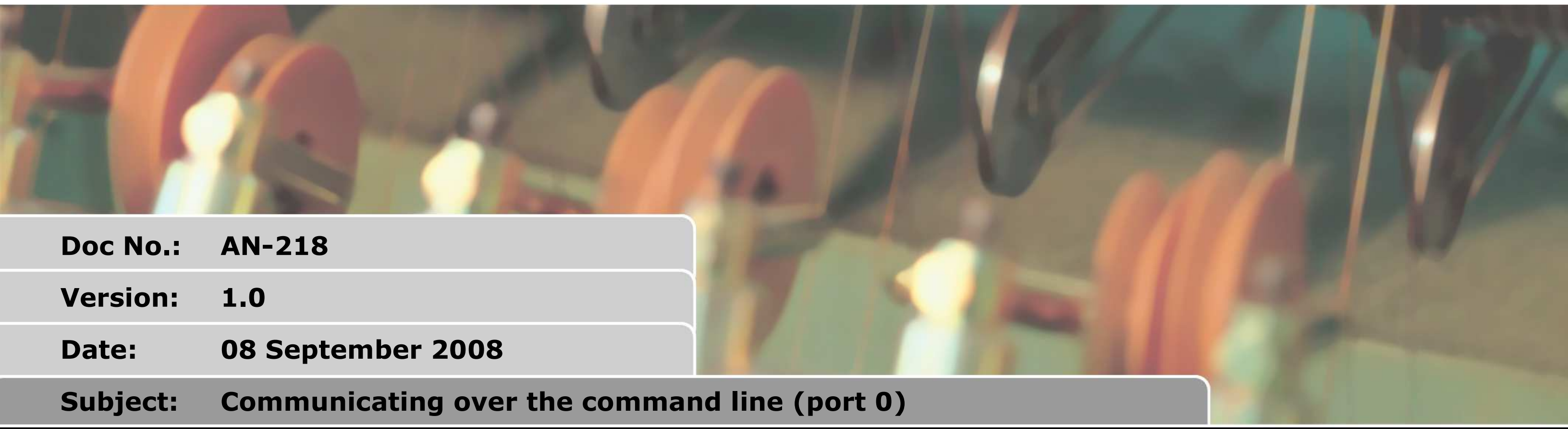
Trio Motion Technology Ltd.
Shannon Way, Tewkesbury,
Gloucestershire. GL20 8ND
United Kingdom
Tel: +44 (0)1684 292333
Fax: +44 (0)1684 297929

1000 Gamma Drive
Suite 206
Pittsburgh, PA 15238
United States of America
Tel: +1 412.968.9744
Fax: +1 412.968.9746

Tomson Centre
118 Zhang Yang Rd., B1701
Pudong New Area, Shanghai,
Postal code: 200122
P. R. CHINA
Tel/Fax: +86-21-58797659



Doc No.: **AN-218**

Version: **1.0**

Date: **08 September 2008**

Subject: **Communicating over the command line (port 0)**

APPLICATION NOTE

www.triomotion.com

1. Background

It is a common request to be able to communicate to the controller from a PC or other device over the command line, port 0. The first recommendation is to use the Trio ActiveX components as these have many built in features to simply interface with the controller. Some customers cannot use the ActiveX and so have to manually interface to the controller. In this situation we recommend using the second serial port (port 1) as you can simply write a small program that receives characters and translates it into the required action. By using port 1 you automatically ignore all the command line information as it is being transmitted over a completely different connection. The other advantage of using port 1 is that you can connect to Motion Perfect 2 and debug your program at the same time. There are still some situations where it is not practical to use port 1 as listed below and port 0 has to be used.

- Not communicating from a PC so cannot use the ActiveX components
- Using a MC302 series controller which does not support the ActiveX
- Programming language does not support ActiveX components
- Port 1 is in use with another protocol or to another device.

1.1. Warning

The main purpose of channel mode is to provide a method for Motion Perfect and the ActiveX components to communicate with the controller. As the controllers develop there is a chance that the channel mode may alter effecting program operation or timings. Please bear this in mind when deciding to communicate over the command line and changing system software versions.

2. Channel Mode

The command line can operate using a channel mode, this is what motion perfect uses to implement the different terminal windows. To enable channel mode you must send the command MPE(1) to channel 0 on the command line. There are five different channels as listed below:

- Channel 0 – the command line
- Channel 5 – user programmable channel
- Channel 6 – user programmable channel
- Channel 7 – user programmable channel
- Channel 9 – error message channel

From the controller you wrote to a channel using the PRINT#channel command.

When in normal mode (MPE(0)) all of these channels are displayed through channel 0. When channel mode is selected (MPE(1)) an escape character is sent before the channel data so for example if your program included PRINT#5, "Hello World" you would see "<esc>5Hello World" printed out of Port 0.

It is important not to get confused between port and channel, Port is the connection to the controller, Channel is a virtual connection on port 0.

To connect using channel mode your application program must parse the received data for <esc> characters and assign the characters to the correct channel. If you can use Trio PCMotion ActiveX this is done automatically in asynchronous mode.

3. Example programs

The example program was written in Visual Basic Express 2008. The aim was to demonstrate channel filtering and a simple method of communicating data and controlling motion. As you have full access to the command line it is possible to send any command that is accepted on the command line (most of them!). This program uses VR(0) to run routines which are pre determined in the controller. The slowest link in the chain is the communications so by simplifying the messages sent to the controller you can increase the overall speed of the process. The same theory applies for running motion over the PCMotion component.

3.1. Visual basic program

The Channel filter example uses 3 rich text boxes to display program information on channel 5, error text on channel 6. The command line is the remaining channels.

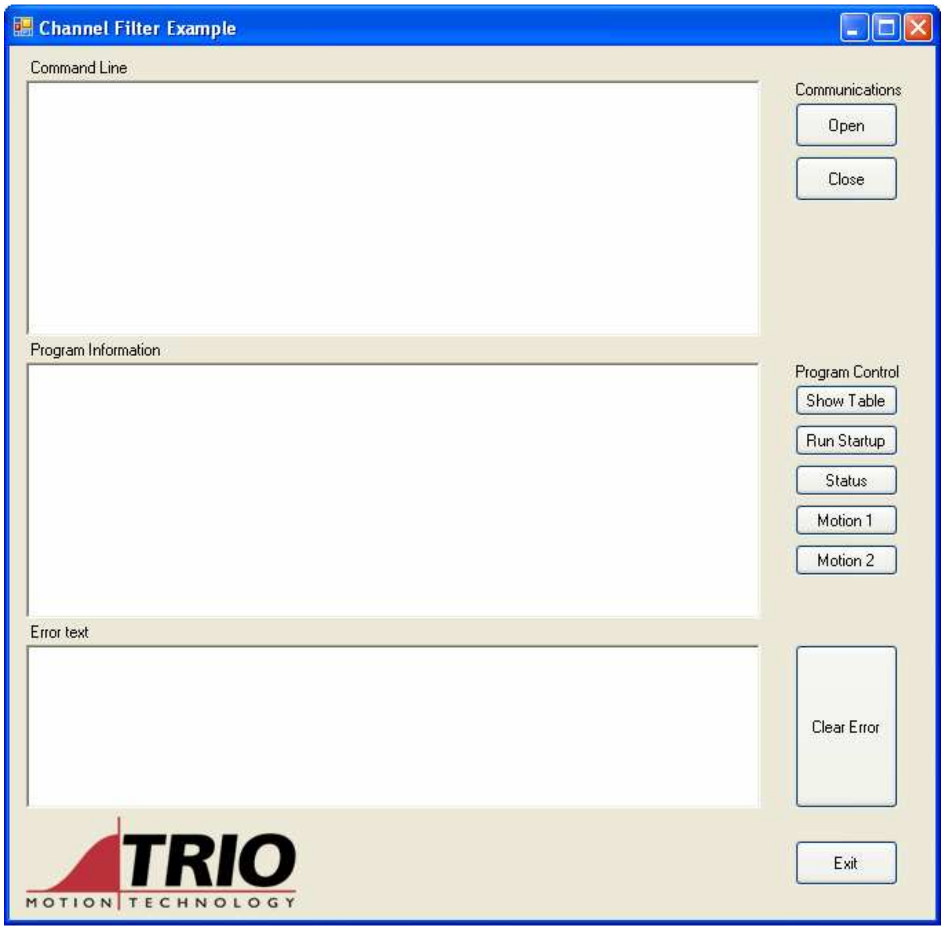
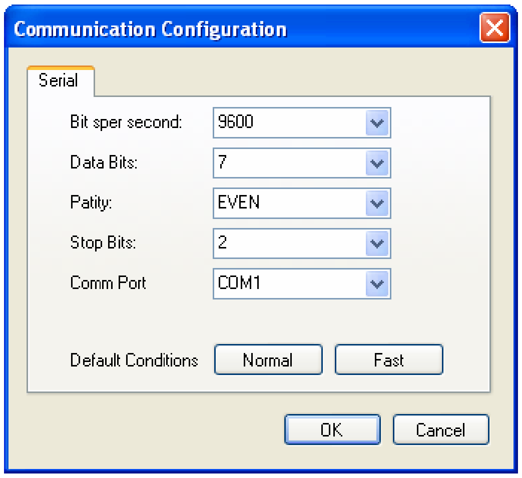


Figure 1Channel Filter Example program

3.1.1. Open

The open button brings up a dialogue to select the correct port settings. As shown in Figure 2 you can set the standard parameters manually or select the normal or high speed default settings. When you press the OK button these parameters are stored in the CCOMMS class, the selected communication channel is then opened. Once opened the program attempts to find the command line, if it is successful it transmits a MPE(1) to set channel mode. You will see an echo of this in the command line as in Figure 3.


Communication Configuration

Serial

Bit sper second:	9600
Data Bits:	7
Patity:	EVEN
Stop Bits:	2
Comm Port	COM1

Figure 2 Communications configuration dialogue

```
Successfully opened comms port.
mpe(1)
>>
```

Figure 3 Successful Open

3.1.2. Close

This button simply releases the serial port so you can connect to Motion Perfect 2 or another program.

3.1.3. The Command line

This displays any characters received on channel 0, 7 and 9. You can also type here and the characters are sent to the controller on channel 0. It is only a simple implementation of the command line so many features do not work as the command line in motion perfect, for example backspace does not delete characters.

3.1.4. Program information

Any characters sent using channel 5 will be displayed in this window, you can test it by typing PRINT#5, “Hello channel 5” in the command line.

3.1.5. Error information

Any characters sent using channel 6 will be displayed in this window, you can test it by typing PRINT#5, “Hello channel 6” in the command line.

3.1.6. Program Control buttons

The program control buttons simply transmit “VR(0)=” and a value. This is a very small overhead as far as communications but can activate a powerful set of instructions on the controller. By communicating in this fashion you can also take advantage of controller only command such as WAIT IDLE where the program will suspend until the motion command is complete. To do this over the command line you will have to poll the MTYPE status repeatedly until you see it is idle. Not only is this a big overhead but it will slow the operation of your application.

The buttons call simple routines in the controller which run programs, transmit data or start motion. Please see the Trio Basic program for full functionality.

3.1.7. Clear Error

The clear error button transmits a “VR(1)=1”, this starts a routine in the controller to clear any motion errors. The error message is also cleared from the error text window.

3.1.8. Exit

The exit button closes the communications channel and quits the example program.

3.2. Trio Basic Program

The program in the controller is running a loop that checks for motion errors and for a value in VR(0). When it finds a value one of 5 subroutines are called, please see the program for details on each function.

By running a program that checks for the change in a VR value you remove the processing time for a program to start which can be in excess of 100msec. This program start time can be demonstrated with the 2 motion routines. The first runs the routine in a separate program, the second runs the routine direct in the Main program, the time taken for each is displayed for you to compare.