**Remote TTL I/O Control**' is enabled as standard. An upgrade is not required to use this functionality.

2.0 Remote TTL I/O Control signalling sequence

2.1 Overview of control sequence

The '**Remote System**' controls the programmer using 4-TTL signal lines:

- Start, Active, OK and FAIL – see diagram below.

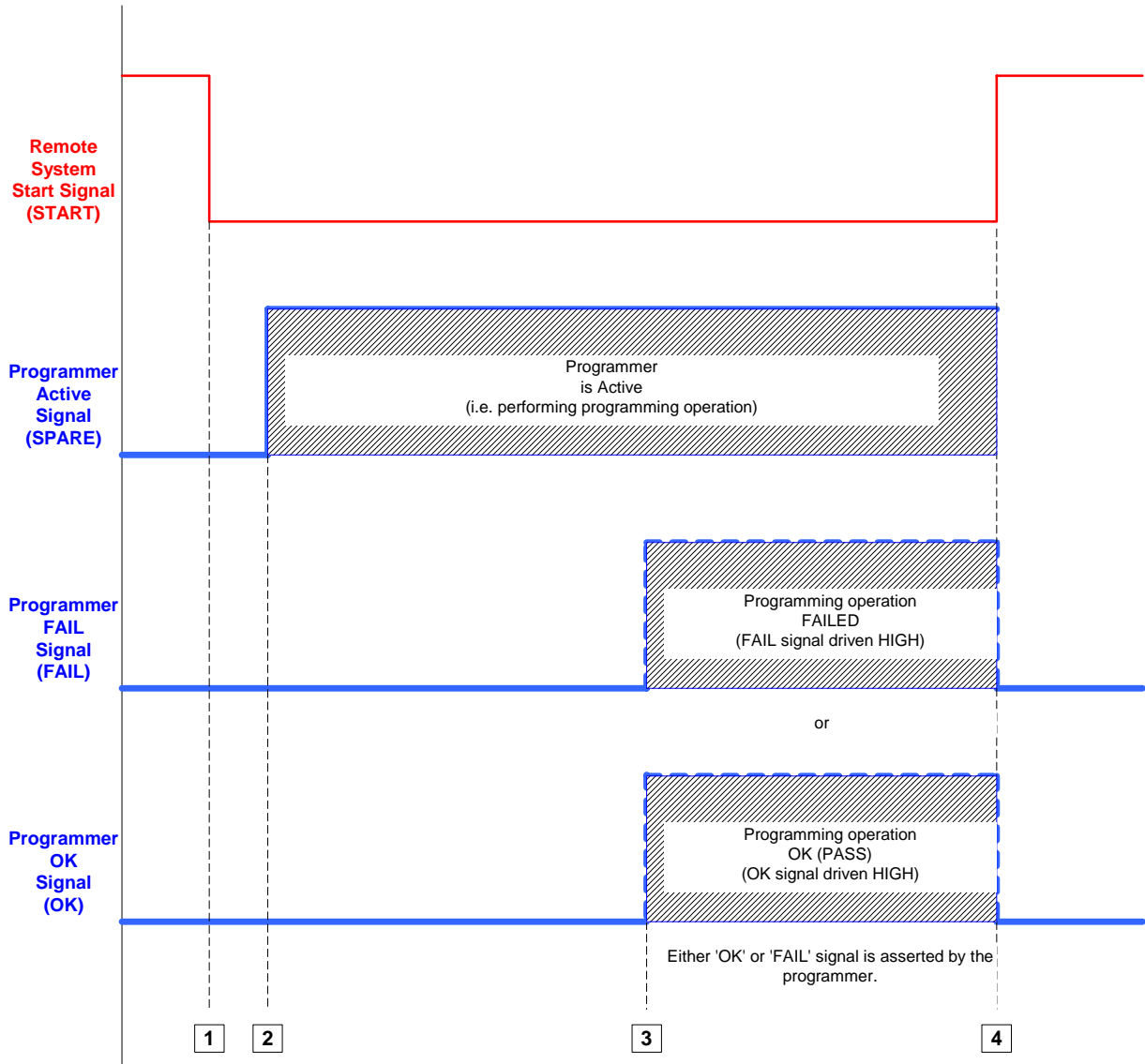


The four control signals are described in the table below:

#	Title	I/O direction	Signal description
1	START	Input to programmer	Remote Start Signal The Remote System drives this pin LOW to start a programming operation.
2	ACTIVE	Output from programmer	Indicates that the programmer is 'ACTIVE' executing a Programming Project
3	FAIL	Output from programmer	Driven HIGH by the programmer if the Programming Project has failed.
4	OK	Output from programmer	Driven HIGH by the programmer if the Programming Project has passed.

2.2 Remote Control Sequence diagram

The '**Remote System**' is always the '**MASTER**' and controls the '**Equinox Programmer**' which is always the '**SLAVE**'. The sequence of events is detailed in the diagram below.



Key	Programmer / Remote System Action
1	The Remote System asserts the programmer <START> signal to initiate the execution of a 'Programming Project'.
2	The programmer will then assert the <ACTIVE> signal to indicate that it has commenced programming.
3	At the end of the Programming operation the programmer asserts either the <OK> or <FAIL> signal depending on the outcome.
4	The Remote System must then de-assert the <Start> signal to allow the programmer to reset ready to program the next device.

2.3 Remote Control sequence explanation

The '**Remote System**' is always the '**MASTER**' and controls the '**Equinox Programmer**' which is always the '**SLAVE**'. The '**Remote System**' starts a programming sequence by taking the **<START>** signal **LOW** and should then wait for the programmer to assert the **<ACTIVE>** signal **HIGH** to indicate that programming has started. The '**Remote System**' should then monitor the **<OK>** and **<FAIL>** signal lines to check whether the programming operation has passed or failed.

The typical control sequence is as follows:

1. The programmer is in the '**Waiting for Start Signal**' state.
 - It is waiting for the '**Remote System**' to assert the **<START>** signal **LOW** to initiate the execution of a '**Standalone Programming Project**'.
 - The yellow '**BUSY**' status LED will come on permanently to indicate that the programmer is in the '**Waiting for Start Signal**' state.
 - On the PPM3-MK2 and PPM4-MK1 programmers, the Programmer LCD will display:
"WAITING for START SIGNAL"
 - All programmer I/O pins are tri-stated
 - The '**Programmer Controlled Power Supply**' is off (no power on the Target System).
2. The '**Remote System**' asserts the programmer **<START>** signal **LOW**
→ This action will start the execution of the selected '**Standalone Programming Project**'.
3. The programmer will then assert the **<ACTIVE>** signal **HIGH**
 - This indicates that the programmer has commenced programming.
4. Once the '**Standalone Programming Project**' has finished executing, the programmer asserts either the **<OK>** or **<FAIL>**
 - The **<OK>** signal is asserted and the GREEN '**PASS**' LED will flash if the Programming Project executed without error.
 - The **<FAIL>** signal is asserted and the RED '**FAIL**' LED will flash if the Programming Project produces an error for any reason.
5. The '**Remote System**' must then de-assert the **<Start>** signal (drive it **HIGH**)
 - This signals to the programmer to clear the PASS / FAIL state.
6. The programmer resets back to the '**Waiting for Start Signal**' state.
7. The sequence is now repeated from step (1).

3.0 Remote TTL Control Interface Connector

3.1 Overview

The Remote System connects to the programmer via the ***'Remote System TTL Control Port'***. This section describes how to connect to the correct pins on the PPM3-MK2, PPM4-MK1 and ISPnano programmer. On a PPM3-MK2 or PPM4-MK1 programmer, the ***'Remote Control'*** pins can be found on the relevant ***'I/O Connector Module'***. On the ISPnano programmer, the ***'Remote Control'*** pins can be found on a dedicated ***'Remote Control Port'***.

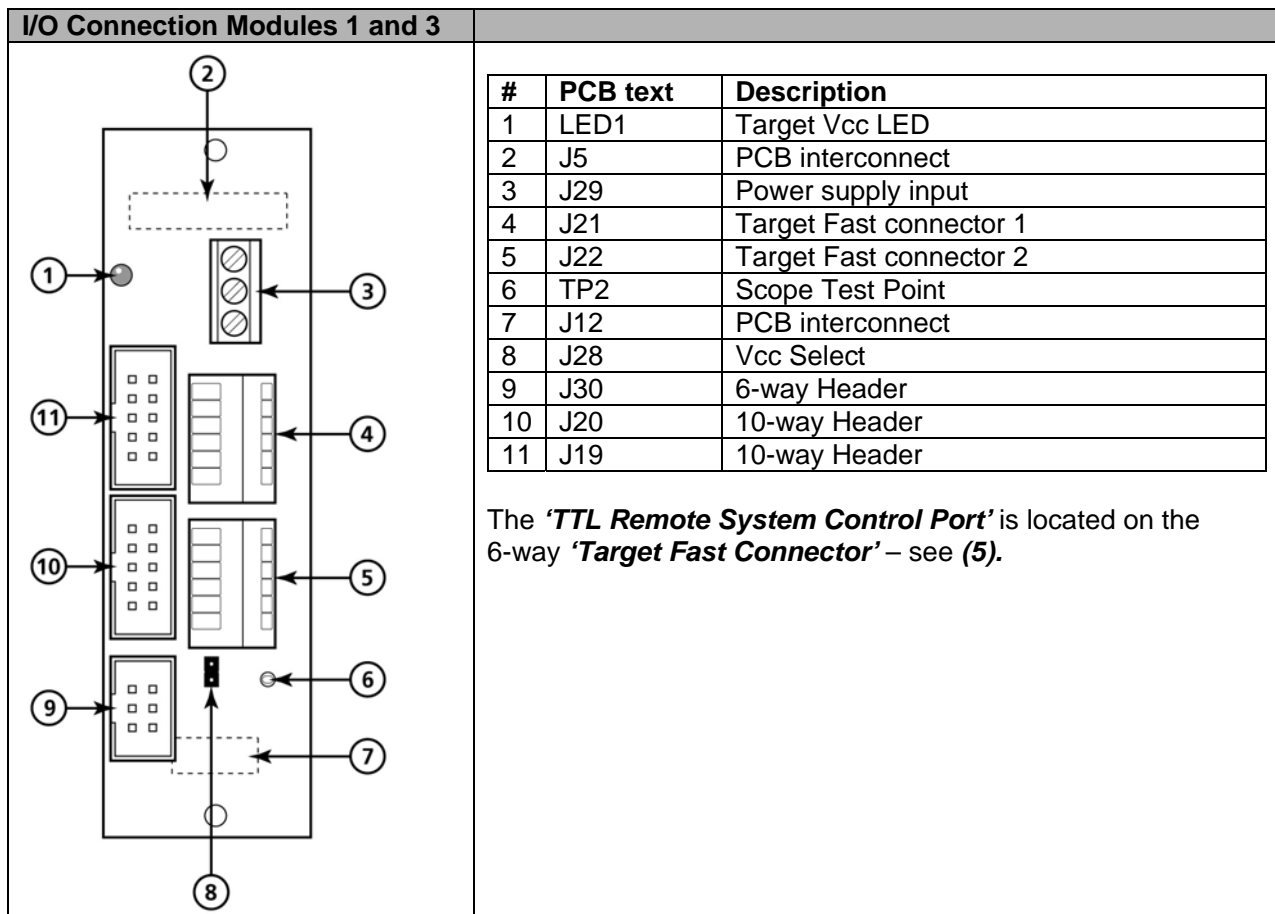
3.2 PPM3-MK2 and PPM4-MK1 – Remote Control Port

The signals required to remote control a PPM3-MK2 or PPM4-MK1 programmer can be found on the relevant ***'I/O Connector Module'***. Please refer to the table below for the position of the ***'Remote System TTL Control Port'*** on each I/O Connector Module.

Section	I/O Connector Module	Remote Control Port
3.2.1	IO-CON-1 Module	Target Fast Connector 1 (J22)
3.2.2	IO-CON-2 Module	Wire-wrap Connector (J25)
3.2.1	IO-CON-3 Module	Target Fast Connector 1 (J4)

3.2.1 TTL Control Port on the IO-CON-1 or IO-CON-3 modules

The '**TTL Remote System Control Port**' on the IO-CON-1 and IO-CON-3 I/O connector modules is located on the 6-way '**Target Fast Connector**' – see (5) on the diagram below. The **START**, **FAIL** and **OK** signals are available on this connector. Unfortunately, the '**ACTIVE**' signal is missing from this connector so this must be manually soldered to one of the red connectors.



The pin-out of the '**Target Fast Connector**' is shown in the table below:

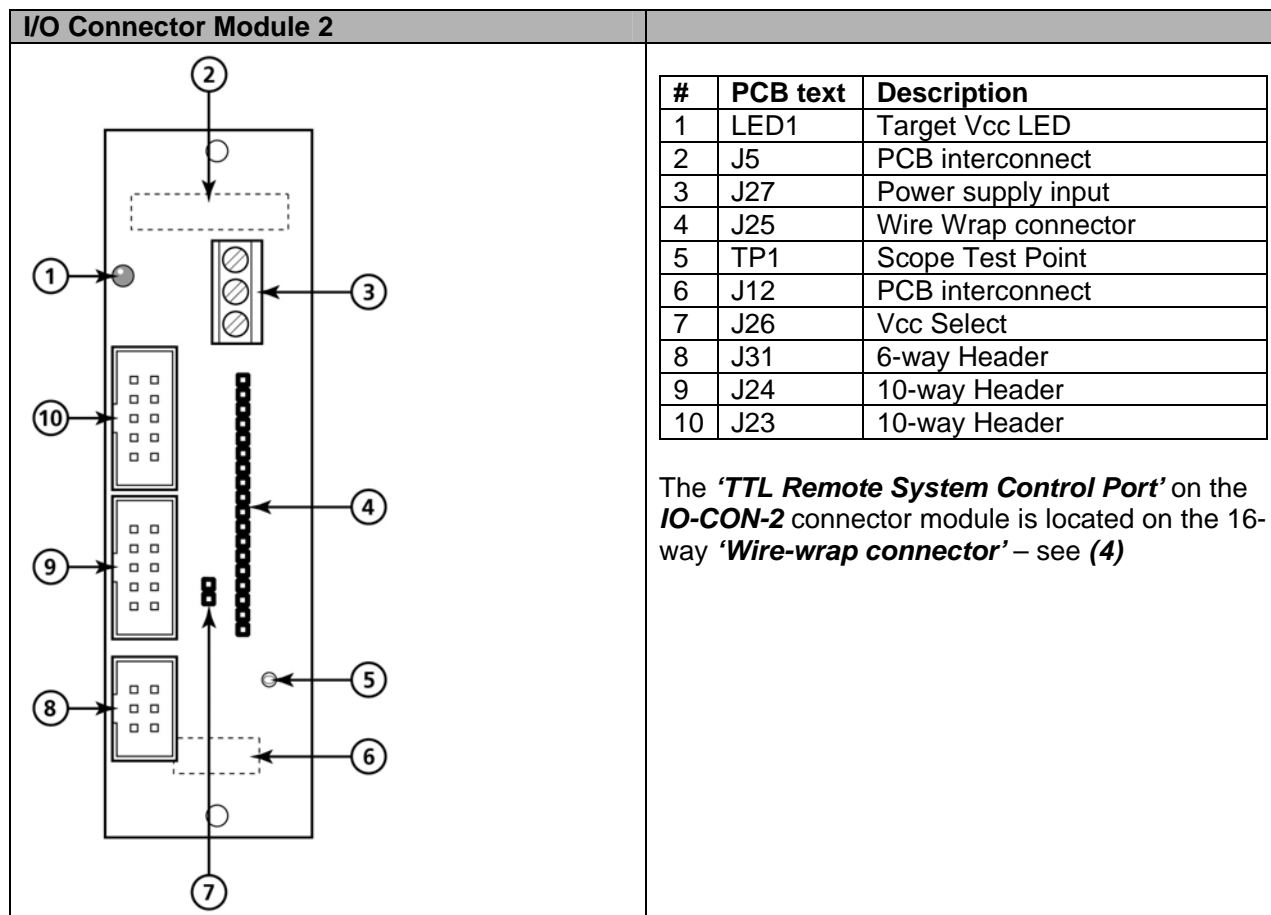
Pin No.	Title	I/O	Description
1	START(+)	I	Remote <Start> Input Signal (3V –12 V DC)
2	ANA1	I	Spare Input Line (referenced to Target Vcc)
3	FAIL	O	FAIL (output)
4	OK	O	PASS (output)
5	TARGET_VCC	P	Target Vcc Voltage
6	TARGET_GND	P	Target GROUND

Please note:

Each pin is also labelled on the circuit board next to the relevant pin.

3.2.2 TTL Control Port on the IO-CON-2 module

The '**TTL Remote System Control Port**' on the **IO-CON-2** connector module is located on the 16-way '**Pin-header strip connector**' – see (4) on the diagram below. The **START**, **FAIL**, **ACTIVE** and **OK** signals are available on this connector.



The relevant pins for '**TTL Remote Control**' on the '**Wire-wrap connector**' are shown in the table below:

Pin No.	Title	I/O	Description
1	START1	I	Note used
2	START(+)	I	Remote <Start> Input Signal (3V –12 V DC)
3	FAIL	O	FAIL (output)
4	OK	O	PASS (output)
5	SPARE (ACTIVE)	O	ACTIVE signal (output)

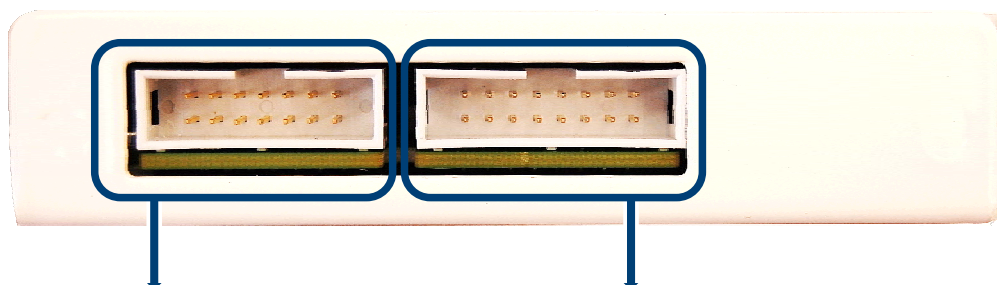
Please note:

- Each pin is also labelled on the circuit board next to the relevant pin.
- The pin labelled '**SPARE**' is the '**ACTIVE**' pin

3.3 ISPnano programmer – Remote Control Port

3.3.1 Overview

The signals required to remote control the ISPnano programmer can be found on the 14-way IDC **'Remote Control Port'** connector. The illustration below details the function of each of the connectors on the rear panel of the programmer.



Programmer Remote Control Port

- This port is used to control the programmer from e.g. an **ATE** via a **"4-wire TTL Interface"**.
- The **"Remote Status LEDs"** PASS, BUSY, FAIL are also on this port. This allows LEDs to be mounted on the lid of the Test Fixture for easier visibility.

Target ISP Connection Port

- This Port features all the Target I/O Signals."
- 5 x Programmer controlled I/O lines Multiplexed for SPI, JTAG, UART, BDM, PDI
- 1 x Programmer Output line (e.g. Relay control)
- Dedicated 2-wire I2C Port
- Controlled Target Vcc Supply

3.3.2 ISPnano - Remote Control Port – pin-out

The ISPnano **'Remote Control'** port features all the signals required to implement TTL control of the programmer. The pins required for remote control are detailed in the table below.

Pin No.	Pin name	I/O	Description
7	REMOTE_START	I	Remote <Start> Input Signal (3V –12 V DC)
9	REMOTE_FAIL	O	FAIL (output)
11	REMOTE_OK	O	PASS (output)
13	REMOTE_BUSY	O	ACTIVE signal (output)

4.0 Standalone Programming Projects

4.1 Overview

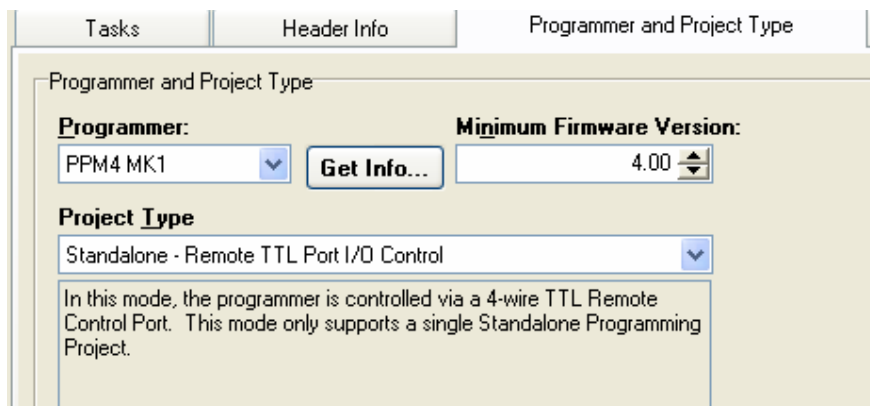
When using the '**Remote TTL I/O Control**' mode, it is recommended that a single pre-compiled '**Standalone Programming Project**' is uploaded into the '**Programmer FLASH Memory Store**' **before** production mode is started. In this mode it is only possible to control a single programming project at any one time as there is no mechanism for the '**Remote System**' to select from more than one project. It is also possible to upload multiple projects to the programmer and then allow the operator to decide which project to execute.

4.2 Creating a Standalone Programming Project

The process of creating a '**Standalone Programming Project**' is described in detail in the relevant application note for the device you are trying to program. The only difference to a standard project is that the '**Project Type**' must now be set to '**Standalone – Remote I/O Control**'.

To create a TTL Mode control project:

- Select **<Project Builder> <Create new project>**
- On the **<Programmer and Project type>** screen, select:
 - Programmer: e.g. **PPM3-MK2**
 - Project type: **Standalone – Remote TTL Port I/O Control**



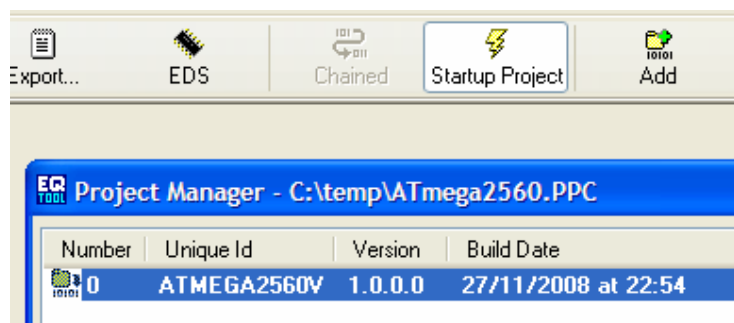
- Set up the FLASH / EEPROM / Fuses / Security Fuses as normal
- Compile the project
- Add the Project Collection to a new / existing Project Collection
- Upload to the programmer using the Upload Wizard

4.3 Forcing the Project to auto-start

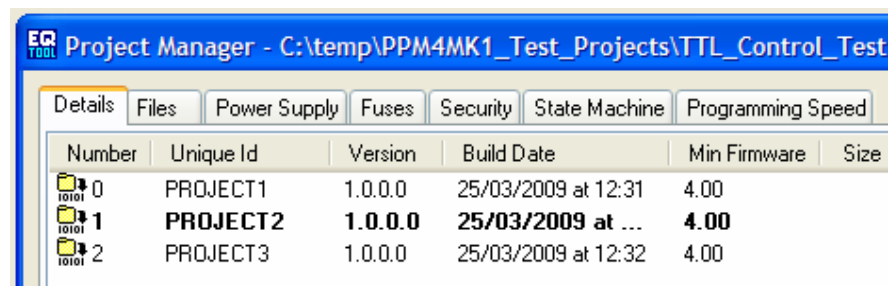
When the '**Standalone Programming Project**' is uploaded to the programmer, it will not automatically start. This means that an operator will have to manually press the **<YES>** key on the programmer once every time the programmer is powered up.

To make it so the uploaded project always starts automatically without any user intervention, you need to make it a 'Start up' project. This can be achieved as follows:

- Add the project to a Project Collection
- Open the Project Collection so that the project is displayed in the 'Project Manager' window.
- Right-click over the project and then select '**Startup**' from the drop-down menu or click the '**Startup Project**' icon



- The selected project will now be highlighted in black to indicate that it has been set up as a '**Startup Project**'.
- If you have multiple projects in the collection, then simply highlight the project which you wish to auto-start and then click the **<Startup Project>** icon.



- The selected project will now be highlighted in black to indicate that it has been set up as a '**Startup Project**'.
- When this Project Collection is uploaded to a programmer, the selected '**Startup Project**' will automatically start.
- In the '**Remote TTL I/O Control**' mode, this means the programmer will go straight into the "**WAITING for START SIGNAL**" mode. i.e. The operator does not have to press any keys on the programmer to start the sequence.

4.4 Uploading a single project to a programmer

Uploading a single '**Standalone Programming Project**' to the programmer is performed using EQTools / Upload Wizard in the normal way – see EQTools – Uploading Projects - Application Note – AN117.

Please note

It is recommended that you only upload ONE project at any time to a programmer when using the '**Remote TTL I/O Control**' mode mode.

4.5 Testing a single project on the programmer

If you have not set up the project as a '**Startup Project**' then the very first time you upload the project to the programmer, the project will not start up automatically. You will need to manually select the project via the programmer keypad and then press the red **<YES>** button to execute the project.

Instructions

- Upload the '**Standalone Programming Project**' to the programmer
- If the project has been set as a '**Startup Project**' then it will start immediately after power up.
- If not, then press the red **<YES>** button to execute the project
- The LCD will now display: "**WAITING for START SIGNAL**" (PPM3-MK2 and PPM4-MK1 programmers only)
- The yellow '**BUSY**' LED will illuminate permanently.
- The programmer is now waiting for the '**Remote System**' to assert the '**START**' signal.
- To manually start the project, use a resistor to short the '**START**' signal to 0V
- The programmer should now start to program the Target Device.
- Once the programming project has completed execution, the **PASS** or **FAIL** LED will illuminate.
- Remove the resistor from '**START**' signal or tie the '**START**' signal to Vcc.
- The programmer LCD should go back to displaying "**WAITING for START SIGNAL**"

4.6 Uploading multiple projects to the programmer

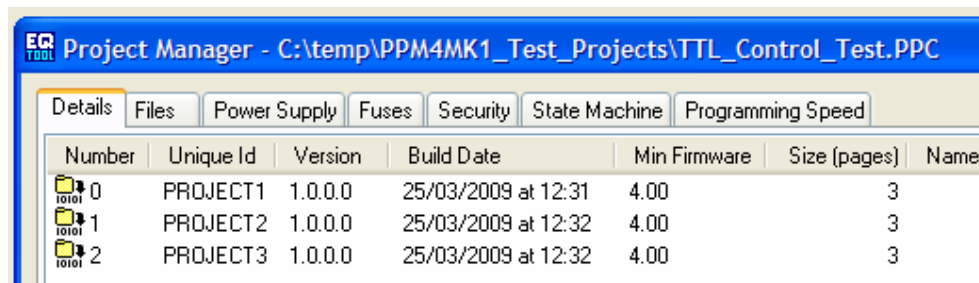
4.6.1 Overview

It is possible to upload multiple projects to the programmer and then allow the operator to select which project to execute by using the programmer keypad. This mode can be useful if the programmer is being controlled by e.g. a PLC and there is no PC available to change the projects. The operator can simply halt production, change the selected project via the programmer keypad and then re-start production without needing to use a PC to upload new projects.

4.6.2 Instructions

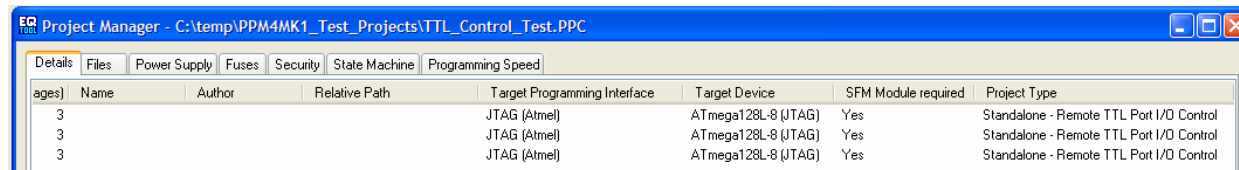
To create an upload multiple projects to the programmer, please follow the instructions below:

- Create a **'Project Collection'** containing all the required projects which are to be uploaded to the programmer.
- In the example below, there are 3 projects in the collection.



Number	Unique Id	Version	Build Date	Min Firmware	Size (pages)	Name
0	PROJECT1	1.0.0.0	25/03/2009 at 12:31	4.00	3	
1	PROJECT2	1.0.0.0	25/03/2009 at 12:32	4.00	3	
2	PROJECT3	1.0.0.0	25/03/2009 at 12:32	4.00	3	

- Make sure all projects have the **'Project Type'** set to **'Standalone - Remote TTL Port I/O Control'** mode. You can view this in the **'Details'** tab in Project Manager – see screenshot.



Pages	Name	Author	Relative Path	Target Programming Interface	Target Device	SFM Module required	Project Type
3				JTAG (Atmel)	ATmega128L-8 (JTAG)	Yes	Standalone - Remote TTL Port I/O Control
3				JTAG (Atmel)	ATmega128L-8 (JTAG)	Yes	Standalone - Remote TTL Port I/O Control
3				JTAG (Atmel)	ATmega128L-8 (JTAG)	Yes	Standalone - Remote TTL Port I/O Control

- If you want the programmer to automatically start with a preset project, set the required project as the **'Startup Project'**.
 - Make sure the programmer keypad is unlocked so the project can be changed via the keypad – see section 4.6.2.
 - Upload the **'Project Collection'** to the programmer.
 - Switch the programmer OFF and then ON again.
- The programmer should now automatically jump to your selected **'Startup Project'**.
- If you now wish to change the selected project, press the **<NO>** button on the keypad once to quit the current project and then select the new project to execute.
 - The programmer will remember the new project setting and this will become the new **'Startup Project'**.

Warning!

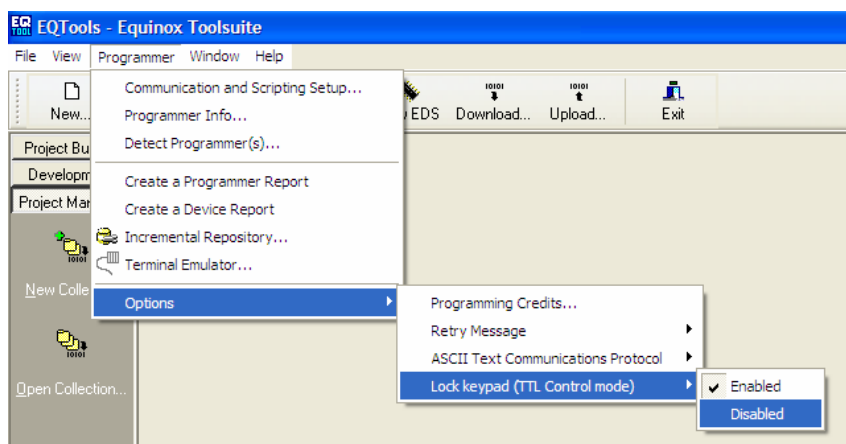
If multiple projects are uploaded to the programmer, it is possible for the operator to select the wrong project by mistake. It is therefore recommended that the operator visually checks the correct project is selected at the start of every production run.

4.6.3 Unlocking the programmer keypad

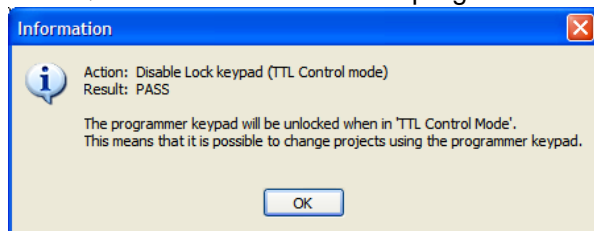
The PPM3-MK2 and PPM4-MK1 programmers can store up to 64 '**Standalone Programming Projects**' in memory. It is usually possible for the operator to select from the list of projects using the programmer keypad. However, when the programmer is in '**TTL Control Mode**', it will not respond to any key presses on the programmer keypad. The keypad is "**locked**" by default so that an operator can not inadvertently change to a different project by mistake.

If you wish to make it so the programmer keypad is "unlocked" allowing the operator to change projects using the programmer keypad, please follow the instructions below.

- Connect the programmer to the PC
- Launch EQTools
- Check that you are using **EQTools build1058** or above
- Check that the attached programmer is running **firmware version 5.02** or above as the ability to unlock the programmer keypad was introduced in this version.
(To check the programmer firmware, select **<Programmer><Programmer Info>**)
- To unlock the programmer keypad, select **<Programmer> <Options> <Lock Keypad (TTL Control Mode)>**



- Select '**Disabled**'
- EQTools will confirm that the programmer keypad is unlocked.



- The keypad should now be unlocked.

Please note:

The setting for locking the keypad is now stored permanently in the programmer EEPROM memory, not in your programming project(s). This procedure needs to be followed on every programmer on which you wish to unlock the keyboard.

4.6.4 Selecting a new project via the keypad (unlocked mode)

Once the '**TTL Mode – Keypad Lock**' has been disabled, it is then possible to select a new project on the programmer by following the instructions below:

- Power up the programmer
- If the programmer is displaying "**WAITING for START SIGNAL**"
- Press the **<NO>** button on the keypad once.
 - The programmer should display "**Select Project**" and the currently selected project name.
- Use the **<Up>** and **<Down>** keys to select the new project. (There must be more than one project in the programmer for this to work.)
- Press the **<Yes>** to select the new project.
 - The new project will automatically start and the programmer should display "**WAITING for START SIGNAL**"
- The programmer will remember the new project setting and this will become the new '**Startup Project**'.
- The next time the programmer is powered OFF and then ON again, it will remember the new settings and will automatically start the new '**Startup Project**'.