

# Packet-Master USB12 Packet-Master USB480 Packet-Master USB480+ Packet-Master USB500 AG User Manual

**MQP Electronics Ltd**  
**Unit 2, Park Road Centre**  
**Malmesbury**  
**Wiltshire SN16 0BX**  
**United Kingdom**

e-mail: [sales@mqp.com](mailto:sales@mqp.com)

Website: [www.mqp.com](http://www.mqp.com)

User Manual Packet-Master-2.07

Copyright © 2006-2009 MQP Electronics Ltd

## Contents

<b>1</b>	<b>GETTING STARTED .....</b>	<b>8</b>
<b>1.1</b>	<b><i>Introduction .....</i></b>	<b>8</b>
<b>1.2</b>	<b><i>Installing the Software and Driver .....</i></b>	<b>9</b>
1.2.1	Install the Software from CD .....	9
1.2.2	Installing the Driver .....	9
1.2.3	Updating the Driver .....	10
1.2.4	Updating the Software.....	10
<b>1.3</b>	<b><i>Sample Capture Files.....</i></b>	<b>11</b>
<b>1.4</b>	<b><i>Front and Back Panels – Packet-Master USB12.....</i></b>	<b>12</b>
<b>1.5</b>	<b><i>Front and Back Panels – Packet-Master USB480(+) .....</i></b>	<b>14</b>
<b>1.6</b>	<b><i>Front and Back Panels – Packet-Master USB500 AG .....</i></b>	<b>16</b>
<b>1.7</b>	<b><i>Test Set Up - Analyser .....</i></b>	<b>18</b>
<b>1.8</b>	<b><i>Data Capture.....</i></b>	<b>19</b>
<b>1.9</b>	<b><i>Saving the Capture.....</i></b>	<b>20</b>
<b>1.10</b>	<b><i>Test Set Up - Generator .....</i></b>	<b>21</b>
<b>2</b>	<b>TECHNICAL DATA .....</b>	<b>22</b>
<b>2.1</b>	<b><i>Requirements .....</i></b>	<b>22</b>
<b>2.2</b>	<b><i>Specifications USB12 .....</i></b>	<b>22</b>
<b>2.3</b>	<b><i>Specifications USB480(+) .....</i></b>	<b>23</b>
<b>2.4</b>	<b><i>Specifications USB500 AG .....</i></b>	<b>23</b>
<b>2.5</b>	<b><i>Maximum Capture File Size.....</i></b>	<b>24</b>

<b>2.6</b>	<b>Safety .....</b>	<b>24</b>
<b>2.7</b>	<b>Feature Connector – USB12.....</b>	<b>24</b>
<b>2.8</b>	<b>Feature Connector – USB480 (+) and USB500 AG in Analyser Mode.....</b>	<b>25</b>
<b>2.9</b>	<b>Feature Connector –USB500 AG in Generator Mode.....</b>	<b>26</b>
<b>2.10</b>	<b>External Power Supply - USB12 or USB480(+). ....</b>	<b>27</b>
<b>2.11</b>	<b>External Power Supply – USB500 AG .....</b>	<b>27</b>
<b>2.12</b>	<b>Trigger Specification .....</b>	<b>28</b>
<b>2.13</b>	<b>Measurement Accuracy (USB500 AG) .....</b>	<b>29</b>
2.13.1	Voltage.....	29
2.13.2	Current.....	29
<b>3</b>	<b>GRAPHICUSB SOFTWARE - ANALYSER.....</b>	<b>30</b>
<b>3.1</b>	<b>Overview .....</b>	<b>30</b>
<b>3.2</b>	<b>Panes.....</b>	<b>31</b>
3.2.1	Event Pane.....	31
3.2.2	Analysis and Data Panes .....	32
3.2.3	Timeline and Bandwidth Panes.....	33
<b>3.3</b>	<b>Pane Properties.....</b>	<b>35</b>
3.3.1	Event Pane Properties .....	35
3.3.2	Detail Pane Properties .....	35
3.3.3	Data Pane Properties.....	35
3.3.4	Timeline Pane Properties.....	35
<b>3.4</b>	<b>Capture Summary .....</b>	<b>36</b>
<b>3.5</b>	<b>Toolbars .....</b>	<b>37</b>
3.5.1	File Functions Toolbar.....	37
3.5.2	View Filter Toolbar .....	39
<b>3.6</b>	<b>Capturing .....</b>	<b>40</b>

<b>3.7</b>	<b><i>Device Information Files</i></b> .....	<b>42</b>
3.7.1	Creating a Device Information File.....	43
3.7.2	Using a Device Information File .....	45
3.7.3	Automatic Assumption Assignment.....	48
3.7.4	Device Information File Syntax Rules .....	50
<b>3.8</b>	<b><i>Triggering (USB480)</i></b> .....	<b>52</b>
3.8.1	Trigger As Input.....	52
3.8.2	Trigger As Output.....	52
3.8.3	Trigger Not In Use.....	53
<b>3.9</b>	<b><i>Advanced Triggering (USB480+ and USB500 AG)</i></b> .....	<b>54</b>
3.9.1	Operation .....	54
3.9.2	Events Available For Triggering .....	55
3.9.3	Modules.....	56
3.9.4	Sequences .....	59
3.9.5	Token Packets .....	61
3.9.6	Data Packets.....	62
3.9.7	Split Packets .....	63
3.9.8	Miscellaneous Events .....	64
3.9.9	Other Settings .....	65
3.9.10	Preset Trigger Settings .....	66
3.9.11	Manual Triggering .....	68
3.9.12	BNC Connector.....	69
3.9.13	Trigger Off.....	70
3.9.14	Save Settings.....	70
3.9.15	Load Settings .....	70
3.9.16	Finding the Trigger Points.....	70
<b>3.10</b>	<b><i>Display Filters</i></b> .....	<b>72</b>
<b>3.11</b>	<b><i>Custom Filter</i></b> .....	<b>79</b>
3.11.1	Custom Filter Settings.....	79
3.11.2	Custom Filter Enable .....	83
<b>3.12</b>	<b><i>Search</i></b> .....	<b>84</b>
3.12.1	Event Search .....	84
3.12.2	Transaction Search .....	85
3.12.3	Data Search .....	86

3.12.4	Setup Search .....	87
3.12.5	Error Search.....	88
<b>3.13</b>	<b><i>Multiple Event Headers.....</i></b>	<b>89</b>
<b>3.14</b>	<b><i>Bookmarks.....</i></b>	<b>89</b>
<b>3.15</b>	<b><i>Printing.....</i></b>	<b>91</b>
<b>3.16</b>	<b><i>Option Settings .....</i></b>	<b>91</b>
3.16.1	File Locations.....	91
3.16.2	Miscellaneous Settings .....	92
3.16.3	Capture .....	93
<b>3.17</b>	<b><i>Command Line Capture Control .....</i></b>	<b>94</b>
3.17.1	Start Capture.....	94
3.17.2	Restart Capture.....	94
3.17.3	Stop Capture .....	95
3.17.4	Display File .....	95
<b>3.18</b>	<b><i>Export Functions.....</i></b>	<b>96</b>
3.18.1	Exporting Capture Events .....	96
3.18.2	Exporting Capture Events – Data Only .....	98
3.18.3	Exporting Descriptors.....	98
3.18.4	Exporting Data from a Specific Event .....	102
<b>3.19</b>	<b><i>Text Editing.....</i></b>	<b>103</b>
3.19.1	Introduction .....	103
3.19.2	Editing .....	104
3.19.3	Bookmarks .....	104
3.19.4	Error Messages.....	104
<b>3.20</b>	<b><i>USB Errors.....</i></b>	<b>105</b>
3.20.1	Invalid PID.....	105
3.20.2	Invalid CRC.....	105
3.20.3	Invalid SOF .....	105
3.20.4	Invalid Control Transfer.....	105
3.20.5	Invalid Transaction.....	106
3.20.6	Bit Stuffing Error.....	106
3.20.7	Byte Error.....	106

3.20.8	Spurious Data .....	106
3.20.9	Both Lines High.....	106
3.20.10	Spurious End of Packet .....	106
<b>3.21</b>	<b><i>Class Analysis Options .....</i></b>	<b>107</b>
3.21.1	Registration.....	107
3.21.2	Analysis Overview.....	108
3.21.3	Vendor Class Analysis .....	111
<b>3.22</b>	<b><i>V<sub>BUS</sub> Current and Voltage Measurement (USB500 AG) .</i></b>	<b>121</b>
<b>3.23</b>	<b><i>Firmware Updates .....</i></b>	<b>123</b>
<b>4</b>	<b>GRAPHICUSB SOFTWARE – GENERATOR .....</b>	<b>128</b>
<b>4.1</b>	<b><i>Introduction .....</i></b>	<b>128</b>
<b>4.2</b>	<b><i>Generator Operation – Host / Device Emulation .....</i></b>	<b>128</b>
<b>4.3</b>	<b><i>Generator Operation - Retry.....</i></b>	<b>129</b>
<b>4.4</b>	<b><i>Creating a Generator Script from a Capture file.....</i></b>	<b>129</b>
<b>4.5</b>	<b><i>Creating a Generator Script from Scratch. ....</i></b>	<b>133</b>
<b>4.6</b>	<b><i>Generator Script Language Commands .....</i></b>	<b>138</b>
<b>4.7</b>	<b><i>Generator Script Language Pre-defined Values.....</i></b>	<b>147</b>
<b>4.8</b>	<b><i>Generator Script Language Syntax Rules .....</i></b>	<b>150</b>
4.8.1	Command Sequence .....	150
4.8.2	Case Sensitivity.....	150
4.8.3	Command Lines .....	150
4.8.4	Labels .....	151
4.8.5	Comments.....	151
4.8.6	Tabs .....	151
4.8.7	Data Values.....	151
4.8.8	Execution .....	152
<b>4.9</b>	<b><i>Test Configurations .....</i></b>	<b>153</b>

<b>4.10</b>	<b><i>Compiling the Generator Script.....</i></b>	<b>156</b>
<b>4.11</b>	<b><i>Running the Generator Script.....</i></b>	<b>157</b>
4.11.1	Different Computer Hosting Analyser.....	157
4.11.2	Same Computer Hosting Analyser.....	158
<b>4.12</b>	<b><i>Generator Error Messages .....</i></b>	<b>159</b>
<b>4.13</b>	<b><i>Generator Script Suggestions .....</i></b>	<b>161</b>
<b>4.14</b>	<b><i>Generator Scripts For On-the-Go .....</i></b>	<b>163</b>
4.14.1	Host Negotiation Protocol – Full Speed .....	163
4.14.2	Host Negotiation Protocol – High Speed.....	165
4.14.3	Session Request Protocol – Full Speed.....	167
4.14.4	Session Request Protocol – High Speed .....	168
<b>4.15</b>	<b><i>Generator Limitations.....</i></b>	<b>169</b>
<b>5</b>	<b>TROUBLESHOOTING .....</b>	<b>170</b>
<b>6</b>	<b>WARRANTY.....</b>	<b>173</b>
<b>6.1</b>	<b><i>Warranty.....</i></b>	<b>173</b>
<b>6.2</b>	<b><i>Limitations .....</i></b>	<b>173</b>
<b>6.3</b>	<b><i>Warranty Period.....</i></b>	<b>173</b>
<b>6.4</b>	<b><i>Obtaining Service.....</i></b>	<b>173</b>

# 1 GETTING STARTED

## 1.1 *Introduction*

Packet-Master is a series of non-intrusive Hardware USB Bus Analysers. The USB12 is intended for development of Low and Full Speed USB devices and hubs etc, whilst the USB480 additionally supports High Speed. Each analyser comes complete with our Windows application GraphicUSB for capturing and displaying every detail of the data interactions on a USB link.

The Packet-Master USB500 AG provides the functionality of either a High/Full/Low Speed Analyser, or a High/Full/Low Speed Generator. In Generator mode it can emulate a host, to exercise a device under development, or it can emulate a device, to exercise a host under development.



## 1.2 Installing the Software and Driver

We suggest that you first install the software from the CD (or download), before plugging in your analyser. This leads to the simplest procedure, as the installer will also pre-install the driver.

### 1.2.1 Install the Software from CD

- Insert the Installation disk into the CD drive.
- The disk should auto-start.
- Follow the on screen instructions.
- If the disk doesn't auto-start, then run the file GraphicUsb\_setup.exe in the root directory of the CD.

### 1.2.2 Installing the Driver

- The first time you plug in the USB cable from your Packet-Master analyser, assuming that you have already installed the software as above, Windows will automatically complete the driver installation, and inform you that it completed successfully.  
On XP, the 'Found new Hardware Wizard' will appear. Answer questions as follows:  
Q. *Can Windows connect to Windows Update to search for software?*  
A. *Not this time.*  
Q. *What do you want the wizard to do?*  
A. *Install the software automatically.*  
On Vista the process will proceed without intervention.
- If you have not installed the application, Windows will start the "Found new Hardware" wizard. If it asks to search "Windows Update", select "No, not this time".
- Ensure that you have the Installation CD in a CD drive. *(if the CD auto-runs and starts the GraphicUSB installation screen, then click "Exit" to leave it before continuing with the driver installation.)*
- For your information: If the CD drive is drive "D:", the driver files are located in "D:\drivers\" and the installation file is called "mqpuba.inf".

### 1.2.3 Updating the Driver

- If there is a requirement to update the USB Driver for the Packet-Master analyser, it will automatically be updated when you install the new version of the application. See the details above for differences between XP and Vista.
- For your information, the driver package is located in:  
"C:\Program Files\MQP Electronics\GraphicUSB\usb drivers\  
(Assuming a default location for the GraphicUSB installation)
- On 64-bit versions of Windows the pathname will be:  
"C:\Program Files (x86)\MQP Electronics\GraphicUSB\usb drivers\"

### 1.2.4 Updating the Software

If at a later date you wish to make use of an update from our website. Please follow the instructions below:

- Download the file.
- Run the downloaded .exe file straight from your hard disk and follow the on screen instructions.
- The latest version of the software is available at  
<http://www.mqp.com/>

We make frequent improvements and enhancements to our software so it is well worth checking on our website for new versions.

### ***1.3 Sample Capture Files***

During installation of the software a number of sample capture files will be placed in the folder “Samples” in the application’s data directory. This is the default location when opening a file from the File menu.

The sample files have the extension \*.mqu  
You may find it helpful to open one of these sample files to become familiar with the capabilities of GraphicUSB.

## ***1.4 Front and Back Panels – Packet-Master USB12***



- The Power Indicator illuminates when the Packet-Master is powered and connected to the Host computer.
- The Activity indicator shows the presence of data exchanges. The indicator flashes once for each DATA0, DATA1 or SETUP packet.
- Capture Start and Capture Stop buttons control the capturing of USB data, and the Capture Indicator shows when capturing is in progress.
- The USB through-connectors are used for connection to the Host and device under test.



- The Packet-Master is normally USB Bus Powered for convenience of use.
- An optional external power input is provided (this is not normally required, but useful if the host won't configure a high-power device)
- A High Speed USB (480 Mbit/s) provides the connection to the Host PC.
- A 10-pin Feature connector provides signals for an oscilloscope or logic analyser.

## 1.5 Front and Back Panels – Packet-Master USB480(+)



- The Power Indicator illuminates when the Packet-Master is powered and connected to the Host computer.
- The Activity indicator shows the presence of data exchanges. The indicator flashes once for each DATA0, DATA1 or SETUP packet.
- The Trigger indicator shows whether the hardware trigger is enabled, and whether it is an input (red) or an output (green).
- Capture Start and Capture Stop buttons control the capturing of USB data, and the Capture Indicator shows when capturing is in progress.
- The USB through-connectors are used for connection to the Host and device under test.



- The Packet-Master is normally USB Bus Powered for convenience of use.
- An optional external power input is provided (this is not normally required, but useful if the host won't configure a high-power device)
- A High Speed USB (480 Mbit/s) provides the connection to the Host PC.
- A 10-pin Feature connector provides signals for an oscilloscope or logic analyser.
- A BNC connector provides a convenient location to connect an external trigger source, or external equipment to be triggered.

## 1.6 Front and Back Panels – Packet-Master USB500 AG



- The Power indicator illuminates when the Packet-Master is powered and connected to the Host computer.
- The Activity indicator shows the presence of data exchanges. The indicator flashes once for each DATA0, DATA1 or SETUP packet.
- The Trigger indicator shows whether the hardware trigger is enabled, and whether it is an input (red) or an output (green).
- The Analyser indicator illuminates when the USB500 AG is configured as an analyser, while the Generator indicator is lit when configured as a generator. When in generator mode, two further indicators, Host and Device, show which hardware function is currently being emulated. If the Generator is about to be asked to apply  $V_{BUS}$  when  $V_{BUS}$  is already present on a connected USB cable, the Generator and Host are alternately flashed to indicate an error.
- Capture Start and Capture Stop buttons control the capturing of USB data, and the Capture Indicator shows when capturing is in progress.
- The USB through-connectors are used for connection to the Host and device under test.





- The Packet-Master can be USB Bus Powered for convenience of use, when in use as an analyser. As a generator, it is necessary to use the external power supply, included with the unit, so that sufficient voltage and current are available to the device under test.
- The supply can also be useful in analyser mode, if the host won't configure a high-power device (which the USB500 AG is.)
- A High Speed USB (480 Mbit/s) provides the connection to the Host PC.
- A 10-pin Feature connector provides signals for an oscilloscope or logic analyser.
- A BNC connector provides a convenient location to connect an external trigger source, or external equipment to be triggered.

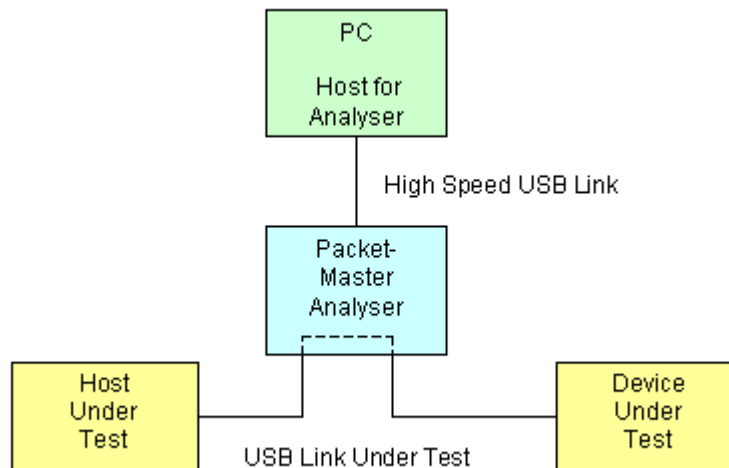
## 1.7 Test Set Up - Analyser

To achieve a good capture rate it is important to provide a suitable test environment. The Packet-Master unit should preferably be hosted by a good performance PC with a high speed USB connection. The USB host controller should not be shared by any other USB device while doing the testing. This is particularly important with the USB480(+) and USB500 AG.

The Packet-Master USB12 will function correctly when connected to a Full Speed link but the capacity will be severely reduced and only low rates of USB traffic may be captured from the device under test.

You should not attempt this with the USB480(+) or USB500 AG.

In a similar fashion it is theoretically possible to use the same host computer for the Packet-Master unit and for the Host under Test. We do not recommend this, but if it is absolutely unavoidable, then you must at least use a separate USB host controller for the device under test. If your computer does not have two host controllers then you will need to buy a plug-in USB host controller card.



It is possible to have more than one device under test. In this case the devices must first be connected to a Hub and the Hub plugged

into the Packet-Master. Both devices and the Hub are downstream of the Packet-Master.

It is not recommended that a Hub be connected upstream of the analyser as traffic from the Host to Hub may be recorded but not the traffic from the Hub to the Host.

This test equipment, by its very nature, has to connect to a junction of two USB cables in order to probe the data on the link. It is important to keep the cables between the 'device under test' and the 'host under test' as short as possible (for example 1m), and to use good quality cables.

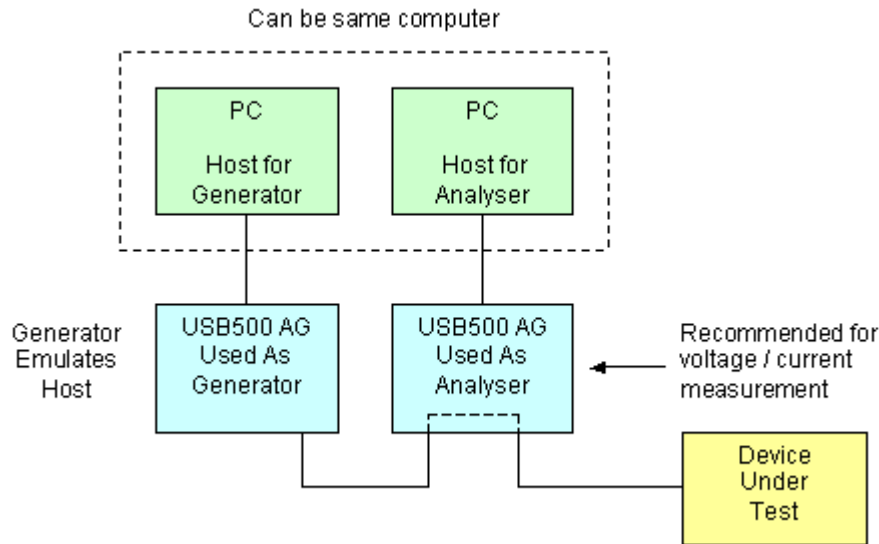
## **1.8 Data Capture**

- Connect the Packet-Master to the Host computer as shown in the Test Set Up section. You must first have installed the GraphicUSB software and Driver.
- Connect the Host under Test to the analyser as shown in the Test Set Up section. Usually, do not connect the device under test yet.
- It may be preferable to use a simple working commercial device such as a low speed mouse for your first data capture.
- Run the GraphicUSB software.
- Start capturing by either momentarily pressing the Start button on the Packet-Master front panel or clicking Capture on the tool bar.
- A capture window will open and you should see the capture statistics incrementing.
- Now is a good time to plug the device under test into the analyser. By capturing the moment of plug-in you will capture the enumeration sequence, which contains information which may help with the rest of the analysis.
- After a few moments stop capturing by either momentarily pressing the Stop Button on the Packet-Master front panel or clicking the Save button in the Capture window.
- A full analysis of the data captured will be displayed.
- Each new capture will create a new capture window. To select the capture you wish to view click on "Window" on the menu bar.

## ***1.9 Saving the Capture***

If you wish to keep the capture for future reference, click on “File...Save” on the menu bar or on the “Save” icon on the Tool Bar. By default, files will be saved in the folder specified in the Options Settings Window. GraphicUSB files have the extension .mqu

## 1.10 Test Set Up - Generator



This is how to connect the generator when emulating a host. The analyser shown is marked USB500 AG, but can in fact be any suitable analyser. The PC which is hosting the generator can be the same as that hosting the analyser, provided both are from the Packet-Master series.

If the analyser in question is from a third party manufacturer, then you should be cautious about performance when using the same PC; we would recommend separate PCs in this case.

Packet-Master USB500 AG has a built-in  $V_{BUS}$  measuring option.

If two USB500 AG units are used, we recommend that the  $V_{BUS}$  voltage and current measurements are specified as being read by the unit used as an analyser. This eliminates any current drawn by downstream analyser units being included in the measurement.

More detail on configuration and using the generator is given later under **GRAPHICUSB SOFTWARE - GENERATOR**.

## 2 TECHNICAL DATA

### 2.1 Requirements

The minimum requirements for the Packet-Master Host are as follows:

- Pentium 3 - 600MHz
- PC with High Speed USB port should be dedicated to the capturing. (Actual target device should be connected to a different Host Computer to ensure reliable capture bandwidth)
- Windows XP (Service Pack 1 or better) or Vista. 32-bit or 64-bit.
- CD ROM / DVD Drive
- 1GB RAM
- 100MB space on Hard Disk

Note: A good performance machine is recommended for good capture rate.

### 2.2 Specifications USB12

Weight:	165 g
Dimensions:	129 x 100 x 31 mm
Temperature:	0°C - 40°C
Humidity:	20% - 80% non condensing
Power:	150mA from USB host Optionally zero mA from USB if powered externally

## 2.3 Specifications USB480(+)

Weight:	360 g
Dimensions:	190 x 140 x 32 mm
Temperature:	0°C - 40°C
Humidity:	20% - 80% non condensing
Power:	225mA from USB host Optionally zero mA from USB if powered externally

## 2.4 Specifications USB500 AG

Weight:	400 g
Dimensions:	190 x 140 x 52 mm
Temperature:	0°C - 40°C
Humidity:	20% - 80% non condensing
Power:	300mA from USB host (analyser mode only) Optionally zero mA from USB if powered externally External power supply (included) required for generator mode operation.

## 2.5 Maximum Capture File Size

The maximum size of the Capture file is limited by the available RAM in the Host computer.

## 2.6 Safety

CE compliant.

## 2.7 Feature Connector – USB12

The signals available on the back panel connector are:

Pin	Signal	Notes
1	D+	De-glitched and synchronised with the 48 or 6MHz clock
2	GND	
3	D-	De-glitched and synchronised with the 48 or 6MHz clock
4	GND	
5	DECODED DATA	Decoded NRZI data
6	GND	
7	DATA CLOCK ENABLE	Data sampling signal phase-locked to the incoming signal transitions
8	GND	
9	PACKET SYNC PULSE	A pulse one clock period wide indicating that the sync pattern has been detected
10	GND	



## **2.8 Feature Connector – USB480 (+) and USB500 AG in Analyser Mode**

The signals available from the transceiver on the back panel connector are:

Pin	Signal	Notes
1	D+	De-glitched and synchronised with the 60MHz clock (FS and LS only)
2	GND	
3	D-	De-glitched and synchronised with the 60MHz clock (FS and LS only)
4	GND	
5	RXVALID	Indicates that a valid data byte has been received.
6	GND	
7	RXACTIVE	Indicates that a start of packet has been detected and that data is being received.
8	GND	
9	60 MHz Clock	
10	GND	

## 2.9 Feature Connector –USB500 AG in Generator Mode

The signals available from the transceiver on the back panel connector are:

Pin	Signal	Notes
1	D+	De-glitched and synchronised with the 60MHz clock (FS and LS only)
2	GND	
3	D-	De-glitched and synchronised with the 60MHz clock (FS and LS only)
4	GND	
5	TXACTIVE	Indicates that a packet is being transmitted.
6	GND	
7	RXACTIVE	Indicates that a start of packet has been detected and that data is being received.
8	GND	
9	60 MHz Clock	
10	GND	

## 2.10 **External Power Supply - USB12 or USB480(+)**

The Packet-Master normally derives its power from the Host computer's USB connection, but may be powered externally. This is not normally required, but useful if the host won't configure a high-power device. The external supply must meet the following requirements:

<b>Output Voltage</b>	9V Regulated
<b>Output Current</b>	300mA (USB12) 500mA (USB480(+))
<b>Polarity</b>	Centre Pin Positive

A suitable Power Supply is available from MQP Electronics Ltd. The use of any other supply is not recommended and is at the users own risk.

## 2.11 **External Power Supply – USB500 AG**

The Packet-Master USB500 AG, operating in Analyser mode normally derives its power from the Host computer's USB connection, but may be powered externally. This is not normally required, but useful if the host won't configure a high-power device.

However, in Generator mode, as the unit has be able to provide a supply to the device of up to 500mA, the external power supply, which we provide with the unit, must be connected to the power socket on the rear.

The external supply must meet the following requirements:

<b>Output Voltage</b>	9V Regulated
<b>Output Current</b>	1A
<b>Polarity</b>	Centre Pin Positive

## 2.12 *Trigger Specification*

### As Input:

Characteristic	Value
Max Voltage allowed (V)	-0.5 to +5.5
VIL max (V)	0.8
VIH min (V)	2

### As Output:

Characteristic	Value	Condition
Max Voltage allowed (V)	-0.5 to +5.5	
Output Impedance ( $\Omega$ )	2k2	
VOL max (V)	0.4	(zero current)
VOH min (V)	2.4	(zero current)

## **2.13 Measurement Accuracy (USB500 AG)**

### **2.13.1 Voltage**

<b>Characteristic</b>	<b>Value</b>
Measurement Range	0 to +6.25V
Accuracy	2% +/- 20mV

### **2.13.2 Current**

<b>Characteristic</b>	<b>Value</b>	<b>Condition</b>
Measurement Range	0 to 600mA	
Accuracy	2% +/- 1.6 mA	> 12.25 mA
Accuracy	2% +/- 32 $\mu$ A	< 12.25 mA

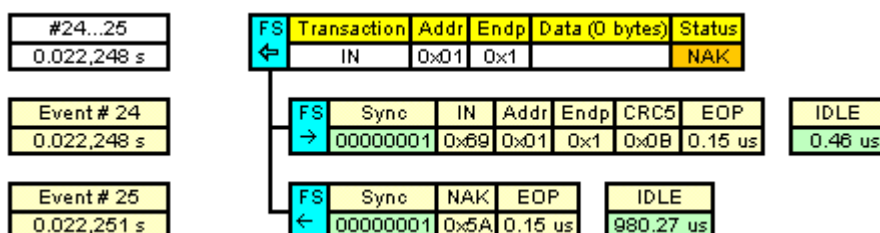


## 3.2 Panes

### 3.2.1 Event Pane

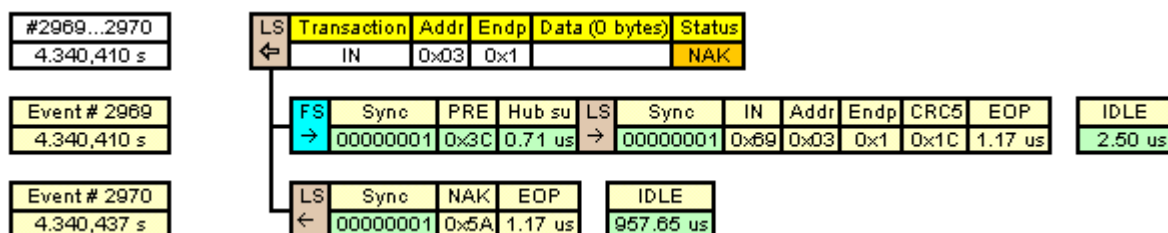
The Event Pane graphically shows every detail of data and timing on the bus. The example shown below is an IN transaction made up of two packets. The left hand column contains an event number and a time stamp. The time stamp has a resolution of 20.83ns for the USB12 and 16.66ns for the USB480(+) and USB500AG. A vertical line associates the packets within the transaction. Looking in detail at the display for event #24 below, the following information is displayed:

- The speed of the transmission. HS is High Speed (480MHz), FS is full speed (12MHz) and LS is low speed (1.5MHz).
- The direction of the packet. A right pointing arrow is for host to device and a left pointing arrow for device to host.
- The Synchronisation field. The bit pattern received is shown.
- The Packet ID (PID). In this case it is an IN token.
- The Address field.
- The Endpoint field.
- The CRC field. Token packets have a five bit CRC while data packets have a sixteen bit CRC.
- End of Packet (EOP). An EOP is made up of a single ended zero for approximately two bit times followed by a J state. The time shown is the length of the single ended zero. For High Speed, an EOP is signalled by a byte of 01111111 (which contains a deliberate bit stuffing error).
- Finally the idle time before the next event.



An error e.g. an incorrect CRC is indicated by the appropriate field being highlighted in red. A field highlighted in orange indicates a potential problem or warning.

This view of a GraphicUSB capture shows data being transferred at two different speeds on the same link. Each packet is preceded by a coloured marker, indicating Full Speed or Low Speed. This makes the function of the PREAMBLE packet very clear. Notice also the detailed timing information for Hub Setup time, End of Packet duration, and IDLE state time.



### 3.2.2 Analysis and Data Panes

By clicking on an event row in the event pane, a complete analysis of the event is displayed in the analysis pane, and the data content is shown in its entirety in the data pane. Where appropriate, any information selected in the analysis pane is highlighted in the data pane for easy identification. All standard requests and descriptors are analysed in detail. Any discrepancies are described.

**i Control Transfer**  
**Get String Descriptor 1**

String descriptors use UNICODE encodings.

Field	Value	Meaning
bLength	16	Valid Length
bDescriptorType	3	String Descriptor
bString	"USB Hub"	

#### Data Content

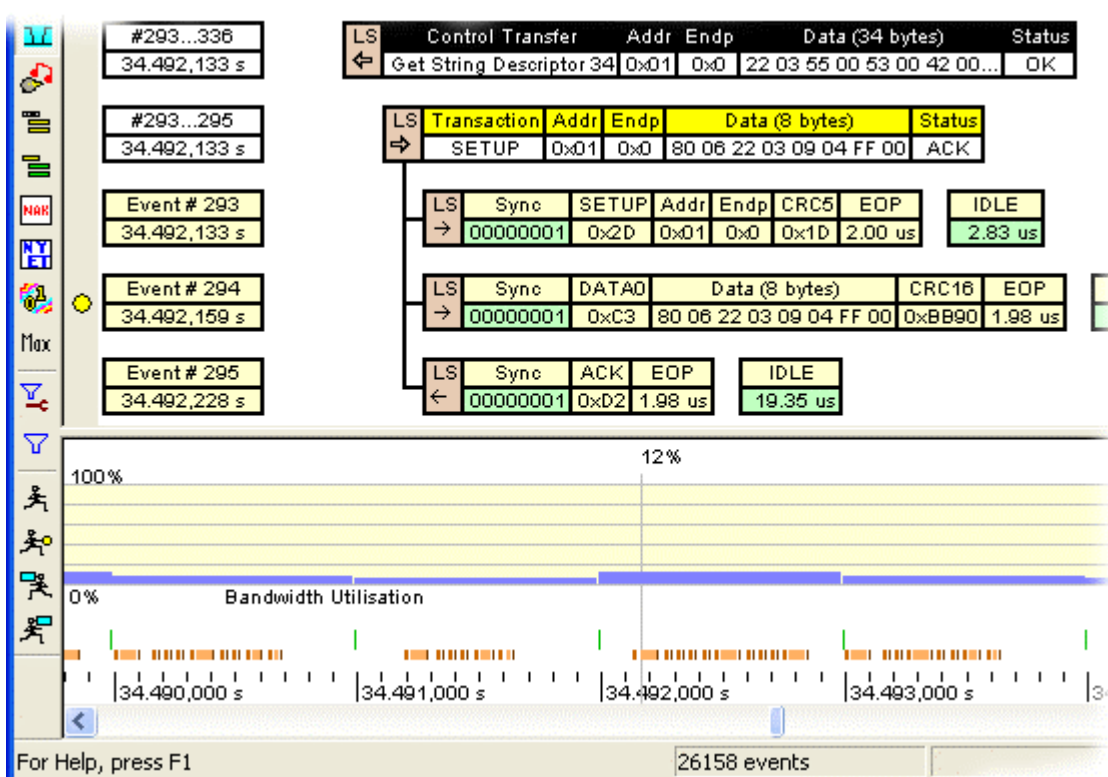
```


0000: 10 03 55 00 53 00    ..U.S.
0006: 42 00 20 00 48 00    B. .H.
000C: 75 00 62 00          u.b.

```



### 3.2.3 Timeline and Bandwidth Panes




Across the bottom of the window is the timeline and bandwidth utilisation pane. The bandwidth utilisation, or both displays can be hidden, using the  toolbar icon.

Initially, on opening a capture file, the timeline will span the complete duration of the capture.

The zoomable timeline pane shows actual bus activity down to packet level, allowing a rapid assessment of bus usage. The SOF packets, or Keep Alive events are shown slightly higher to show immediately where the frames begin and end.

The bandwidth utilisation indicates the proportion of data throughput compared with the maximum possible.

### 3.2.3.1 Zooming the Timeline/Bandwidth Pane

Zooming will always remain centered around the selected time. There are a number of ways to zoom this pane. To zoom without clicking in the pane, use the zoom buttons on the toolbar .

Clicking in the timeline pane enables zooming by means of mouse wheel, or by using cursor up or down keys. Note that if you click at the bottom of the pane, while the cursor looks like a hand, the selected time will not be affected.

### 3.2.3.2 Dragging the Timeline/Bandwidth Pane

While the cursor is at the the bottom of the pane, and looks like a hand, you can drag the view left or right by holding the left mouse button down. The left/right cursor keys have the same effect.

### 3.2.3.3 Selecting Events in the Timeline/Bandwidth Pane

If you click the left mouse button with the cursor further up the pane, and looking like a pointer, the nearest event to the left will be selected.

It is necessary to understand that the event selected will actually be the first event left of the selection point which has not been filtered out in the event pane. For example, if the event is a NAK packet, and NAKed transfers are not currently being displayed, then the first unfiltered transfer to the left will be selected. If SOFs are currently filtered then they will not be selected by this method.

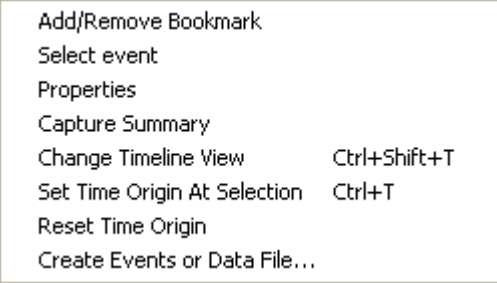
To be certain of identifying a particular packet, first click the Max button to the left of the event pane.

The selected event in the event pane will always be the same as in the timeline pane.

### 3.3 Pane Properties

If you right click in any of the panes, an appropriate properties menu will appear. This menu allows you to perform functions particularly relevant to the pane clicked on.

#### 3.3.1 Event Pane Properties



A screenshot of a context menu for the Event Pane. The menu items are: 'Add/Remove Bookmark', 'Select event', 'Properties', 'Capture Summary', 'Change Timeline View' (with a keyboard shortcut 'Ctrl+Shift+T'), 'Set Time Origin At Selection' (with a keyboard shortcut 'Ctrl+T'), 'Reset Time Origin', and 'Create Events or Data File...'. The menu is enclosed in a thin black border.

These are mostly self-explanatory. 'Capture Summary' is described below.


#### 3.3.2 Detail Pane Properties



A screenshot of a context menu for the Detail Pane. The menu contains a single item: 'Create Descriptor File...'. The menu is enclosed in a thin black border.

'Create Descriptor File' allows you to produce a text file output of any selected descriptor.

#### 3.3.3 Data Pane Properties



A screenshot of a context menu for the Data Pane. The menu items are: 'Create Current Data File...' and 'Select All' (with a keyboard shortcut 'Ctrl+A'). The menu is enclosed in a thin black border.

'Create Current Data File' allows you to export some or all of the data in the data pane in a variety of formats.

#### 3.3.4 Timeline Pane Properties



A screenshot of a context menu for the Timeline Pane. The menu items are: 'Change Timeline View' (with a keyboard shortcut 'Ctrl+Shift+T'), 'Set Time Origin At Selection' (with a keyboard shortcut 'Ctrl+T'), and 'Reset Time Origin'. The menu is enclosed in a thin black border.

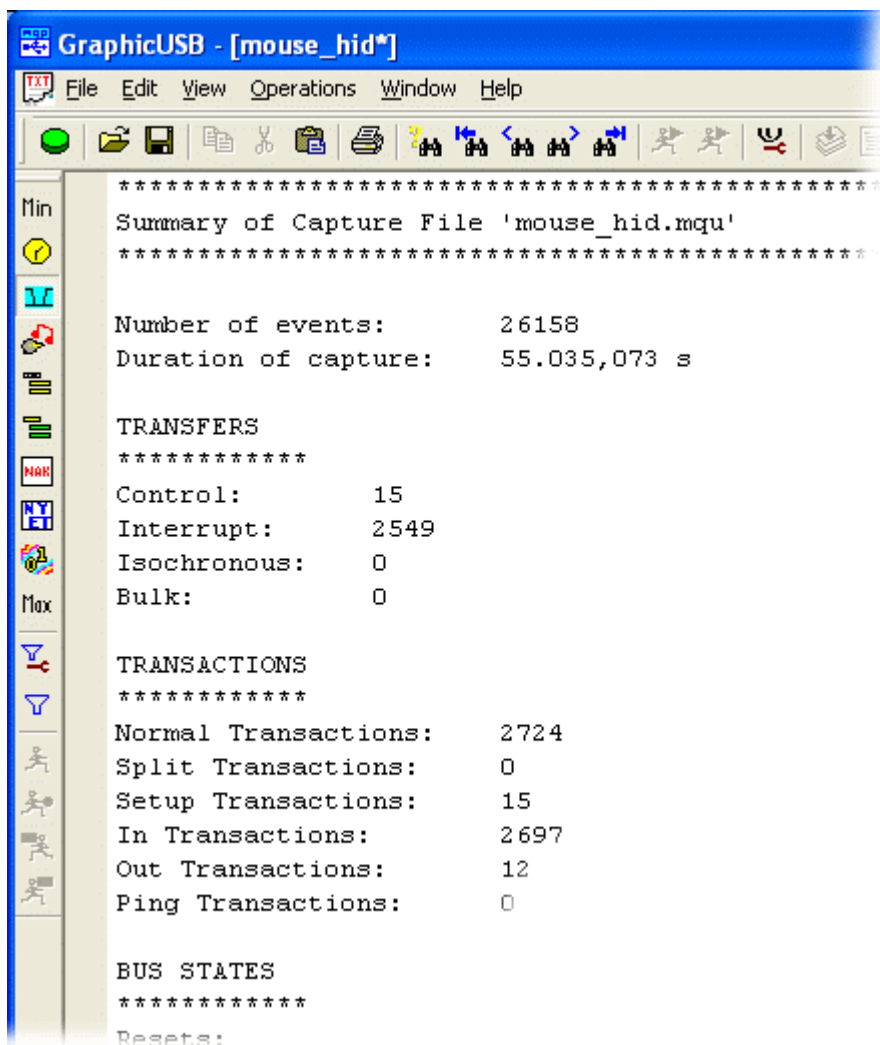
'Change Timeline View' (or Ctrl+Shift+T) allows you to show or hide parts of the timeline view.

### 3.4 Capture Summary

When viewing a capture, a summary of file statistics is available from menu item... View...Capture Summary.

The same summary is available by right clicking in the event pane.

The summary produces a text file giving statistics of each event type, of how many errors were detected, and of the devices encountered in the capture. This file may be saved or printed.



## 3.5 Toolbars

### 3.5.1 File Functions Toolbar



Start Capture



Open an existing Capture file



Save the currently active Capture document



Select Print Pane



Show/Hide Timeline/Bandwidth



Print the selected Pane



About GraphicUSB



Help on GraphicUSB



Search Settings



Find First



Find Previous



Find Next



Find Last



Goto Trigger Start



Goto Trigger Stop



Trigger Settings



















Compile (Generator Script)



Run (Generator Script)

### 3.5.2 View Filter Toolbar

	Show Top Level Events Only
	Show Start of Frame events
	Show bus events
	Show Chirps
	Show Transactions in Control Transfer
	Show Packets
	Show NAKed Transactions
	Show NYETed Transactions
	Show Spurious Data
	Show All Events
	Custom Filter Settings
	Custom Filter Enable
	Go to Event number
	Go to selected Event
	Go to Previous Bookmark
	Go to Next Bookmark

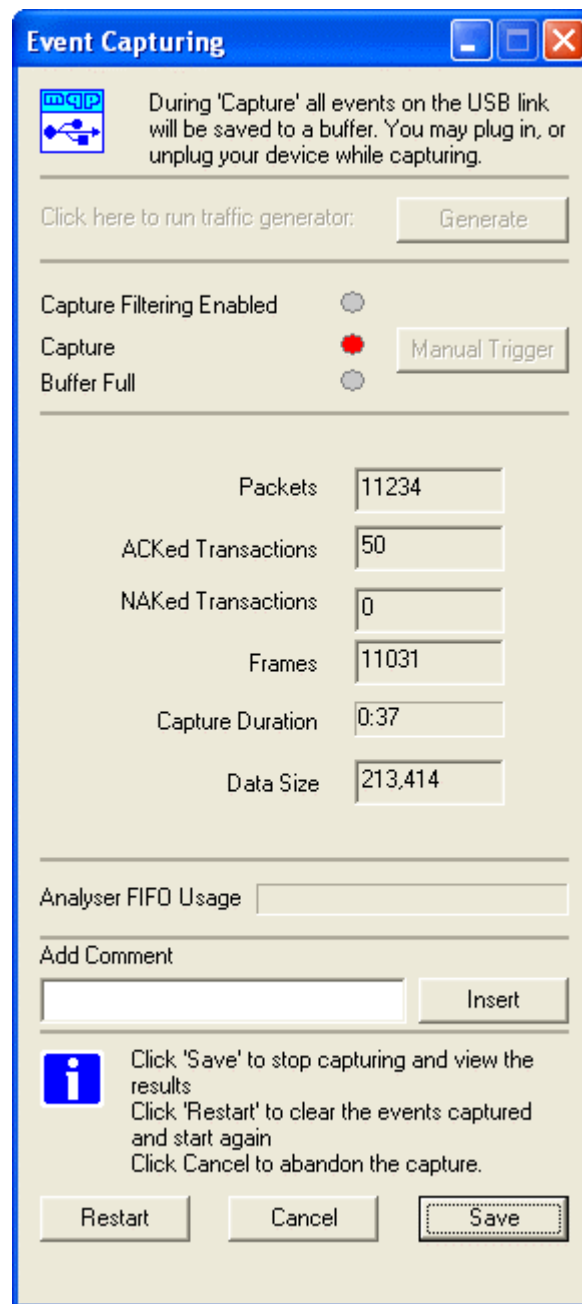
### **3.6 Capturing**

Capturing may be started by either clicking on the Capture button situated on the tool bar of GraphicUSB or by momentarily pressing the Start button on the front panel. During capturing an Event Capturing Window opens. This window displays the following information:

- The number of packets received
- The number of ACKed transactions
- The number of NAKed transactions
- The number of frames received
- The duration of the current capture
- The size of the captured data in bytes
- Analyser FIFO usage (USB480(+) and USB500 AG only)

At any time during the capture, devices may be plugged in or out. We recommend that this be done using the cable to the device, and not the cable to the host, to enable the most accurate interpretation of events. So the host cable should be connected to the analyser first, and then left connected during captures.





The screenshot shows the 'Event Capturing' window with a blue title bar and standard Windows window controls. Inside, there's a header section with the mqp logo and a descriptive text about capturing events on a USB link. Below this is a 'Generate' button. A section for 'Capture Filtering Enabled' has three radio buttons: 'Capture' (selected with a red dot), 'Buffer Full', and an unlabeled one. A 'Manual Trigger' button is next to the 'Capture' radio button. A list of statistics follows: Packets (11234), ACKed Transactions (50), NAKed Transactions (0), Frames (11031), Capture Duration (0:37), and Data Size (213,414). Below these is an 'Analyser FIFO Usage' field. An 'Add Comment' section includes a text input field and an 'Insert' button. An information icon (i) is followed by instructions: 'Click 'Save' to stop capturing and view the results', 'Click 'Restart' to clear the events captured and start again', and 'Click Cancel to abandon the capture.' At the bottom are three buttons: 'Restart', 'Cancel', and 'Save'.

**Event Capturing**

During 'Capture' all events on the USB link will be saved to a buffer. You may plug in, or unplug your device while capturing.

Click here to run traffic generator:

Capture Filtering Enabled ☐

Capture ☒

Buffer Full ☐

Packets

ACKed Transactions

NAKed Transactions

Frames

Capture Duration

Data Size

Analyser FIFO Usage

Add Comment

**i** Click 'Save' to stop capturing and view the results  
Click 'Restart' to clear the events captured and start again  
Click Cancel to abandon the capture.

To end capturing and display the data, either click on SAVE in the Capture window or momentarily press the STOP button on the front panel. Clicking RESTART will clear all the events so far and start again. Clicking CANCEL will abandon the capture.

### **3.7 Device Information Files**

A potential limitation of a USB analyser, is that correct USB protocol and class analysis depends on knowing basic information about the device which is only transferred on the bus during enumeration. This may require any captures which are performed to include the enumeration phase, perhaps by only plugging the device after the capture has started, or by using a pre-capture buffer (as is possible with the Packet-Master advanced triggering capability).

A capture started after enumeration has taken place lacks this information.

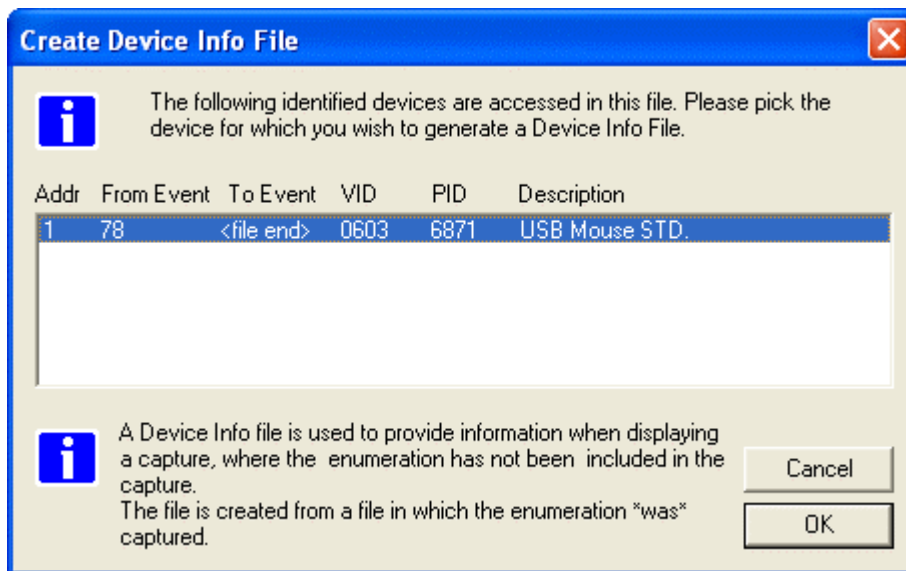
In order to overcome this problem, Packet-Master analysers have a special feature called 'Device Information' files, which essentially contain the enumeration data for a specific device.

GraphicUSB allows you to associate one or more particular device addresses within a capture, with specified 'Device Information' files, allowing the usual, better-informed analysis to take place.

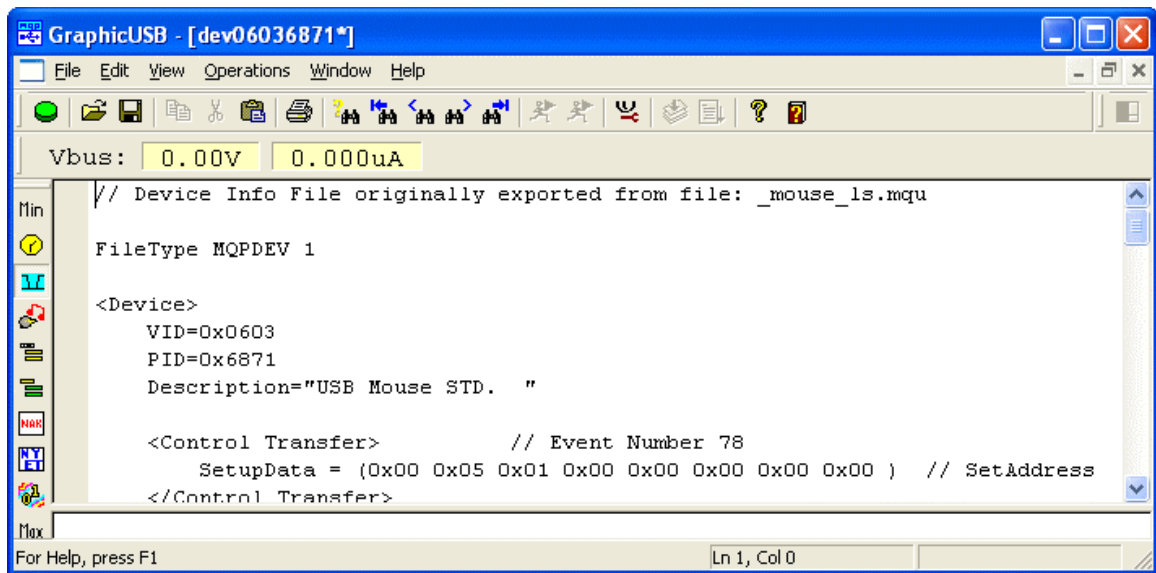
### 3.7.1 Creating a Device Information File

In order to create a device information file, start by performing a capture which includes the enumeration phase, by only plugging in the device after the capture has been started. Now, with the capture showing in the GraphicUSB window, select menu item:

File... Create Device Info File...



All the devices included in the capture are shown; if you had more than one device involved you need to choose the one from which you wish to create the Device Info file. Then click OK and the created file will appear in the GraphicUSB window.



Although the file can be edited, there is usually no need to do so. If it is necessary, then you need to observe the strict syntax rules described below, and to check the file, using the built-in validation function, by selecting menu item:

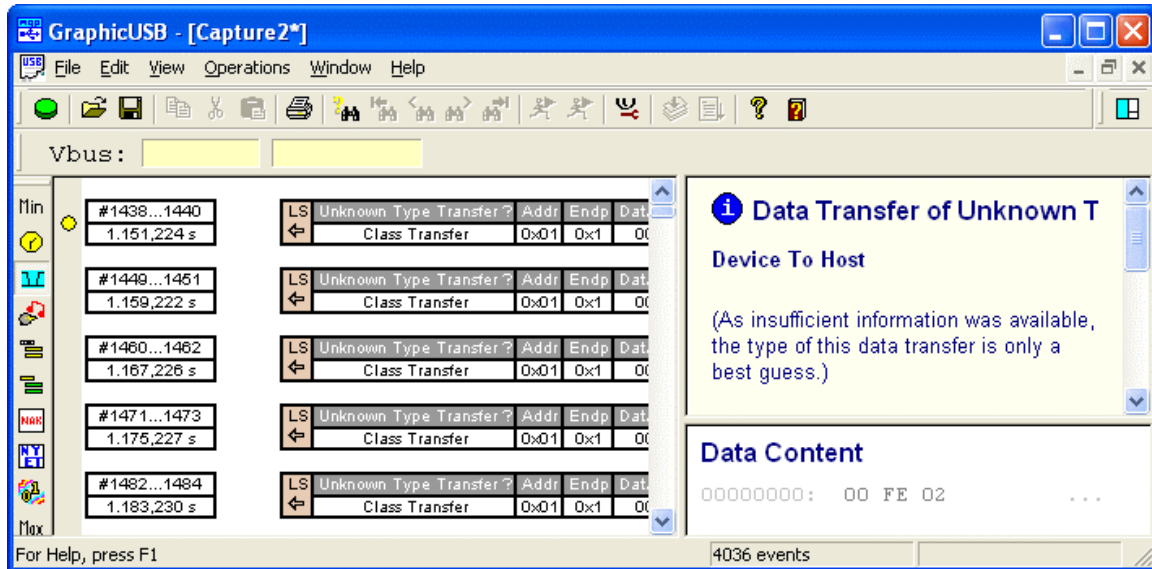
Operations... Validate Device Info File...

You now need to save this Device Info file. It will be given its own filename and special extension, which you must not change, and be saved into the folder used by the application for its data files (so you should accept the suggested file location).

In the example above, the filename will be dev06036871.mdev .

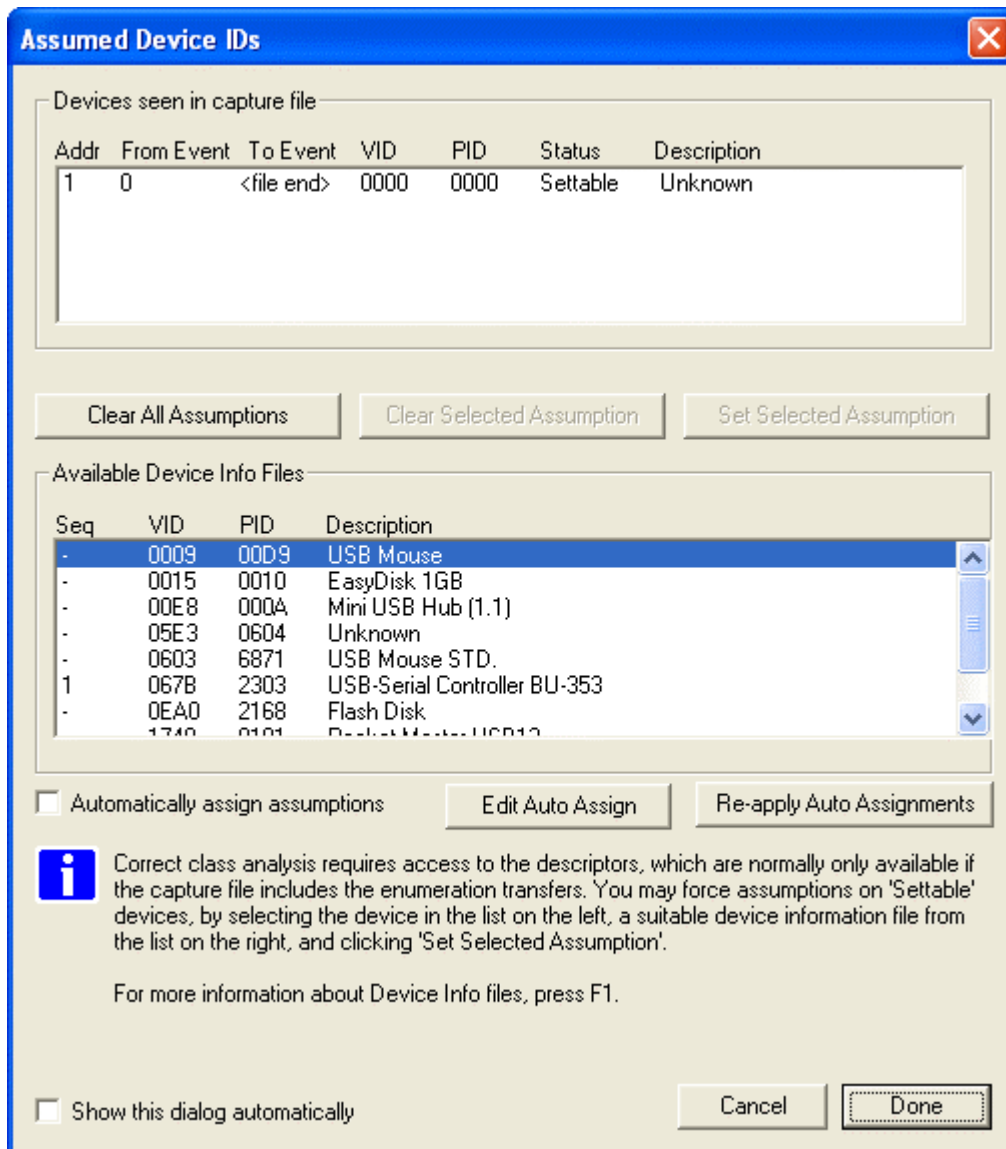
### 3.7.2 Using a Device Information File

Without a 'Device Info' file the displayed capture would appear like this:

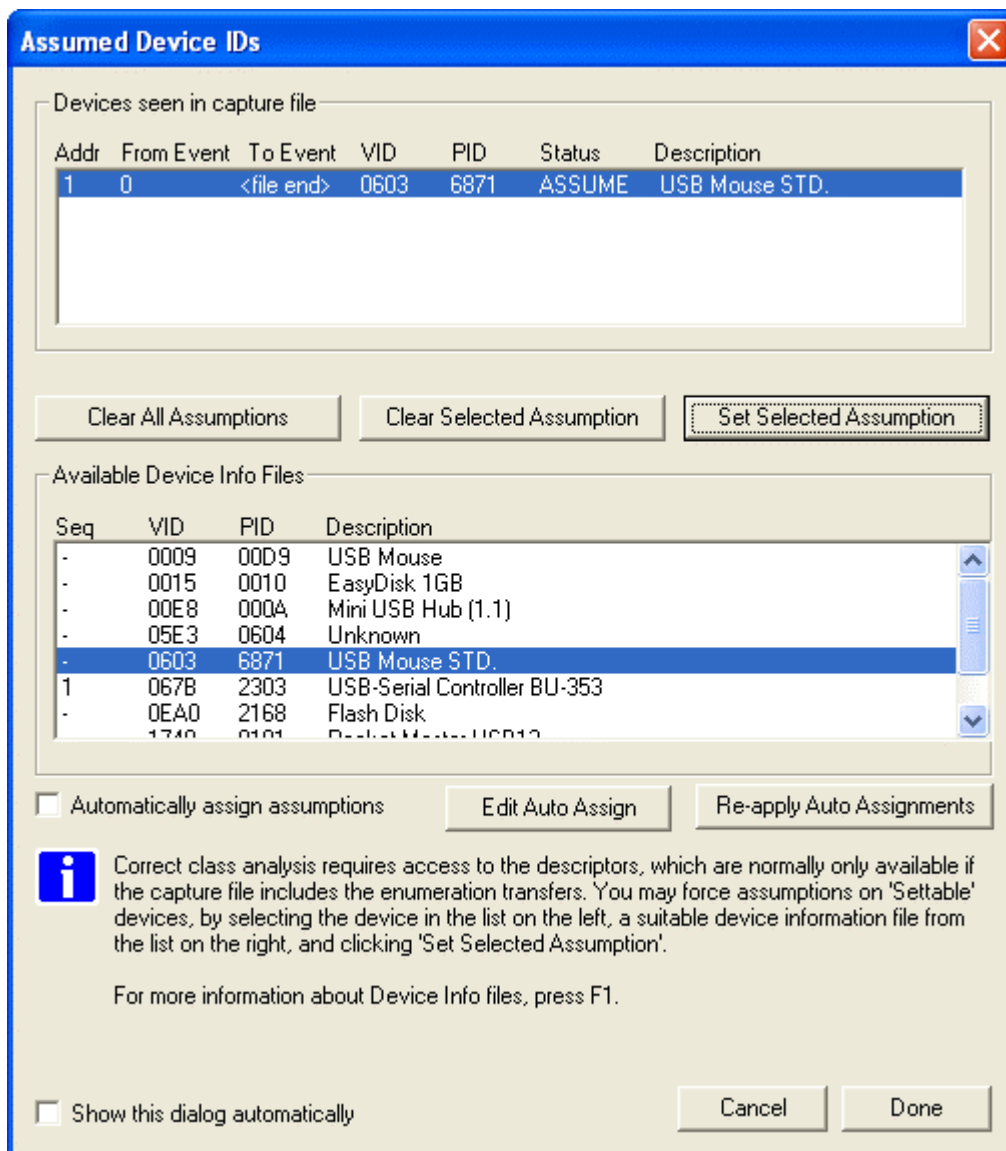


Whenever you perform a capture, which misses the enumeration phase, GraphicUSB will display a dialog, inviting you to associate a particular Device Info file with any device addresses in the capture.

Note that this dialog can be prevented from automatically appearing by unchecking the 'Show this dialog automatically' box. The dialog is always available in the Edit... Show Assumed Devices... menu item.

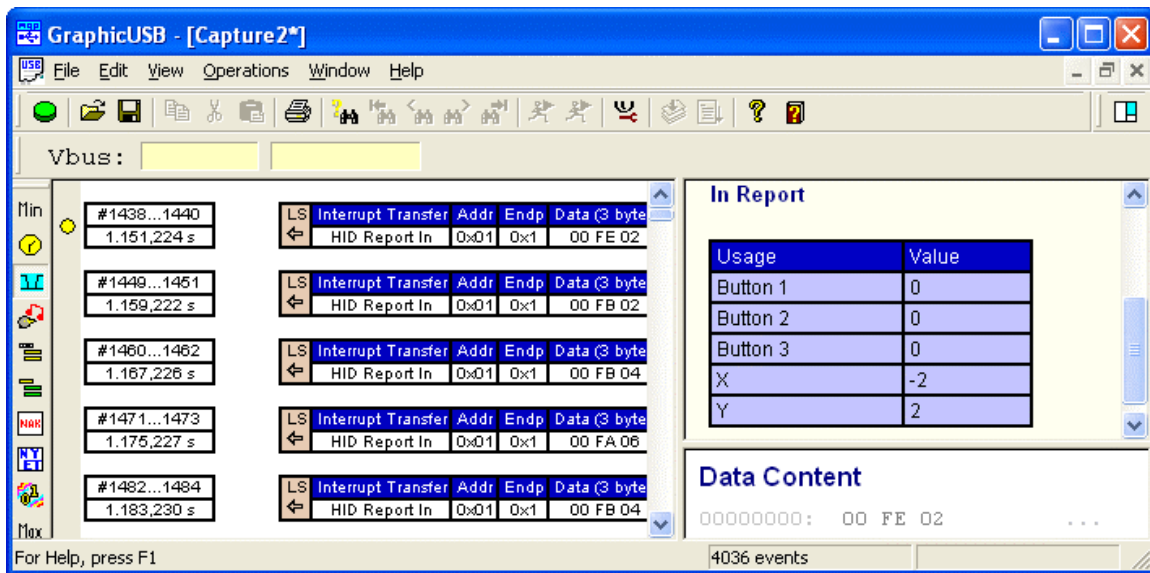


Click on the unknown device in the top list box. The lower list box will show you all the Device Info files you have previously created. Click on the appropriate one and then on the 'Set Selected Assumption' button.



The assumption will be registered in the top list box, as shown above. Click on 'Done' and the capture file will now reflect the assumption made.

You will not be allowed to apply an assumption to any device in the top list box, whose status shows 'Fixed'. These are devices with known enumeration information.

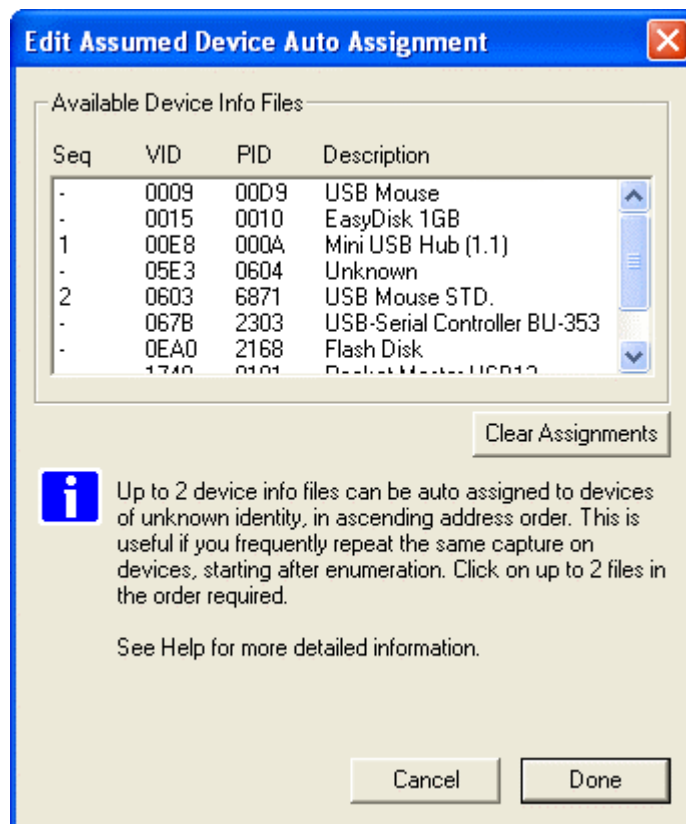


If you now save the capture file, the assumption will be stored as part of the file, so that the same assumption will automatically be made on the next opening of that file.

### 3.7.3 Automatic Assumption Assignment

If you are working on the same device or devices for some time, and are not capturing enumeration, then you may wish that the device assumption be made automatically. You may assign up to two Device Info files to be automatically assigned. Click on the 'Auto Edit Assign' button in the 'Assumed Device IDs' dialog, and the auto-assumption edit box appears.





Follow the instructions given, to select up to two Device Info files to assume.

Back in the 'Assumed Device IDs' dialog, make sure that the 'Automatically Assign Devices' checkbox is checked. The assumption will be applied to unknown devices in the capture file, in order of ascending address.

### 3.7.4 Device Information File Syntax Rules

#### 3.7.4.1 File Syntax

##### 3.7.4.1.1 *Comments*

A comment is introduced by the pair of characters '//'. Everything to the right on the same line is part of the comment and ignored.

##### 3.7.4.1.2 *Indentation*

The example file uses (tabbed) indentation to emphasise the structure of the syntax, but it is not necessary to do this.

##### 3.7.4.1.3 *Numbers*

Numerical values may be expressed in decimal, or in hexadecimal introduced by the prefix 0x. So 10 and 0x0a represent the same value.

##### 3.7.4.1.4 *Strings*

String values must be enclosed in double quote marks, e.g. "this is a string".

Any string must fit on one line, and may not include a line break.

##### 3.7.4.1.5 *FileType*

The first statement in the file must be FileType. E.g.

FileType MQPDEV 1

Type must be 'MQPDEV' and version must be 1, as shown.

##### 3.7.4.1.6 *<Device>*

*</Device>*

The whole file, other than the 'FileType', is the description of a device, and so must start with the '<Device>' tag, and end with '</Device>'.

##### 3.7.4.1.7 *VID=*

*PID=*

Following the '<Device>' tag, the next two lines must define the Vendor ID and the Product ID of the device.

#### 3.7.4.1.8 *Description =*

Following the PID= statement, the next line must be a string describing the device.

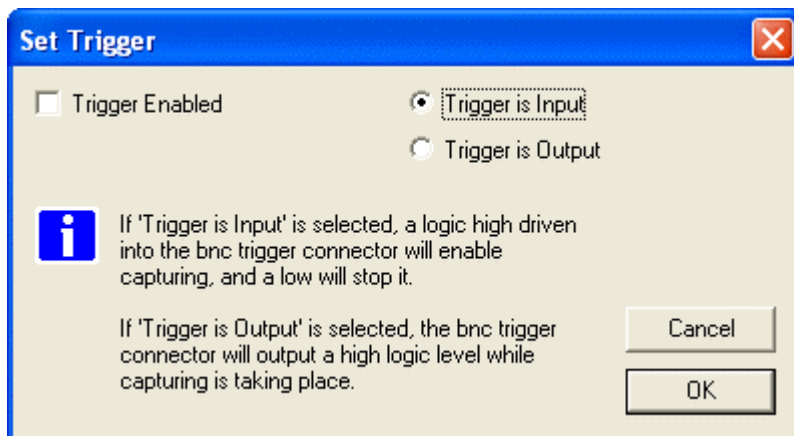
#### 3.7.4.1.9 *<Control Transfer>* *</Control Transfer >*

Each defined control transfer must be introduced by the '<Control Transfer>' tag, and ended with '</Control Transfer >'. Between the tags you should define the parameters of the request by specifying the following:

Parameter	Status	Value type	Purpose
SetupData = (...)	Mandatory	8 numbers from 0x00 - 0xff	Specifies the 8 SETUP packet data bytes
Data = (...)	Mandatory	Numbers from 0x00 - 0xff	Specifies the data bytes transferred in this control transfer.

### 3.8 Triggering (USB480)

Hardware triggering can be controlled from the Trigger Dialog. Select Menu..Operations..Trigger Setting..



#### 3.8.1 Trigger As Input

To use the trigger as an input, first select 'Trigger is Input', and check 'Trigger Enabled'. Connect a suitable logic signal to the BNC trigger connector. This signal will need to go to a logic high to start capturing and low to stop capturing.

To prime the trigger, use the usual Start Capture button. The capture dialog will be displayed, but capturing will not start. The 'primed' state is indicated by the capture led giving a slow flash. When the trigger input goes high, the capture led will show a steady on state, and capturing will start. When the trigger input goes low, the capture will be saved, and the capture led will go out.

#### 3.8.2 Trigger As Output

To use the trigger as an input, first select 'Trigger is Output', and check 'Trigger Enabled'. When capturing starts, the trigger output will go to a logic high level, and when capturing is stopped, it will go low again.

### 3.8.3 Trigger Not In Use

When not in use, ensure that the 'Trigger Enabled' box is unchecked, or you will not be able to start capturing.

## **3.9 Advanced Triggering (USB480+ and USB500 AG)**

### **3.9.1 Operation**

When triggering is enabled, pressing the Start Capture button on the Packet-Master analyser causes the Capture LED to start flashing to indicate that the trigger is primed. At this time Pre-Trigger capturing takes place into a circular buffer of a size specified by the user. From 2K to 4M bytes of event data may be stored prior to the trigger point.

When the trigger capture conditions are met, the Capture LED changes to 'steady on', and event capture continues, until either a stop capture condition is satisfied, or until the Stop Capture button is pressed.

The event which caused the Trigger Capture will be marked in the display by a green arrow, and the event which caused the Stop Capture will be marked in the display by a red arrow. Icons are displayed if the file contains either of these events, and clicking the icon takes you straight to the event in question.


If triggering is not enabled, then the Pre-Capture Buffer is not used. In this case pressing the Start Capture button on the Packet-Master analyser causes capturing to commence immediately, and the Capture LED to show 'steady on'.

The Trigger LED is lit (green) if triggering is enabled. In addition, its colour will be red if the BNC connector is being used as an input.

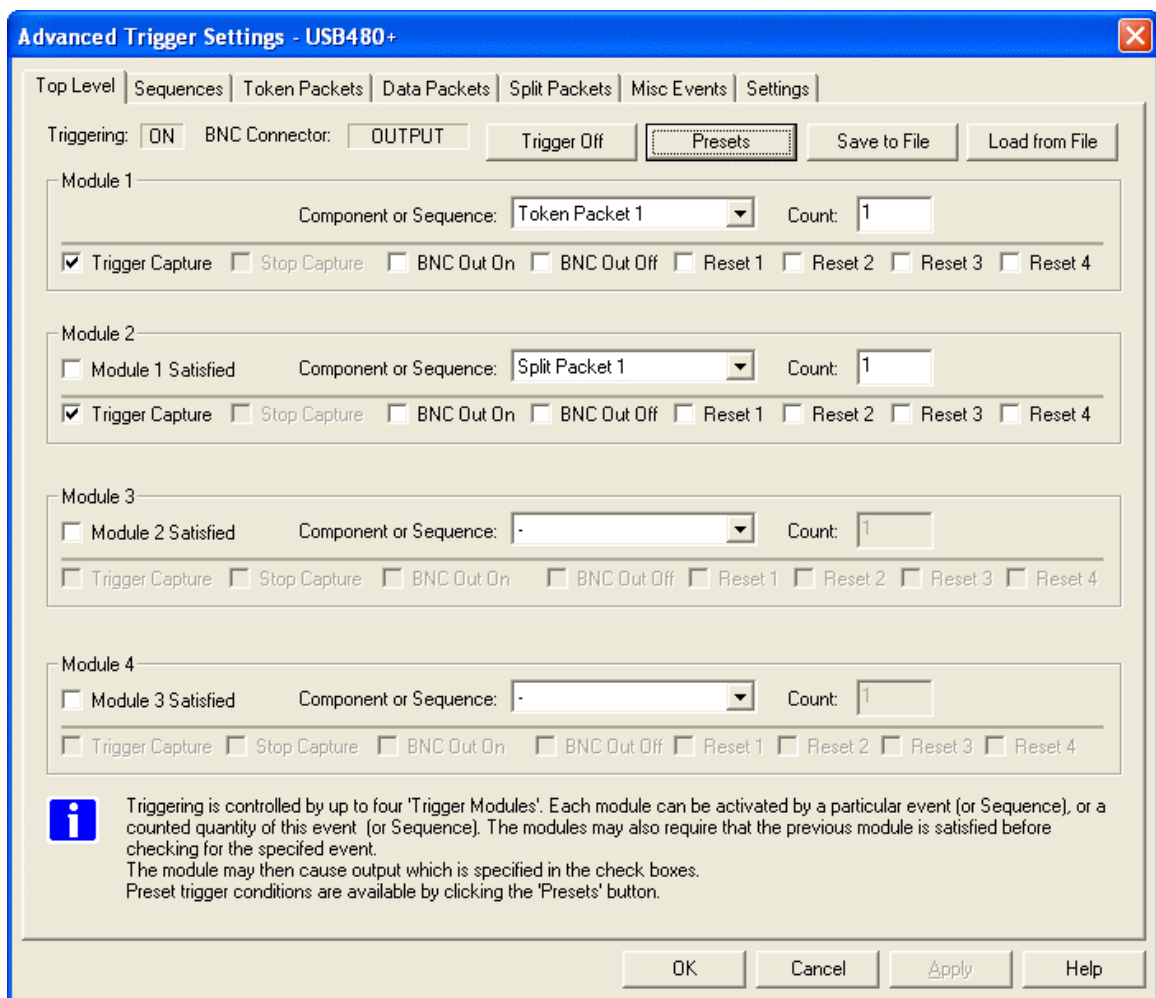
### 3.9.2 Events Available For Triggering

- Any Event
- Start Button (Manual)
- BNC Socket Input On Condition
- BNC Socket Input Off Condition
- Any of four specified Token Packets
- Any of four specified Data Packets
- Any of four specified Split Packets
- ACK Packet
- NAK Packet
- STALL Packet
- NYET Packet
- Not Handshake Packet (any event other than ACK, NAK, STALL or NYET packet)
- ERR Packet
- PRE Packet
- Specified SOF Packet
- Reserved PID Packet
- Bus Reset
- Suspend
- Resume
- HS Handshake OK
- HS Handshake Fail
- Chirp
- Keep Alive
- Data Line High
- Data Line Low
- Vbus On
- Vbus Off
- Specified Error Condition
- Any of four sequences of up to four of the above events occurring consecutively
- A counted quantity (up to of any of 65535) of any of the above
- A set of up to four of any of the above which occur in a particular order though not necessarily consecutively

### 3.9.3 Modules

The Packet-Master USB480+ is provided with an advanced trigger capability which is controlled from the Trigger Dialog. Select Menu..Operations..Trigger Setting.. or the toolbar icon .

The dialog which appears consists of seven tabbed pages. The main page 'Top Level' appears first.





Triggering is controlled by up to four 'Modules', each of which can be activated by a user defined event or sequence of events, or a counted number of such events or sequences.

Each module can result in a number of possible outcomes, such as triggering capture, stopping capture, producing an output on the BNC connector, or resetting the count on any of the four modules.

To allow more complex trigger conditions to be specified, if you check the checkbox 'Module x Satisfied', then the conditions of that module will not be looked for until the previous module (x) condition has been satisfied. So a set of conditions has to be satisfied in turn. Note that unlike 'sequences', the events in question do not need to be consecutive. A green arrow provides a visual indication of modules linked in this way.

Advanced Trigger Settings - USB480+

Top Level

Sequences

Token Packets

Data Packets

Split Packets

Misc Events

Settings

Triggering: ☒ ON

BNC Connector: ☐ OUTPUT

Trigger Off

Presets

Save to File

Load from File

Module 1

Component or Sequence: ACK Packet

Count: 1

☐ Trigger Capture

☐ Stop Capture

☐ BNC Out On

☐ BNC Out Off

☐ Reset 1

☐ Reset 2

☐ Reset 3

☐ Reset 4

Module 2

☒ Module 1 Satisfied

Component or Sequence: SDF Packet 1

Count: 1

☒ Trigger Capture

☐ Stop Capture

☐ BNC Out On

☐ BNC Out Off

☐ Reset 1

☐ Reset 2

☐ Reset 3

☐ Reset 4

Module 3

☐ Module 2 Satisfied

Component or Sequence: -

Count: 1

☐ Trigger Capture

☐ Stop Capture

☐ BNC Out On

☐ BNC Out Off

☐ Reset 1

☐ Reset 2

☐ Reset 3

☐ Reset 4

Module 4

☐ Module 3 Satisfied

Component or Sequence: -

Count: 1

☐ Trigger Capture

☐ Stop Capture

☐ BNC Out On

☐ BNC Out Off

☐ Reset 1

☐ Reset 2

☐ Reset 3

☐ Reset 4

i

Triggering is controlled by up to four 'Trigger Modules'. Each module can be activated by a particular event (or Sequence), or a counted quantity of this event (or Sequence). The modules may also require that the previous module is satisfied before checking for the specified event.

The module may then cause output which is specified in the check boxes.

Preset trigger conditions are available by clicking the 'Presets' button.

OK

Cancel

Apply

Help

User Manual Packet-Master-2.07

Copyright © 2006-2009 MQP Electronics Ltd

58

### 3.9.4 Sequences

The second tab in the dialog allows up to four 'Sequences' to be defined, which can then be used as inputs to any of the four Trigger Modules. A sequence is a set of events, which must occur consecutively to satisfy the condition.

In the example below, 'Sequence 1' has been defined as a token packet defined by 'Token Packet 1' (see below), followed by a data packet defined by 'Data Packet 1' (see below), followed by any event other than a Handshake packet. (This has the effect of the sequence being satisfied by an isochronous transaction.)

**Advanced Trigger Settings - USB480+**

Top Level | **Sequences** | Token Packets | Data Packets | Split Packets | Misc Events | Settings

**Sequence 1**

Event 1: Token Packet 1  
Event 2: Data Packet 1  
Event 3: Not Handshake Packet  
Event 4: -

**Sequence 2**

Event 1: -  
Event 2: -  
Event 3: -  
Event 4: -

**Sequence 3**

Event 1: -  
Event 2: -  
Event 3: -  
Event 4: -

**Sequence 4**

Event 1: -  
Event 2: -  
Event 3: -  
Event 4: -

Sequences are events which must occur one after another to satisfy the condition; for example, the packets of a transaction

OK Cancel Apply Help

Sequences are particularly useful for defining specific transactions. Each packet of a SETUP, IN, OUT, PING or SPLIT transaction can be defined in more or less detail.

Be aware that the packet PRE in low speed transactions counts as a separate event, so take care when specifying sequences for LS devices.

### 3.9.5 Token Packets

The third tab in the dialog allows up to four 'Token Packets' to be defined, which can then be used as inputs to any of the four Trigger Modules or any of the four sequences. A token packet is a SETUP, IN, OUT or PING packet.

In the example below 'Token Packet 1' has been set to respond to IN packets to device address 2 on any endpoint.

**Advanced Trigger Settings - USB480+**

Top Level | Sequences | **Token Packets** | Data Packets | Split Packets | Misc Events | Settings

**Token Packet 1**

☐ Any ☐ SETUP ☒ IN  
☐ OUT ☐ PING

Device Addr ☐ Any  h

Endpoint No. ☒ Any  h

**Token Packet 2**

☒ Any ☒ SETUP ☒ IN  
☒ OUT ☒ PING

Device Addr ☒ Any  h

Endpoint No. ☒ Any  h

**Token Packet 3**

☒ Any ☒ SETUP ☒ IN  
☒ OUT ☒ PING

Device Addr ☒ Any  h

Endpoint No. ☒ Any  h

**Token Packet 4**

☒ Any ☒ SETUP ☒ IN  
☒ OUT ☒ PING

Device Addr ☒ Any  h

Endpoint No. ☒ Any  h

The type, device address and endpoint number of a token packet may be specified. The device address is a 2-character hexadecimal number (max 7F), and the endpoint number is a single character hexadecimal number.

OK Cancel Apply Help

### 3.9.6 Data Packets

The fourth tab in the dialog allows up to four 'Data Packets' to be defined, which can then be used as inputs to any of the four Trigger Modules or any of the four sequences. A data packet is a DATA0, DATA1, DATA2 or MDATA packet.

In the example below 'Data Packet 1' has been set to respond to any data packet containing 13 bytes and having the bytes 0x80, 0x34 and 0x02 sequentially within its data field. The mask settings can be used to ignore specific bits within a byte if required.

The image shows a screenshot of the 'Advanced Trigger Settings - USB480+' dialog box. The 'Data Packets' tab is selected. It contains four sections for 'Data Packet 1' through 'Data Packet 4'. Each section has a 'Clear' button, checkboxes for 'Any', 'DATA0', 'DATA1', 'DATA2', and 'MDATA', a 'Match' row of 8 byte boxes, a 'Mask' row of 8 bit boxes, and a 'Size' section with an 'Any' checkbox and a text box for the number of bytes.

**Data Packet 1:**

- Clear
- ☒ Any ☒ DATA0 ☒ DATA1 ☒ DATA2 ☒ MDATA
- Match: 80 34 02 00 00 00 00 00
- Mask: FF FF FF 00 00 00 00 00
- Size: ☐ Any 13 bytes

**Data Packet 2:**

- Clear
- ☒ Any ☒ DATA0 ☒ DATA1 ☒ DATA2 ☒ MDATA
- Match: 66 00 00 00 00 00 00 00
- Mask: FF 00 00 00 00 00 00 00
- Size: ☒ Any [ ] bytes

**Data Packet 3:**

- Clear
- ☒ Any ☒ DATA0 ☒ DATA1 ☒ DATA2 ☒ MDATA
- Match: 00 00 00 00 00 00 00 00
- Mask: 00 00 00 00 00 00 00 00
- Size: ☒ Any [ ] bytes

**Data Packet 4:**

- Clear
- ☒ Any ☒ DATA0 ☒ DATA1 ☒ DATA2 ☒ MDATA
- Match: 00 00 00 00 00 00 00 00
- Mask: 00 00 00 00 00 00 00 00
- Size: ☒ Any [ ] bytes

**Information:** Each Data Packet component can specify a pattern of up to 8 bytes to match. The Mask allows individual bits to be excluded from the test. You can also specify the length of the data field which will result in a match.

Buttons: OK, Cancel, Apply, Help

### 3.9.7 Split Packets

The fifth tab in the dialog allows up to four 'Split Packets' to be defined, which can then be used as inputs to any of the four Trigger Modules or any of the four sequences.

In the example below 'Split Packet 1' has been set to respond to any Start Split packet intended for device address 3.

**Advanced Trigger Settings - USB480+**

Top Level | Sequences | Token Packets | Data Packets | **Split Packets** | Misc Events | Settings

**Split Packet 1**

Device Addr	<input type="checkbox"/> Any	3	h
Port	<input checked="" type="checkbox"/> Any		h
SC	<input type="checkbox"/> Any	0	
S	<input checked="" type="checkbox"/> Any		
E	<input checked="" type="checkbox"/> Any		
ET	<input checked="" type="checkbox"/> Any		

**Split Packet 2**

Device Addr	<input checked="" type="checkbox"/> Any		h
Port	<input checked="" type="checkbox"/> Any		h
SC	<input checked="" type="checkbox"/> Any		
S	<input checked="" type="checkbox"/> Any		
E	<input checked="" type="checkbox"/> Any		
ET	<input checked="" type="checkbox"/> Any		

**Split Packet 3**

Device Addr	<input checked="" type="checkbox"/> Any		h
Port	<input checked="" type="checkbox"/> Any		h
SC	<input checked="" type="checkbox"/> Any		
S	<input checked="" type="checkbox"/> Any		
E	<input checked="" type="checkbox"/> Any		
ET	<input checked="" type="checkbox"/> Any		

**Split Packet 4**

Device Addr	<input checked="" type="checkbox"/> Any		h
Port	<input checked="" type="checkbox"/> Any		h
SC	<input checked="" type="checkbox"/> Any		
S	<input checked="" type="checkbox"/> Any		
E	<input checked="" type="checkbox"/> Any		
ET	<input checked="" type="checkbox"/> Any		

**i** The fields of the split packet may be specified by entering values into the appropriate boxes. The device address and port must be entered in hexadecimal format.

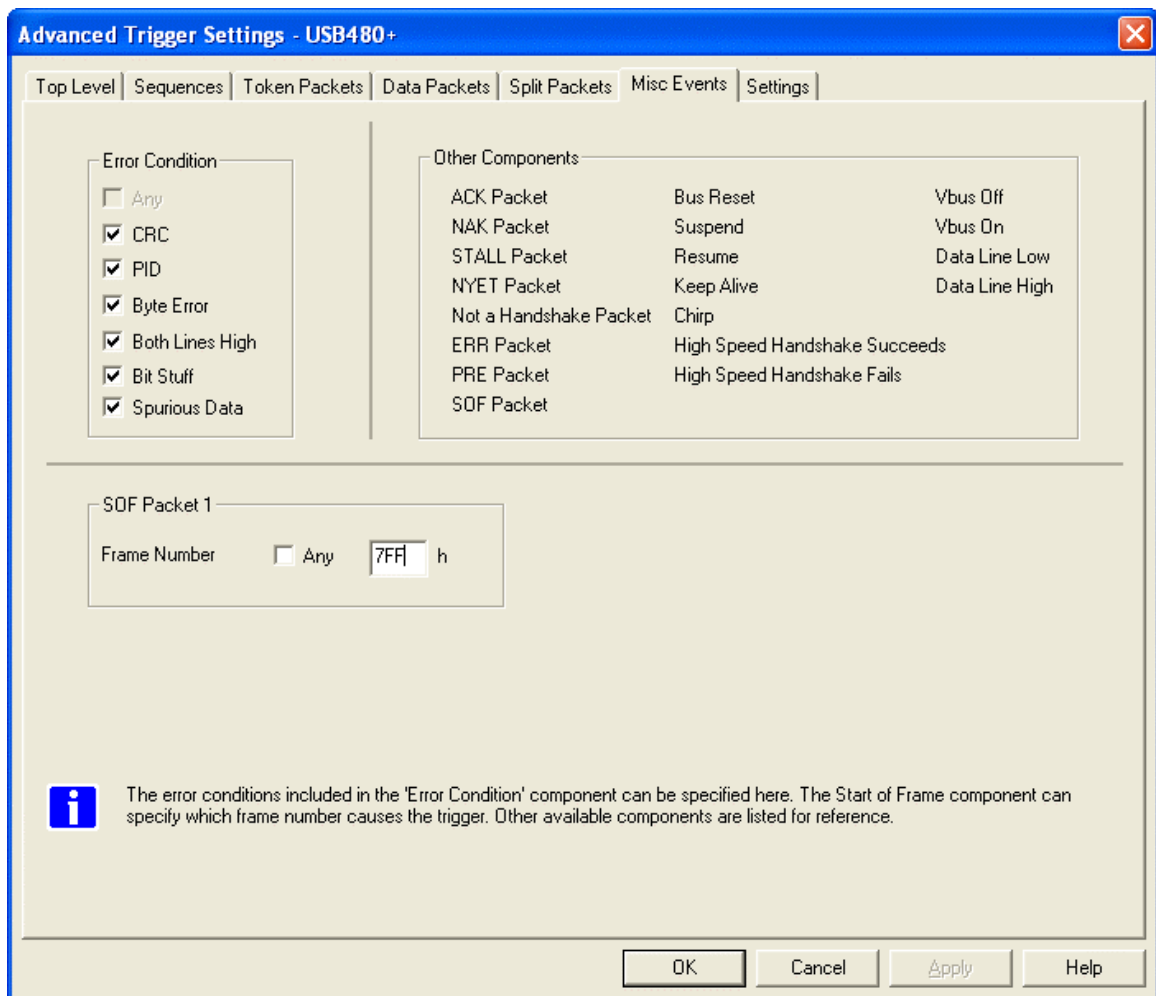
OK Cancel Apply Help

### 3.9.8 Miscellaneous Events

The fifth tab in the dialog allows an 'Error Condition' and a 'Start of Frame Packet' to be defined, which can then be used as inputs to any of the four Trigger Modules or any of the four sequences.

In the example below 'Error Condition' has been set to respond to any any of the detectable errors, and 'SOF Packet 1' has been set to respond to a SOF packet having the frame number 0x7FF.

Other Components available as inputs to any of the four Trigger Modules or any of the four sequences are also listed for reference.



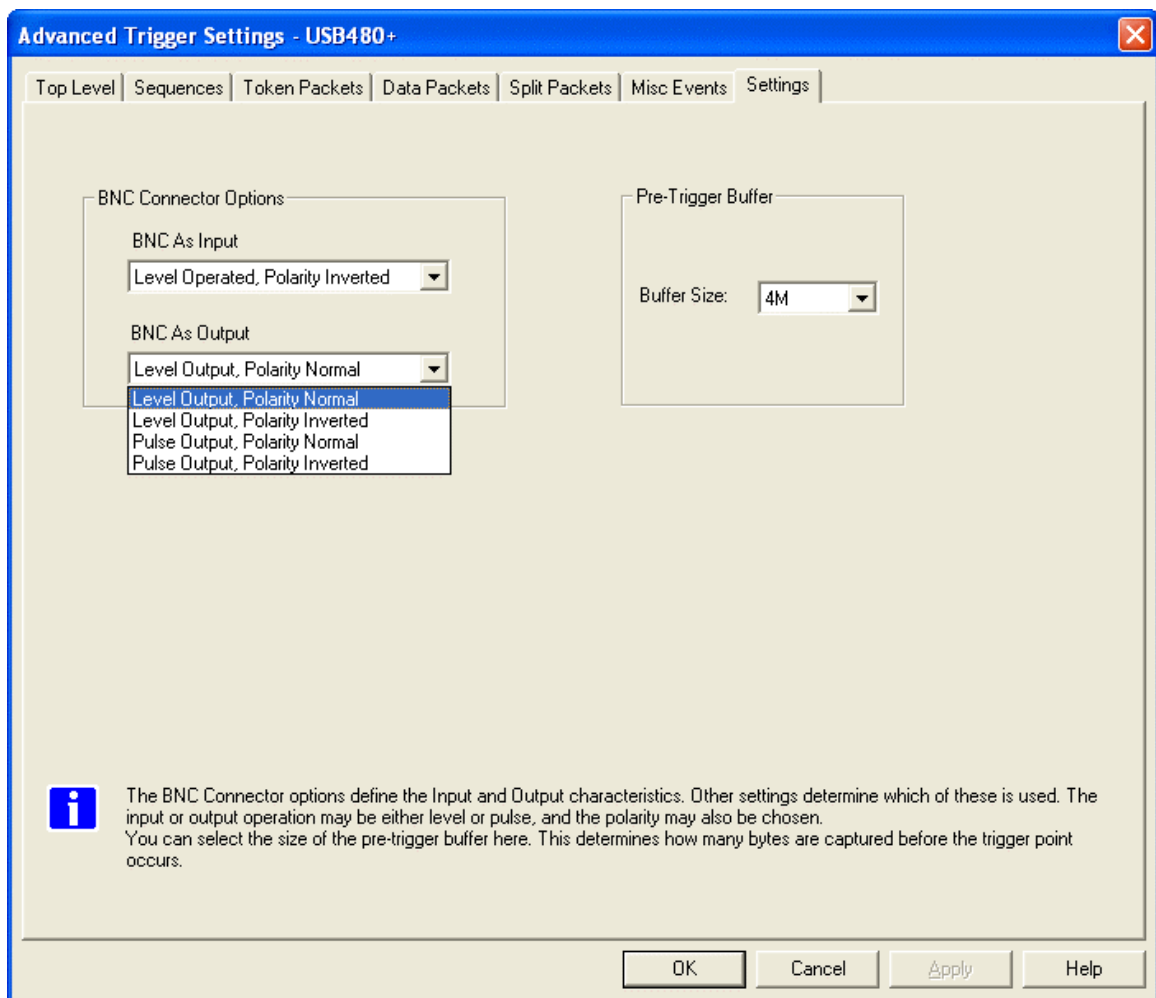


### 3.9.9 Other Settings

The sixth tab in the dialog allows The BNC Connector options and the size of the Pre-Trigger Buffer to be defined.

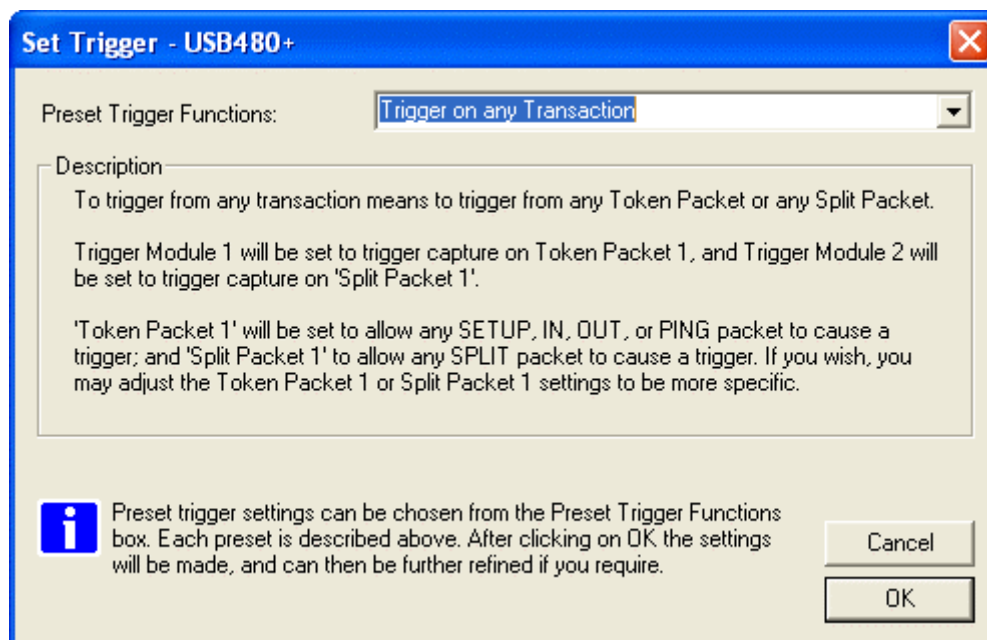
In the example below the available 'BNC as Output' settings are being displayed. 'BNC as Input' has similar options.

The Pre-Trigger Buffer captures a number of bytes before the trigger condition is satisfied, as well as the bytes captured after the trigger point. The actual size of this buffer can be defined here.



### 3.9.10 Preset Trigger Settings

In order to allow fast trigger setup, a number of presets are provided to cover common triggering requirements, to act as a starting point for more complex settings, or simply to demonstrate how typical triggering requirements can be set up.



Available preset functions include:

- Triggering OFF
- Manual Triggering
- Trigger on any Transaction
- Trigger on any Data Packet
- Trigger on Bus Reset
- Trigger on Preamble with Setup Transaction
- Trigger on Isochronous In Transaction
- Trigger on Isochronous Out Transaction
- Trigger on In or Out Transaction



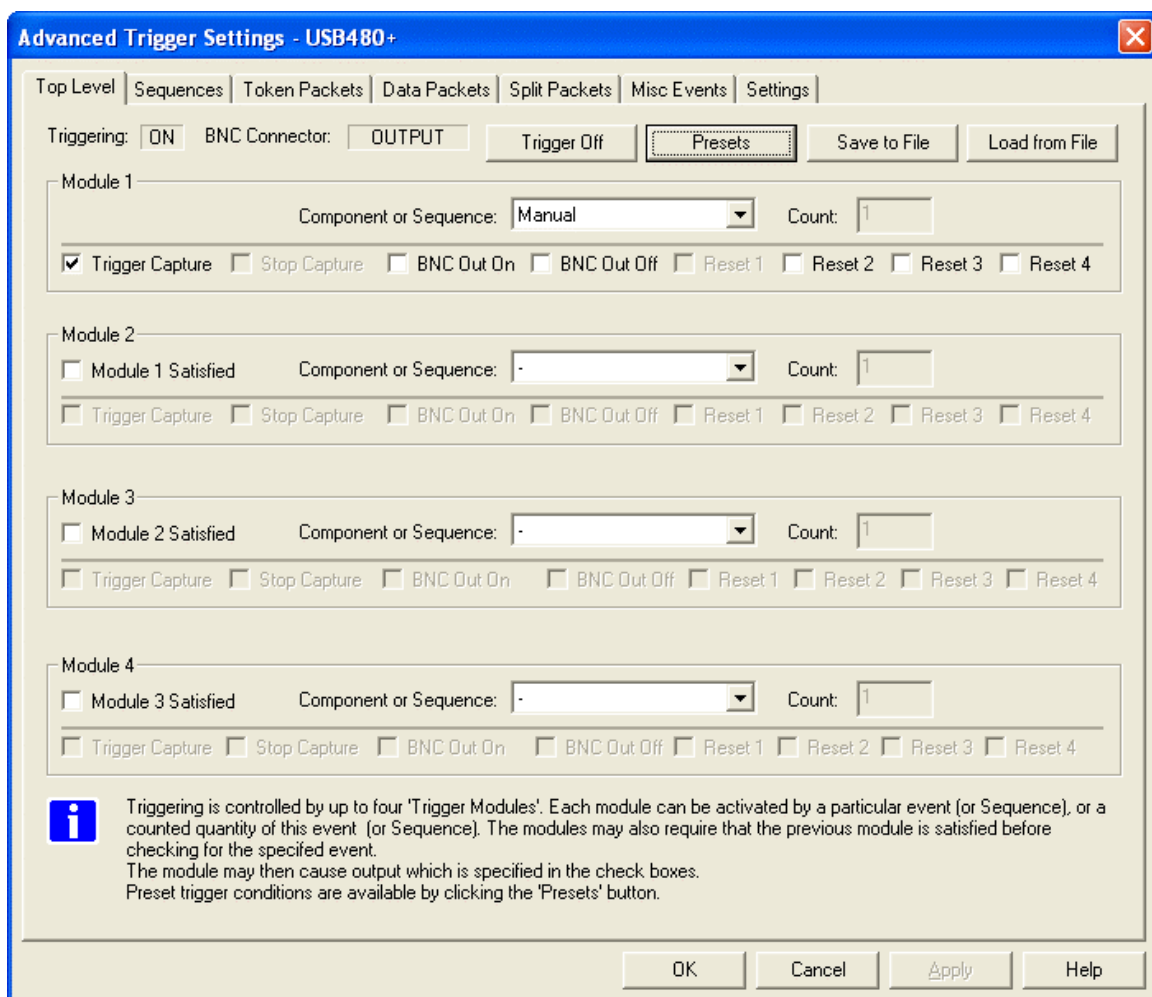
- Trigger on Ping Transaction
- Trigger on Split Setup Transaction
- Trigger on Split Bulk In or Out Transaction
- Trigger on Split Interrupt In or Out Transaction
- Trigger on Split Isochronous In or Out Transaction
- Trigger on Specified Data Pattern
- Trigger on Chirp
- Trigger on Suspend or Resume
- Trigger on Error Condition

### 3.9.11 Manual Triggering

Manual Triggering requires a word of explanation. In this mode, pressing the Packet-Master Start Button will prime the trigger ready for the trigger event, which in this case is a further press of the Start Button.

The advantage of this is that the Pre-Trigger Buffer is used from the moment of the first button press. You may be trying to capture events, which occur just before an observable condition occurs on your device. So when you trigger the capture with the second button press, up to 4 million bytes of event data, which occurred before the trigger, may be captured.

Manual triggering can be set by of the Preset Triggering options.



### 3.9.12 BNC Connector

The BNC connector can be used as an input to a Trigger Module, to trigger or stop capture. It can also be used as an output from a Trigger Module to trigger other test equipment. Obviously it cannot be used for both input and output simultaneously, and you will not be allowed to set up such a situation.

Both input and output can be set for level operation, or pulsed operation, and the polarity can be positive or negative.

Level based operation is straightforward and unambiguous.

For pulsed output operation, the pulse length is approx 83 ns, and there is, of course, no way to distinguish 'BNC Out On' from 'BNC Out Off'.

For pulsed input operation the leading edge of the pulse is used, and the pulse must last for at least 50ns. The first pulse will be taken to be a BNC Socket Input On, and the second to be BNC Socket Input Off.

### **3.9.13 Trigger Off**

Clicking this button on the 'Top Level' tab, or selecting it from the Preset options, results in the Trigger being switched off. In this mode, pressing the Packet-Master Start button will result in capturing taking place immediately. In this mode the Pre-Trigger Buffer is not operational.


### **3.9.14 Save Settings**

Clicking this button on the 'Top Level' tab will save all the current settings (as though you had pressed the OK button) and then allow you to save the trigger settings to a file of your choice.

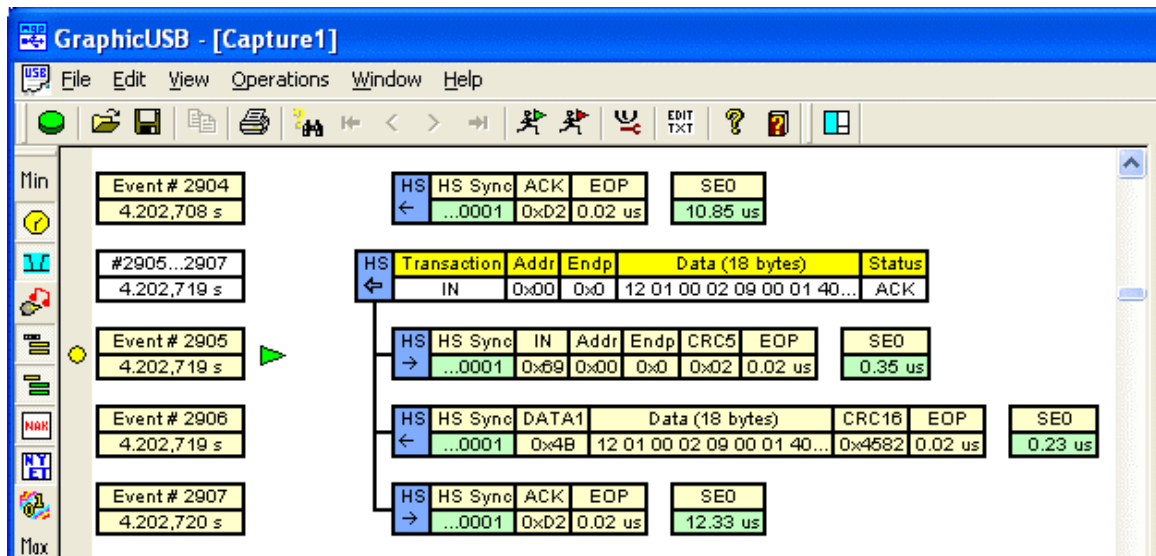
### **3.9.15 Load Settings**

Clicking this button on the 'Top Level' tab allows you to select a file of previously saved settings to configure the triggering settings.

### **3.9.16 Finding the Trigger Points**

If the capture file contains a Trigger Start or Trigger Stop event the appropriate icon on the toolbar will be enabled.  (Green for 'Go To Trigger Start Event' and red for 'Go To Trigger Stop Event').

Clicking the icon will take you to the event in question, which is marked with an arrow of the appropriate colour.



### 3.10 Display Filters

Toolbar buttons allow the filtering out of events that you do not wish to display. The following filters are available:



#### Show SOF

Start of Frame packets on high and full speed devices and Keep Alive events on low speed devices come at approximately one millisecond intervals (125 us for high speed). Clicking this tool bar button removes/shows these events.



#### Show Bus States

Clicking the Bus States button removes/shows the following events:

- Plugged in
- Unplugged
- Reset
- Suspend
- Resume



#### Show Chirps

Clicking the Show Chirps button removes/shows chirp events within a High Speed Detection Handshake. Chirps are only used on high speed links.





## Show Transactions

A Control Transfer contains a number of transactions starting with a SETUP. Clicking this button removes/shows the transactions within a Control Transfer. The example below shows the effect of filtering out the transactions.

#81...87 5.126,195 s	FS Control Transfer Addr Endp Data (0 bytes) Status ⇒ Set Address (0x01) 0x00 0x0 OK
#81...83 5.126,195 s	FS Transaction Addr Endp Data (8 bytes) Status ⇒ SETUP 0x00 0x0 00 05 01 00 00 00 00 00 ACK
#85...87 5.127,195 s	FS Transaction Addr Endp Data (0 bytes) Status ⇐ IN 0x00 0x0 ACK
#127...145 5.166,196 s	FS Control Transfer Addr Endp Data (18 bytes) Status ⇐ Get Device Descriptor 0x01 0x0 12 01 00 02 00 00 00 08... OK
#127...129 5.166,196 s	FS Transaction Addr Endp Data (8 bytes) Status ⇒ SETUP 0x01 0x0 80 06 00 01 00 00 12 00 ACK

## Show Transactions

#81...87 5.126,195 s	FS Control Transfer Addr Endp Data (0 bytes) Status ⇒ Set Address (0x01) 0x00 0x0 OK
#127...145 5.166,196 s	FS Control Transfer Addr Endp Data (18 bytes) Status ⇐ Get Device Descriptor 0x01 0x0 12 01 00 02 00 00 00 08... OK

## Hide Transactions

If transactions within control transfers have been filtered out, then double clicking on a particular control transfer will reveal the transactions within it, as shown below.

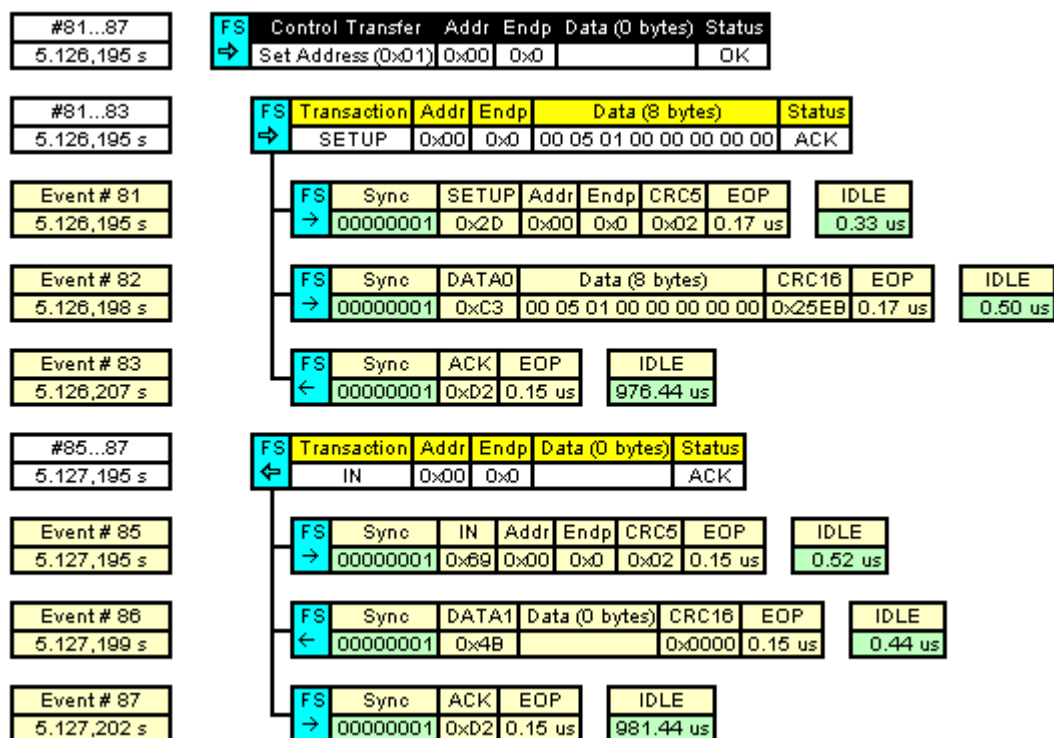
#47...58 5.086,194 s	FS ←	Control Transfer	Addr	Endp	Data (8 bytes)	Status
		Get Device Descriptor	0x00	0x0	12 01 00 02 00 00 00 08	OK
#81...87 5.126,195 s	FS →	Control Transfer	Addr	Endp	Data (0 bytes)	Status
		Set Address (0x01)	0x00	0x0		OK
#81...83 5.126,195 s	FS →	Transaction	Addr	Endp	Data (8 bytes)	Status
		SETUP	0x00	0x0	00 05 01 00 00 00 00 00	ACK
#85...87 5.127,195 s	FS ←	Transaction	Addr	Endp	Data (0 bytes)	Status
		IN	0x00	0x0		ACK
#127...145 5.166,196 s	FS ←	Control Transfer	Addr	Endp	Data (18 bytes)	Status
		Get Device Descriptor	0x01	0x0	12 01 00 02 00 00 00 08...	OK
#148...162 5.172,196 s	FS ←	Control Transfer	Addr	Endp	Data (9 bytes)	Status
		Get Configuration Descriptor	0x01	0x0	09 02 22 00 01 01 00 A0...	OK

Show Selected Transactions

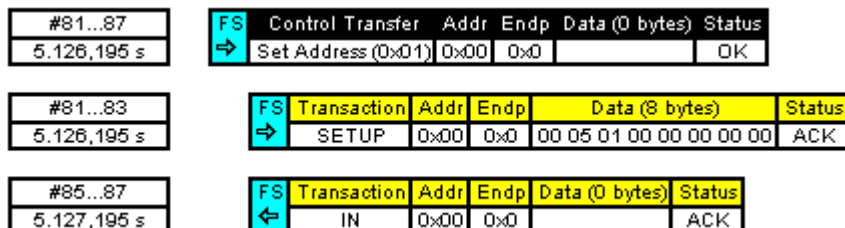


## Show Packets

A transaction contains a number of packets. Clicking this button removes/shows the packets within the transactions.



## Show Packets



## Hide Packets

If packets have been filtered out, then double clicking on a particular transaction will reveal the packets within it, as shown below.

#127...145 5.166,196 s	FS Control Transfer Addr Endp Data (18 bytes) Status Get Device Descriptor 0x01 0x0 12 01 00 02 00 00 00 08... OK
#127...129 5.166,196 s	FS Transaction Addr Endp Data (8 bytes) Status SETUP 0x01 0x0 80 06 00 01 00 00 12 00 ACK
#131...133 5.167,195 s	FS Transaction Addr Endp Data (8 bytes) Status IN 0x01 0x0 12 01 00 02 00 00 00 08 ACK
Event # 131 5.167,195 s	FS Sync IN Addr Endp CRC5 EOP IDLE 00000001 0x69 0x01 0x0 0x1D 0.15 us 0.54 us
Event # 132 5.167,199 s	FS Sync DATA1 Data (8 bytes) CRC16 EOP IDLE 00000001 0x4B 12 01 00 02 00 00 00 08 0xE757 0.15 us 0.42 us
Event # 133 5.167,207 s	FS Sync ACK EOP IDLE 00000001 0xD2 0.15 us 980.04 us
#135...137 5.168,195 s	FS Transaction Addr Endp Data (8 bytes) Status IN 0x01 0x0 D8 04 00 00 01 00 01 02 ACK
#139...141 5.169,195 s	FS Transaction Addr Endp Data (2 bytes) Status IN 0x01 0x0 00 01 ACK
#143...145 5.170,195 s	FS Transaction Addr Endp Data (0 bytes) Status OUT 0x01 0x0 ACK

Show Selected Packets



### Show NAKs

Clicking this button removes/shows any NAKed transactions. This differs from filtering NAKs during capture where NAKed control transactions will always be included.



### Show NYETs

Clicking this button removes/shows any NYETed transactions. Note that successful transactions responded to by NYET are not hidden. NYET is only used on high speed links.



### Show Spurious Data

Clicking this button removes/shows any spurious data packets. Such packets cannot be determined to be valid members of a transaction and may be caused by inadequate cabling, or result from data sent by a high-speed host as the device is being unplugged. The first in any sequence of spurious packets is shown automatically but the ones following may be hidden.

A small, light gray rectangular button with the word 'Min' in black text.

### Show Top level Events Only

This is a quick way to view a summary of the sequence of events. Clicking this button turns off the following buttons in one click:

- Show SOF
- Show Bus States
- Show Chirps
- Show Transactions in Control Transfers
- Show Packets
- Show NAKs
- Show NYETs
- Show Spurious Data

It turns on:

- Show Bus States

It has no effect on the Custom Filter.

A small, light gray rectangular button with the word 'Max' in black text.

### Show All Events

This is a quick way to view every one of the sequence of events. Clicking this button turns on the following buttons in one click:

- Show SOF
- Show Bus States
- Show Transactions in Control Transfers
- Show Packets
- Show NAKs
- Show NYETs

It has no effect on the Show Chirps button and also no effect on the Custom Filter.

### 3.11 Custom Filter

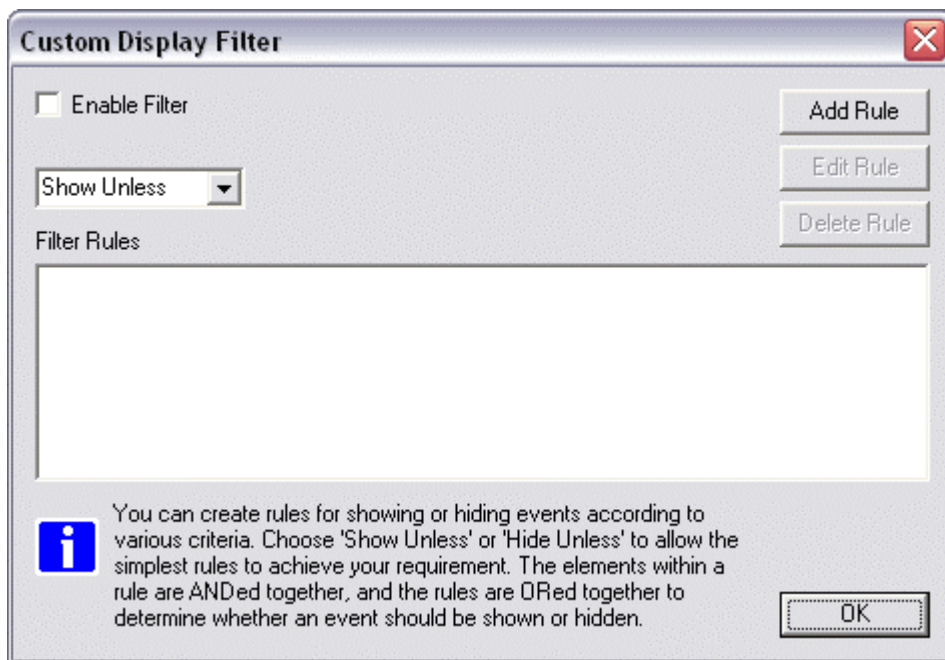
It is sometimes required to filter out transactions to particular addresses or endpoints, in order to simplify the display of events. For example, a capture may contain spurious part transactions intended for an upstream hub, which may perhaps be misinterpreted by the analyser. Hiding them allows you to concentrate on the important transactions.

#### 3.11.1 Custom Filter Settings



##### Custom Filter Settings

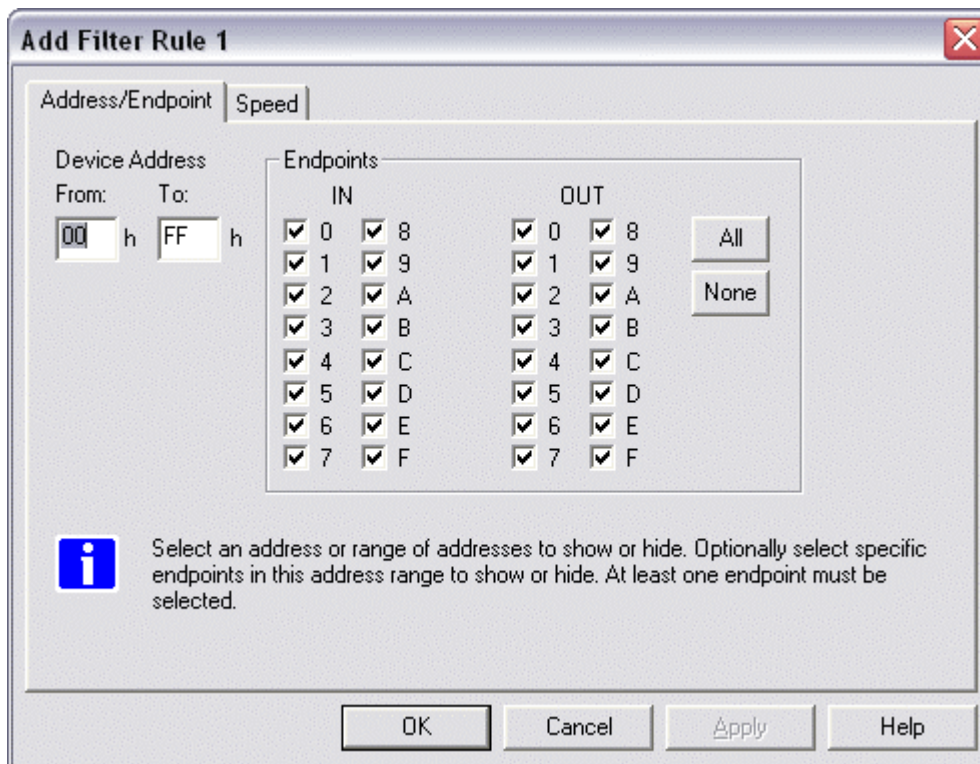
When you click on the 'Custom Filter Settings' button (or select it from the View menu), the following dialog appears:



First decide if your requirement is to 'Show Unless' or to 'Hide Unless'. Choose whatever will result in the simplest rules. When you

add your first rule the filter will automatically be enabled, though you can choose to disable it again.

To add a rule, click on Add Rule. This will display the following dialog:



The dialog box titled "Add Filter Rule 1" has two tabs: "Address/Endpoint" and "Speed". The "Address/Endpoint" tab is active. It contains a "Device Address" section with "From:" and "To:" labels, each followed by a text box containing "00" and "FF" respectively, and a "h" suffix. To the right is an "Endpoints" section with two columns: "IN" and "OUT". Each column has eight rows of checkboxes labeled 0 through 7, with corresponding letters A through F. All checkboxes are checked. To the right of the endpoints are "All" and "None" buttons. At the bottom left is an information icon and a text box with the following text: "Select an address or range of addresses to show or hide. Optionally select specific endpoints in this address range to show or hide. At least one endpoint must be selected." At the bottom right are "OK", "Cancel", "Apply", and "Help" buttons.

Device Address		Endpoints	
From:	To:	IN	OUT
00	FF	<input checked="" type="checkbox"/> 0	<input checked="" type="checkbox"/> 8
		<input checked="" type="checkbox"/> 1	<input checked="" type="checkbox"/> 9
		<input checked="" type="checkbox"/> 2	<input checked="" type="checkbox"/> A
		<input checked="" type="checkbox"/> 3	<input checked="" type="checkbox"/> B
		<input checked="" type="checkbox"/> 4	<input checked="" type="checkbox"/> C
		<input checked="" type="checkbox"/> 5	<input checked="" type="checkbox"/> D
		<input checked="" type="checkbox"/> 6	<input checked="" type="checkbox"/> E
		<input checked="" type="checkbox"/> 7	<input checked="" type="checkbox"/> F

Select an address or range of addresses to show or hide. Optionally select specific endpoints in this address range to show or hide. At least one endpoint must be selected.

There are two tabs to select the conditions for your rule. Both tabs can contribute to the rule if required. The first tab defines device address and endpoint. The default settings include all possible addresses and endpoints, so if you do not make a change the rule will have no effect, and you will not be allowed to create it. Similarly you will not be able to generate a rule which excludes all events.



As an example we specify here that we do not wish to see transactions using (address 3) AND ( (endpoint 1 in) OR (endpoint 2 out) ):

**Add Filter Rule 1**

Address/Endpoint | Speed

Device Address  
From: 3 h To: 3 h

Endpoints

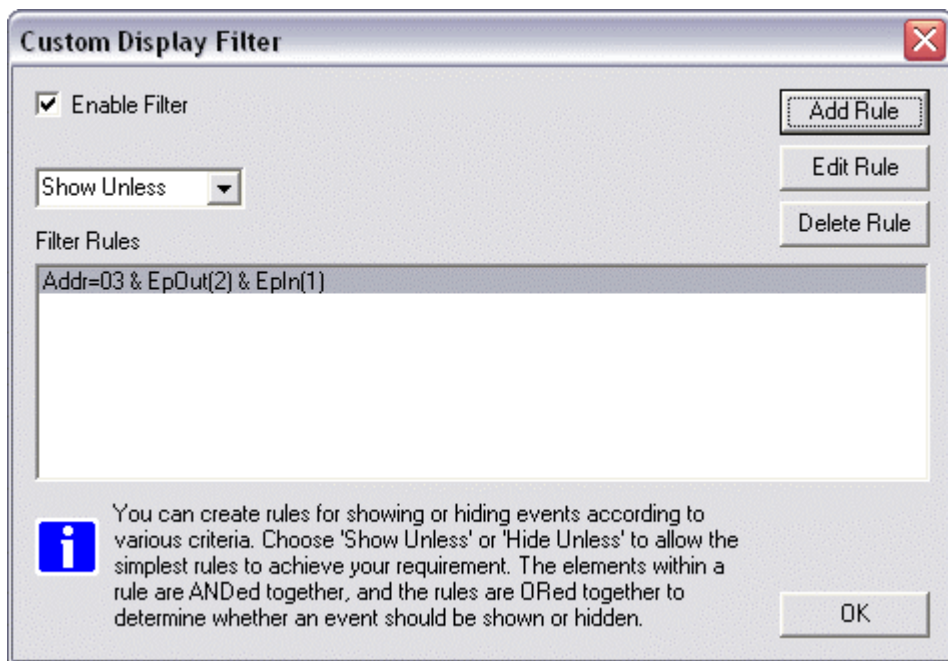
IN		OUT	
<input type="checkbox"/> 0	<input type="checkbox"/> 8	<input type="checkbox"/> 0	<input type="checkbox"/> 8
<input checked="" type="checkbox"/> 1	<input type="checkbox"/> 9	<input type="checkbox"/> 1	<input type="checkbox"/> 9
<input type="checkbox"/> 2	<input type="checkbox"/> A	<input checked="" type="checkbox"/> 2	<input type="checkbox"/> A
<input type="checkbox"/> 3	<input type="checkbox"/> B	<input type="checkbox"/> 3	<input type="checkbox"/> B
<input type="checkbox"/> 4	<input type="checkbox"/> C	<input type="checkbox"/> 4	<input type="checkbox"/> C
<input type="checkbox"/> 5	<input type="checkbox"/> D	<input type="checkbox"/> 5	<input type="checkbox"/> D
<input type="checkbox"/> 6	<input type="checkbox"/> E	<input type="checkbox"/> 6	<input type="checkbox"/> E
<input type="checkbox"/> 7	<input type="checkbox"/> F	<input type="checkbox"/> 7	<input type="checkbox"/> F

All  
None

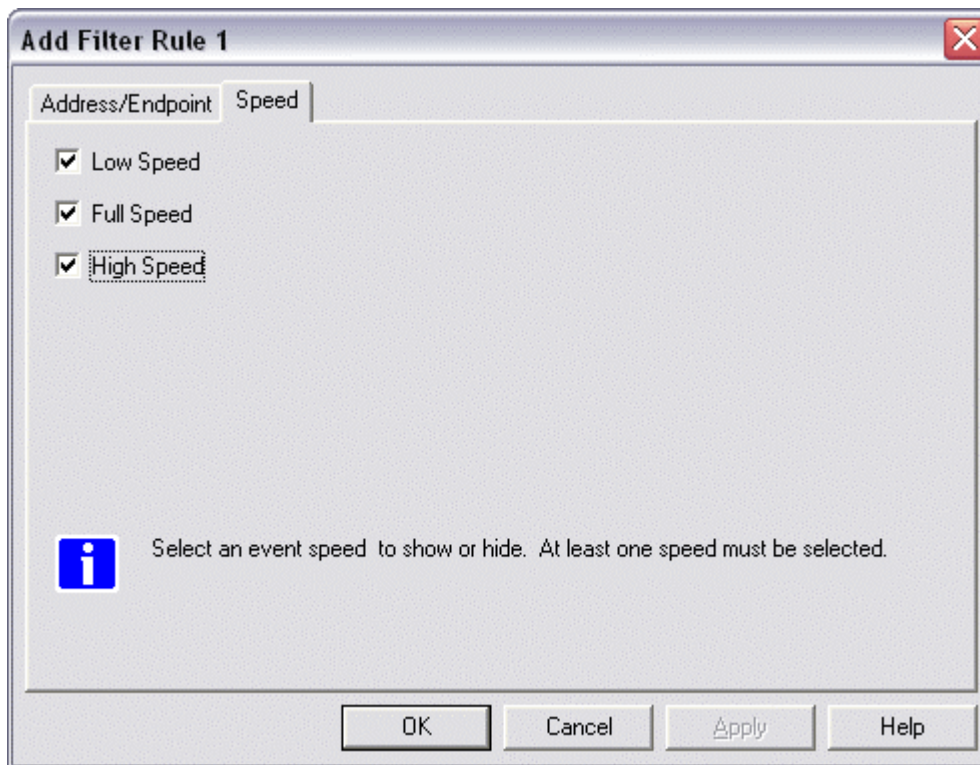
Select an address or range of addresses to show or hide. Optionally select specific endpoints in this address range to show or hide. At least one endpoint must be selected.

OK Cancel Apply Help

When we click on OK we find that the rule has been added to the filter rules box (the filter was also automatically enabled):



The other tab of the Add Rule dialog allows event speed to be included in the filter rule:




### 3.11.2 Custom Filter Enable







#### Custom Filter Enable

The filter rules are global to the application, and can quickly be turned on and off using the custom filter enable button. The Menu item in the View Menu can also be used.

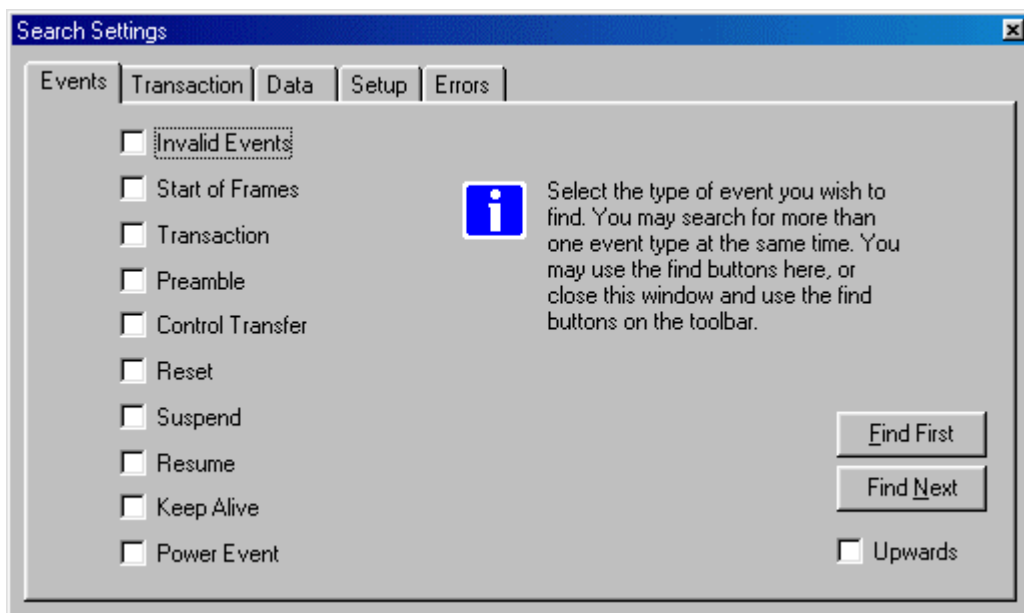
### 3.12 Search

The Search function is used to locate particular events within the captured data. Events which are not currently being displayed will still be found by the search function and the display filter settings will be adjusted accordingly. Select the Search Settings by either clicking the Tool Button  or selecting the item on the Edit menu. Items may be searched for by Event, Transaction, Data, Setup or Error.

Once a search has been defined the Search Settings Window may be closed and the Toolbar Search buttons     used instead. This provides a clearer view of the data.

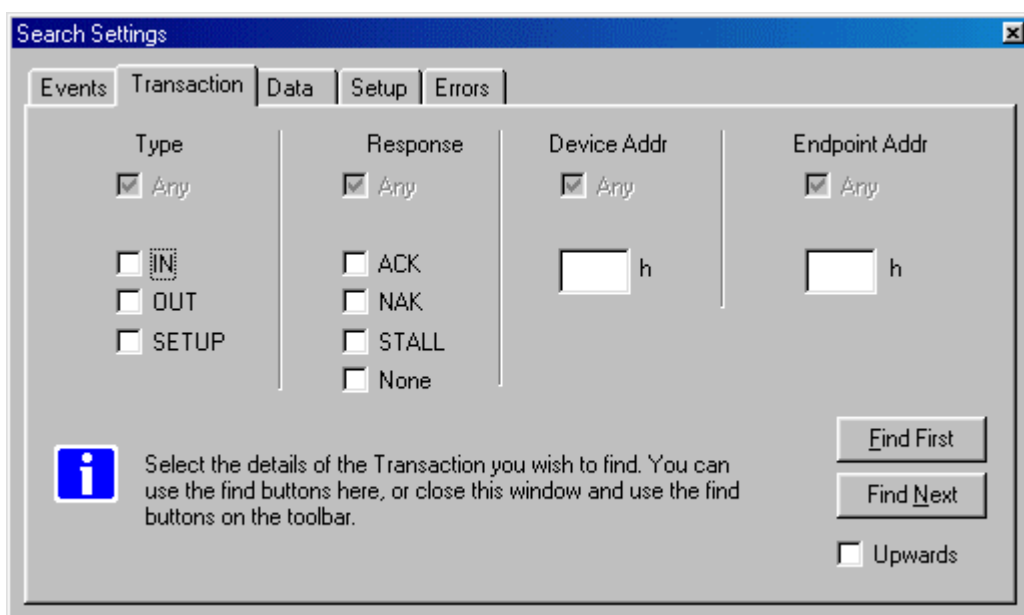
#### 3.12.1 Event Search

The events search allows you to find such items as Reset, Preamble etc.



### 3.12.2 Transaction Search

Transactions may be searched for according to their Type (IN, OUT, or SETUP), the Response (ACK, NAK, STALL or NONE), the Device Address and/or Endpoint. If no selection is made in any particular column then any transaction meeting the requirements of the other columns will be found.



### 3.12.3 Data Search

The data to be searched for is entered as a sequence of Hex bytes separated by spaces. The size of the data field, the Device Address and Endpoint Address can also be specified. The example below shows a search for Hex bytes 'A3 00' in an 8 byte Data field with Device Address 1 and Endpoint 0. If no selection is made in any particular column then any transaction meeting the requirements of the other columns will be found.

The screenshot shows a 'Search Settings' dialog box with a blue title bar and a close button. It has five tabs: 'Events', 'Transaction', 'Data' (selected), 'Setup', and 'Errors'. The 'Data' tab contains four input fields: 'Size' (set to 8, decimal), 'Pattern' (set to A3 00, hexadecimal), 'Device Addr' (set to 1, hexadecimal), and 'Endpoint Addr' (set to 0, hexadecimal). Each field has a checkbox for 'Any' above it. Below these fields is an information icon and a text box explaining that the search will find data in Control Transfers or Transactions based on the specified content, length, device address, and endpoint address. At the bottom right are 'Find First', 'Find Next', and 'Upwards' buttons.

Size	Pattern	Device Addr	Endpoint Addr
<input type="checkbox"/> Any	<input type="checkbox"/> Any	<input type="checkbox"/> Any	<input type="checkbox"/> Any
8 decimal	A3 00 h	1 h	0 h

This will find data in Control Transfers or Transactions. Select the content of a data field you wish to find. You can specify a series of hex bytes, separated by spaces, or you can specify the total length of the data field you wish to find.

You may use the find buttons here, or close this window and use the find buttons on the toolbar.

Find First  
Find Next  
☐ Upwards

### 3.12.4 Setup Search

The example below demonstrates a search for a bRequest of 05h in a standard Setup to a device having Address 0 and Endpoint 0. Masks are available if you wish to test for only a part of a field. If no selection is made in any particular column then any transaction meeting the requirements of the other columns will be found.

**Search Settings**

Events | Transaction | Data | **Setup** | Errors

Direction:	Type:	Recipient:	bRequest	wValue	wIndex	wLength
<input type="checkbox"/> Either	<input type="checkbox"/> Any	<input type="checkbox"/> Any	<input type="checkbox"/> Any	<input checked="" type="checkbox"/> Any	<input checked="" type="checkbox"/> Any	<input checked="" type="checkbox"/> Any
<input checked="" type="checkbox"/> To Dev	<input checked="" type="checkbox"/> Standard	<input checked="" type="checkbox"/> Device	05 h	h	h	h
<input type="checkbox"/> To Host	<input type="checkbox"/> Class	<input type="checkbox"/> Interface	Mask	Mask	Mask	Mask
	<input type="checkbox"/> Vendor	<input type="checkbox"/> Endpoint	FF h	FFFF h	FFFF h	FFFF h
	<input type="checkbox"/> Other	<input type="checkbox"/> Reserved				

Destination

Device Addr	Endpoint Addr
<input type="checkbox"/> Any	<input type="checkbox"/> Any
0 h	0 h

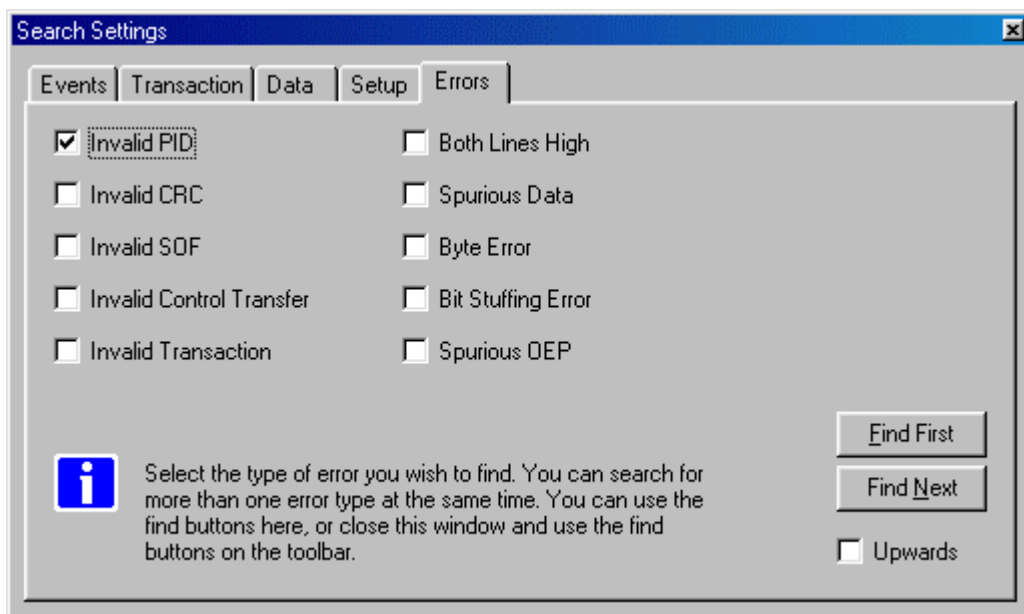
**i** Select the details of the Setup you wish to find.

You can use the find buttons here, or close this window and use the find buttons on the toolbar.

☐ Upwards

### 3.12.5 Error Search

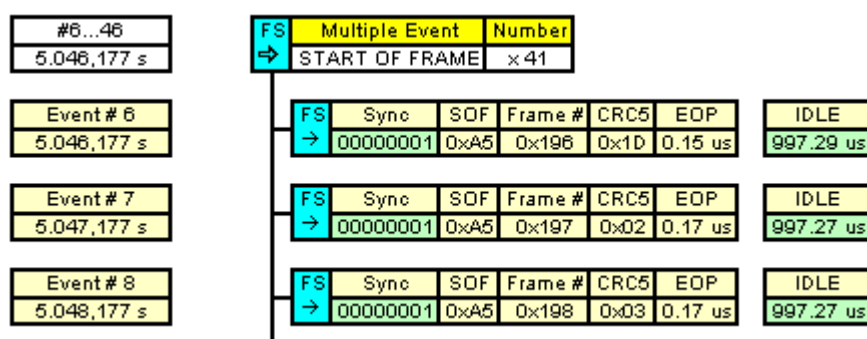
Errors such as Invalid PID, Invalid CRC etc may be found by selecting the appropriate boxes. A more detailed explanation of these errors is given in the Errors Chapter.



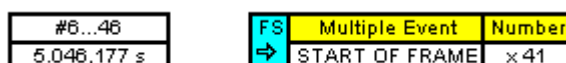


### 3.13 Multiple Event Headers

Numerous consecutive Start of Frame packets or Keep Alive events make the display difficult to read. GraphicUSB inserts multiple Event Headers before such sequences. The packets can be hidden by clicking on the “Show Packets” button with the multiple Events Header still being visible. The multiple Events Headers can be hidden by clicking on the “Show SOFs” button. The example below shows 41 SOF’s grouped together.



Show SOF Packets



Hide SOF Packets

### 3.14 Bookmarks

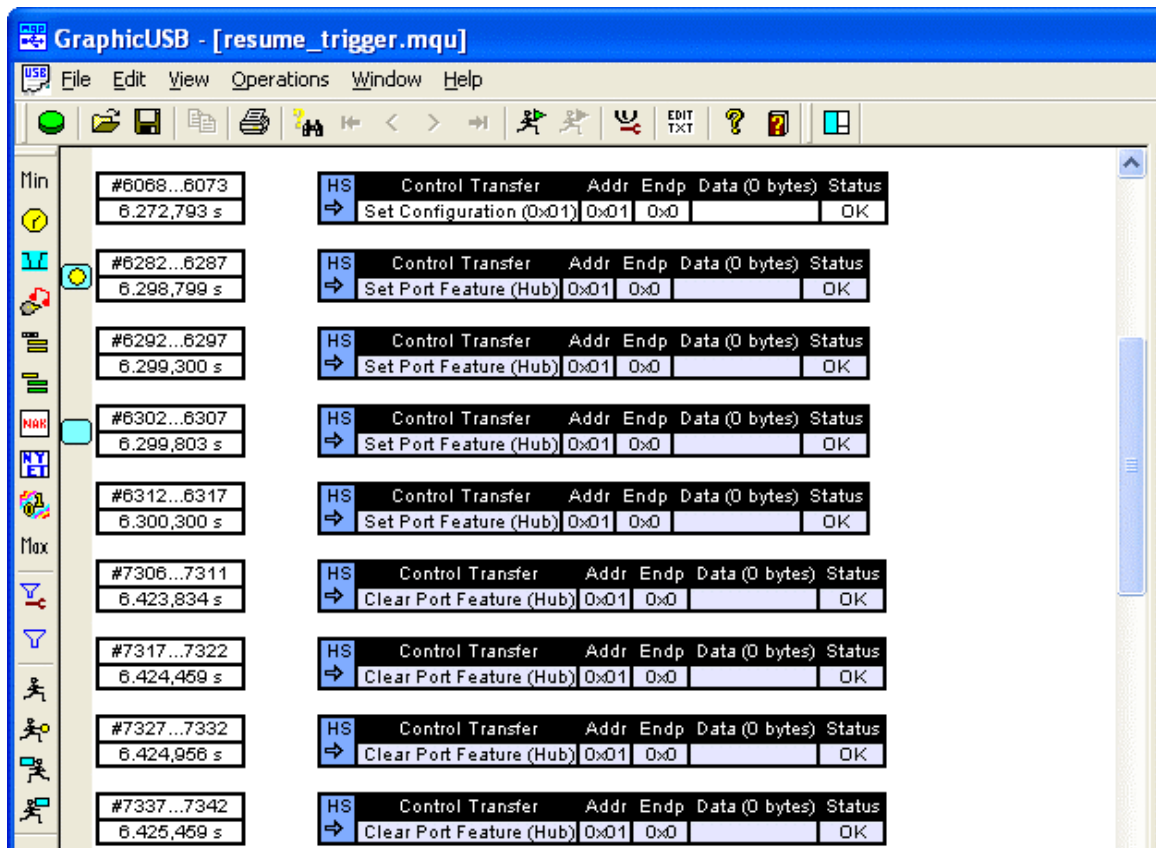
A bookmark allows you to mark an event of interest, allowing you to locate it quickly when it is not showing in the window.

You can add a Bookmark to any event in the display by any of the following methods.



- Click on the event in question to select it, then Menu...View... Add Bookmark.
- Click on the event in question to select it, then use keyboard Ctrl+F2.

- Right click on the event in question, and choose 'Add/Remove Bookmark' from the pop-up menu.



Bookmarked events are marked with a light blue rectangle. The following screenshot shows two bookmarked events (one of them is also selected).



You can locate the next or the previous Bookmark by:

- clicking on the tool bar icons,  or 
- Menu...View... Go to Next Bookmark or Go to Previous Bookmark respectively, or
- Using keyboard F2 or Shift+F2 respectively

### 3.15 Printing

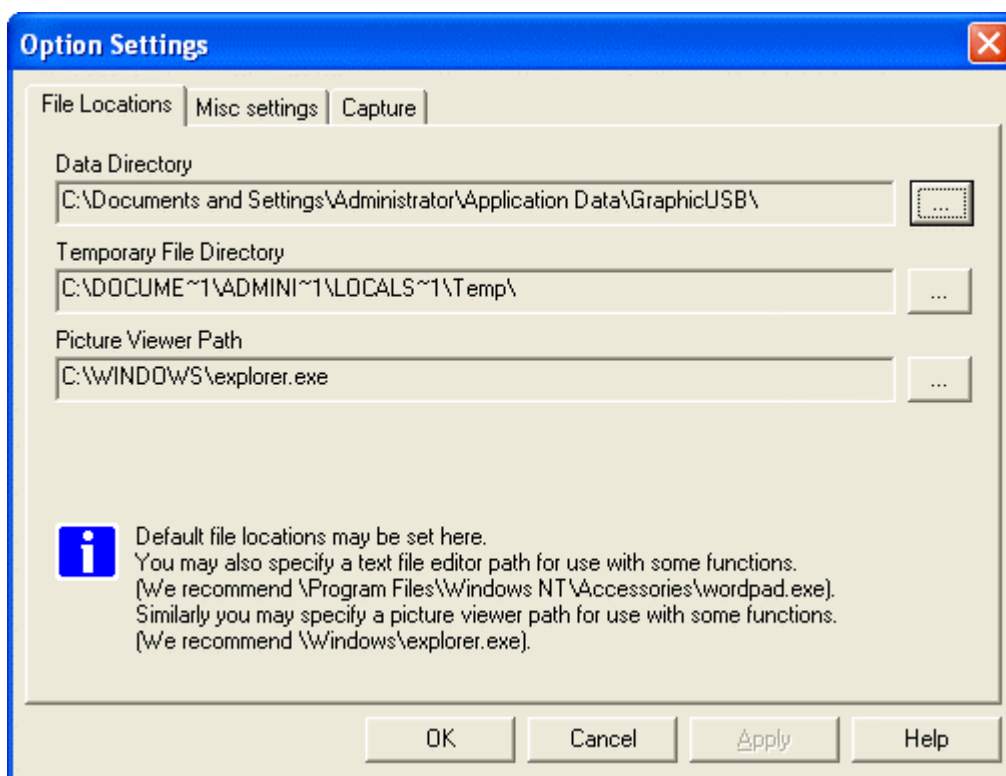
Any of the panes may be printed. To select a pane to print, click on that pane, or click on the Select Print Pane icon  on the tool bar until the required pane is indicated. Then print in the usual way, either from the file menu, or using the print icon  on the tool bar.

### 3.16 Option Settings

Select Options in the Edit menu to open the Option Settings Window.

#### 3.16.1 File Locations

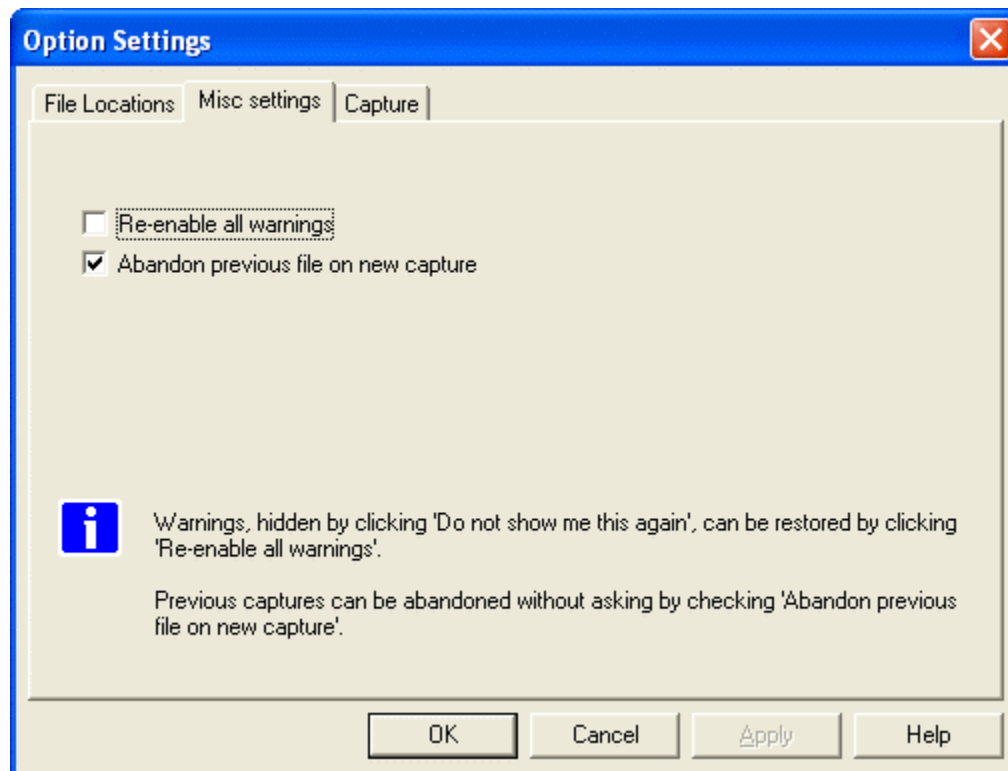
Use this to specify the locations of the Capture and temporary files.



A Picture Viewer may also be specified, to assist with certain functions, such as showing images transferred in Image Class devices.

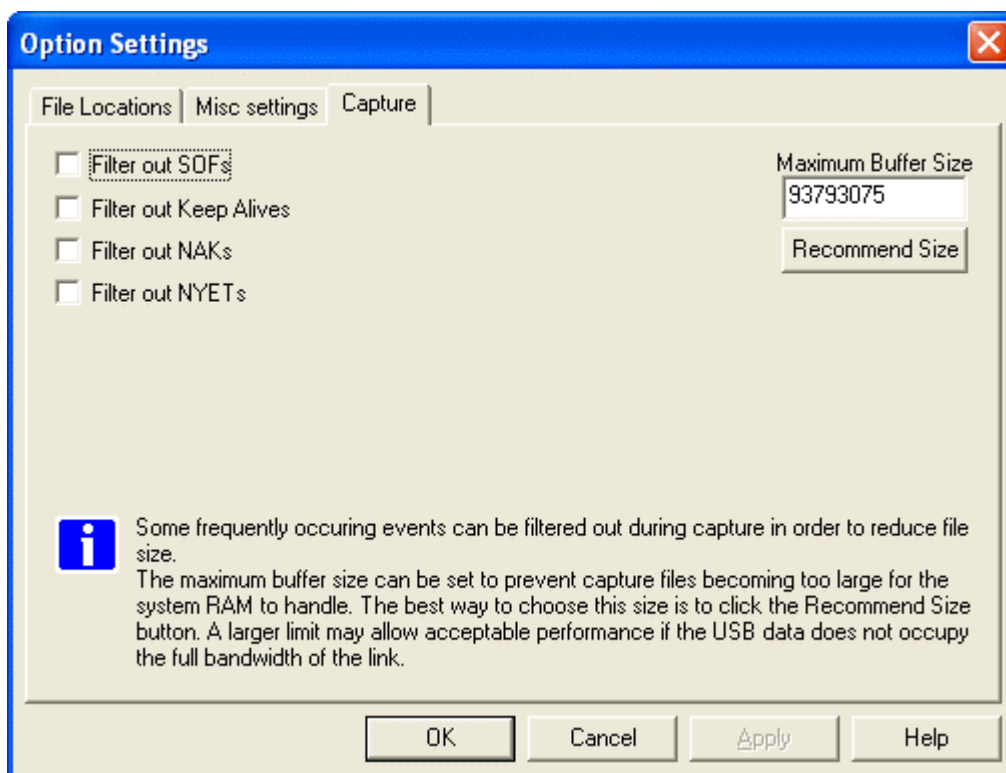
### 3.16.2 Miscellaneous Settings

- User warnings e.g. concerning the test set-up, may be re-enabled.
- If several captures are to be done in a row without the need to save each capture, then it can be beneficial to check the box 'Abandon previous file on new capture'. This will save RAM and make shutting down GraphicUSB quicker.



### 3.16.3 Capture

The size of the Capture file may be minimized by filtering Start of Frame or Keep Alive events or NAKed transactions or NYETed transactions. Please note that NAKed control transactions will always be included in the capture.



The size of the capture buffer defaults to a safe value, which should guarantee a reliable capture under most circumstances. You may increase this size as required, but you should be prepared to decrease it again if you start to have poor response caused by the system using virtual memory to satisfy your requirement. The 'Recommend Size' button returns the buffer size to the default value for your system. An option well worth considering is to increase the size of the RAM in your PC.

### 3.17 Command Line Capture Control

It is possible to control GraphicUSB from another application using a command-line syntax. This allows another application to perform a capture without GraphicUSB becoming visible. For the sake of simplicity, the following examples show the commands being issued by use of the older WinExec() function. You may wish to use a more recent function such as CreateProcess() or the .NET function Process.Start(). The strings are all supposed to be on one line but will be shown split in this document.

The filename is shown in the examples without a full path. In this case the file will be saved in the Application Data folder for GraphicUSB. You can also specify a full pathname in order to save the file in a folder of your choice.

The available commands, which make use of the switches '-c0', '-c1' and '-c2', are as follows:

#### 3.17.1 Start Capture

```
::WinExec("c:\\Program Files\\MQP  
Electronics\\GraphicUSB\\GraphicUSB -c1 TestCapt.mqu",  
SW_HIDE);
```

This will start GraphicUSB running, define TestCapt.mqu as the location to save the capture when complete, and start the capture. If GraphicUSB is already running visibly, it will become invisible and start the capture.

#### 3.17.2 Restart Capture

```
::WinExec("c:\\Program Files\\MQP  
Electronics\\GraphicUSB\\GraphicUSB -c2", SW_HIDE);
```

This assumes that a capture is in progress, (else an error message is displayed). It will abandon the capture in progress and start it again.

### 3.17.3 Stop Capture

```
::WinExec("c:\\Program Files\\MQP  
Electronics\\GraphicUSB\\GraphicUSB -c0", SW_HIDE);
```

This assumes that a capture is in progress, (else an error message is displayed). It will stop the capture in progress and save it to the file named in the Start Capture command.

### 3.17.4 Display File

```
::WinExec("c:\\Program Files\\MQP  
Electronics\\GraphicUSB\\GraphicUSB TestCapt.mqu ",  
SW_SHOW);
```

This will display the captured file specified, in a visible instance of GraphicUSB.

### **3.18 Export Functions**

GraphicUSB allows various types of information to be exported to text based formats for further analysis or processing by the user. The following types of export are currently provided:

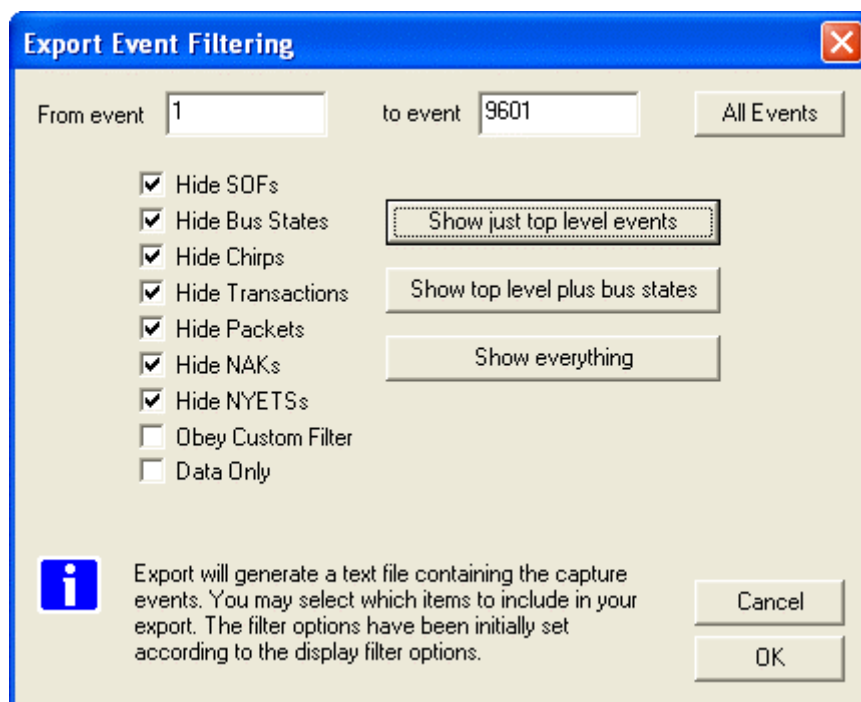
- Capture Event Information
- Data from a specific event
- Descriptors

Typically the exported text will be displayed in the application window, ready to be saved to a file using the usual File...Save... functions. The exception is that when the exported file is defined as 'binary', the file save dialog appears immediately, and the file is not displayed in the application window.

#### **3.18.1 Exporting Capture Events**

With a capture document open, select the item 'Create Events File...' from the File menu. The 'Export' dialog will be displayed:



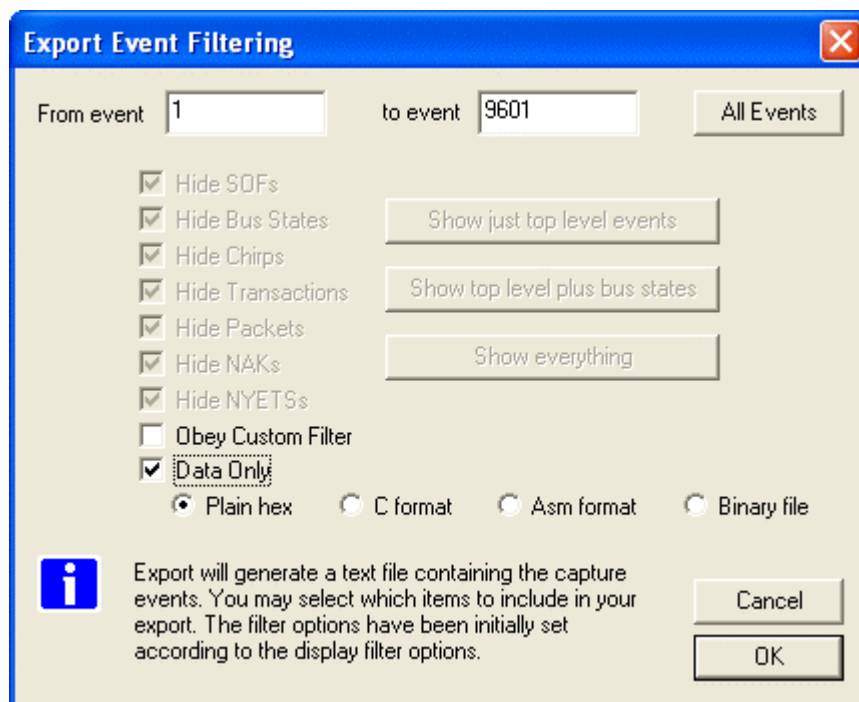


Select the event range, and the level of detail you wish to export, then click on OK. A typical output text file (showing just top level events) is shown below:

```
[3.742,754] LS: Control Transfer Addr:00 Endp:0 - Get Device Descriptor
12 01 10 01 00 00 00 08 62 0F 01 10 01 00 01 02
00 01
[3.773,991] LS: Control Transfer Addr:00 Endp:0 - Set Address (0x01)
[Zero Data Bytes]
[3.836,513] LS: Control Transfer Addr:01 Endp:0 - Get Device Descriptor
12 01 10 01 00 00 00 08 62 0F 01 10 01 00 01 02
00 01
[3.838,518] LS: Control Transfer Addr:01 Endp:0 - Get Configuration Descriptor
09 02 22 00 01 01 00 A0 32
[3.839,614] LS: Control Transfer Addr:01 Endp:0 - Get Configuration Descriptor
09 02 22 00 01 01 00 A0 32 09 04 00 00 01 03 01
02 00 09 21 10 01 00 01 22 34 00 07 05 81 03 04
00 0A
[3.842,658] LS: Control Transfer Addr:01 Endp:0 - Get String Descriptor 238
0C 03 41 00 42 00 43 00 44 00 41 00
```

### 3.18.2 Exporting Capture Events – Data Only

A useful option in the Export events dialog is 'Data Only'.

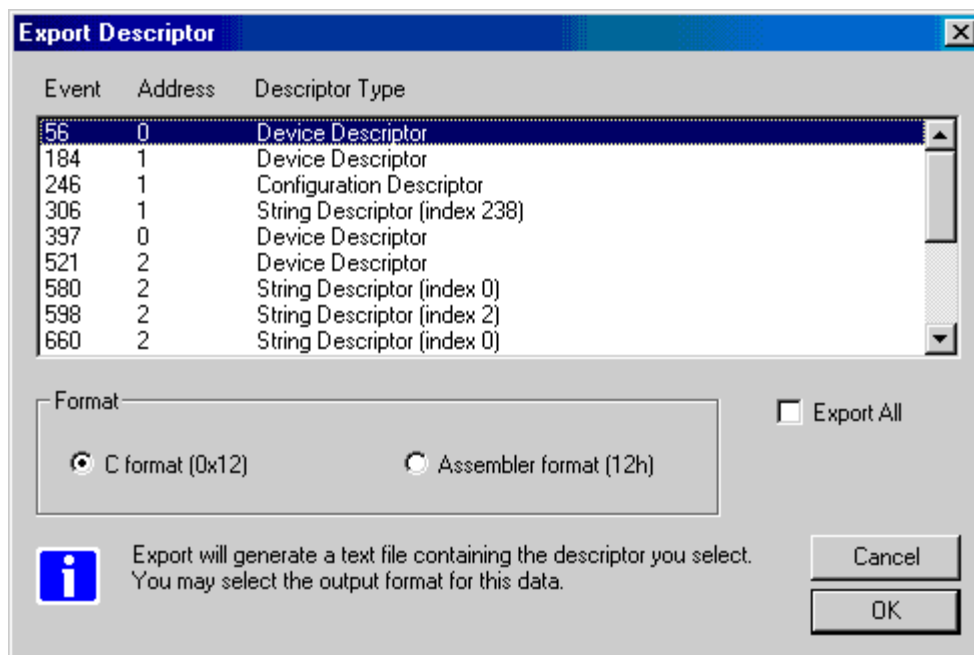


Selecting the 'Data Only' option allows data, as selected by the custom filter, and the event range, to be exported in a number of useful file formats.

### 3.18.3 Exporting Descriptors

#### 3.18.3.1 Standard Descriptors

With a capture document open, select the item 'Create Descriptor File...' from the File menu. The 'Export Descriptor' dialog will be displayed:



Select the descriptor you wish to export (or click on Export All). To assist you in deciding which is the appropriate descriptor, the event number and the device address are displayed. If you had previously selected a valid descriptor in the capture pane, then this descriptor will be pre-selected when you open this dialog.

You should now choose the format in which you wish to export the descriptor. By default it will be output as a (commented) 'c' code structure.

When you have made your selection, click on OK and you will be invited to choose the name and location of the exported file. An example file is shown below:

```
// Device Descriptor (event number 56)
static const unsigned char descriptor56[] =
{
    0x12,          // bLength
    0x01,          // bDescriptorType (DEVICE)
    0x10,          // bcdUSB (ls byte)
    0x01,          // bcdUSB (ms byte)
    0x00,          // bDeviceClass (Defined in Interface)
    0x00,          // bDeviceSubClass
    0x00,          // bDeviceProtocol
    0x08,          // bMaxPacketSize0
    0x62,          // idVendor (ls byte)
    0x0F,          // idVendor (ms byte)
    0x01,          // idProduct (ls byte)
    0x10,          // idProduct (ms byte)
    0x01,          // bcdDevice (ls byte)
    0x00,          // bcdDevice (ms byte)
    0x01,          // iManufacturer
    0x02,          // iProduct
    0x00,          // iSerialNumber
    0x01,          // bNumConfiguration
};
```

### 3.18.3.2 Class Specific Descriptors

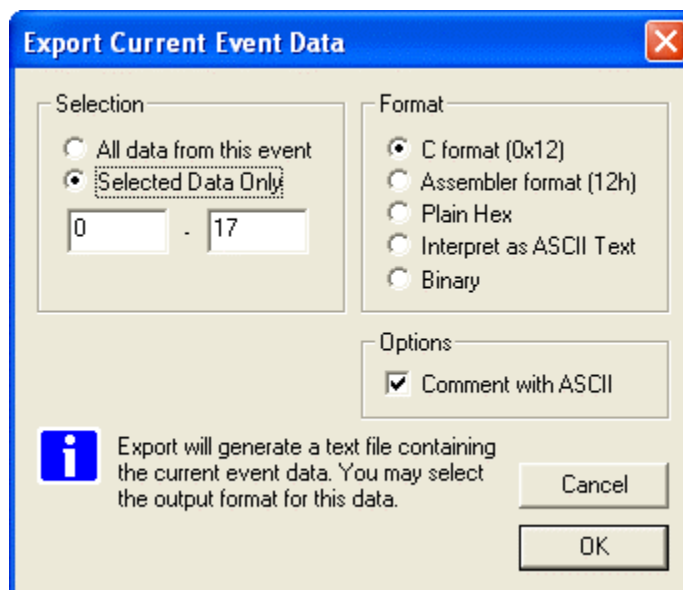
In some cases, where class analysis options have been installed, specific class descriptors are annotated, e.g.:

```
// HID Report Descriptor (event number 6185)
static const unsigned char descriptor6185[] =
{
    0x05,      // Usage Page (Generic Desktop Controls)
    0x01,      //
    0x09,      // Usage (Mouse)
    0x02,      //
    0xA1,      //   Collection (Application)
    0x01,      //
    0x09,      //     Usage (Pointer)
    0x01,      //
    0xA1,      //       Collection (Physical)
    0x00,      //
    0x05,      //         Usage Page (Button)
    0x09,      //
    0x19,      //           Usage Minimum (1)
    0x01,      //
    0x29,      //           Usage Maximum (5)
    0x05,      //
    0x15,      //             Logical Minimum (0)
    0x00,      //
    0x25,      //             Logical Maximum (1)
    0x01,      //
    0x95,      //             Report Count (5)
    0x05,      //
    0x75,      //             Report Size (1)
    0x01,      //
    0x81,      //             Input (Data, Variable, Absolute, Bit Field)
    0x02,      //
    0x95,      //             Report Count (1)
    0x01,      //
    0x75,      //             Report Size (3)
    0x03,      //             Input (Data, Variable, Absolute, Bit Field)
```

Other class descriptors can still be exported, but without the comment annotation.

### 3.18.4 Exporting Data from a Specific Event

With a capture document open, select the event from which you wish to export the data, by clicking on the event in the event pane (making it the 'Current Event'). Then select the item 'Create Current Data File...' from the File menu. The 'Export Current Event Data' dialog will be displayed:



Select the format in which you wish to export the data. When you have made your selection, click on OK and the text file will be displayed, ready for you to edit, or save to your chosen location.

If a binary format is selected, you will be invited to choose the name and location of the exported file.

An example text file is shown below:

```
0xE2, 0x00, 0xE2, 0x00, 0xEA, 0x00, 0xEA, 0x00
0xED, 0x00, 0xED, 0x00, 0xEA, 0x00, 0xEA, 0x00
0xDC, 0x00, 0xDC, 0x00, 0xC2, 0x00, 0xC2, 0x00
0x9C, 0x00, 0x9C, 0x00, 0x6C, 0x00, 0x6C, 0x00
0x34, 0x00, 0x34, 0x00, 0xF6, 0xFF, 0xF6, 0xFF
0xB2, 0xFF, 0xB2, 0xFF, 0x6D, 0xFF, 0x6D, 0xFF
0x2C, 0xFF, 0x2C, 0xFF, 0xF0, 0xFE, 0xF0, 0xFE
0xBA, 0xFE, 0xBA, 0xFE, 0x89, 0xFE, 0x89, 0xFE
0x60, 0xFE, 0x60, 0xFE, 0x41, 0xFE, 0x41, 0xFE
0x2D, 0xFE, 0x2D, 0xFE, 0x23, 0xFE, 0x23, 0xFE
```

## **3.19 Text Editing**

### **3.19.1 Introduction**

A number of GraphicUSB document types are basically normal text files, sometimes with special extensions. These include:

- exported event file (\*.txt)
- exported data file (\*.txt)
- exported descriptor file (\*.c, \*.asm)
- exported current event data file (\*.c, \*.asm)
- generator script (\*.mgen)
- vendor class information file (\*.mven)
- device information file (\*.mdev)

These file types are all opened in the GraphicUSB built-in text editor. The editor has the following features:

- contextual colouring
- bookmark capability
- goto line number
- printing and selection printing
- find and replace functions
- word selection by double mouse click
- dragging of selected blocks

Additionally a second, 'output' pane is associated with certain file types.

- generator script
- vendor class information file
- device information file

This is used to display validation or compilation output.

### 3.19.2 Editing

All the normal text-editing functions are implemented in an industry standard way, so that using the editor should be instinctive, therefore not requiring much description here.

Available keyboard accelerators are shown against the functions in the menus, in the usual way.

e.g. Add Bookmark      Ctrl+F2

### 3.19.3 Bookmarks

Any line the text file may be book-marked, by first putting the caret on that line, and then pressing Ctrl+F2. A blue marker appears in the grey left-hand column to indicate that the line is bookmarked.

Pressing the F2 button takes the caret in turn to the start of each bookmarked line working in a forward direction through the file. Shift+F2 takes the caret in turn to the start of each bookmarked line working in a backward direction through the file.

To remove a bookmark, put the caret on that line, and then press Ctrl+F2.

Bookmarks only exist while the file is open.

### 3.19.4 Error Messages

In files types with an output pane below, this pane is use to display the result of validation or compilation. If any error messages are shown, then you may cycle through the errors by pressing F4. Each error message will be highlighted and the corresponding source line will be marked.

In a similar way, if you double-click on an error message in the lower pane, the line in question will be marked in the upper pane.



## **3.20 USB Errors**

### **3.20.1 Invalid PID**

A Packet Identifier, PID, is a 4 bit code. The 4 bits of the PID are complemented and repeated making an 8 bit PID in total. An error in the transmission of the PID will result in an Invalid PID being reported.

### **3.20.2 Invalid CRC**

A Cyclic Redundancy Check is performed on the data transmitted in a packet. Token packets have a 5 bit CRC while Data packets have a 16 bit CRC. The CRC is checked by the Packet-Master and, if incorrect, an error is reported.

### **3.20.3 Invalid SOF**

A Start of Frame packet contains a frame number. If a frame number is out of sequence then an Invalid SOF error is reported. It's likely that frame numbers will be out of sequence after a Reset or Suspend; in these cases the error can be ignored.

### **3.20.4 Invalid Control Transfer**

A Control transfer consists of a SETUP packet (which defines a from-host or to-host direction), followed by an optional set of 'Data Stage' DATA0/DATA1 packets in that direction, completed by a 'Status Stage' zero-length DATA1 packet, in the other direction. If this sequence is not correct then an Invalid Control Transfer error is reported.

The correct sequence for the data toggle in a Control Transfer is that the SETUP should contain a DATA0 packet, the Data Stage should start with a DATA1 packet and then alternate, and finally the Status Stage should be a zero-length DATA1 packet. If these polarities are not correct then an Invalid Control Transfer error is reported.

### **3.20.5 Invalid Transaction**

A transaction consists of a token packet (SETUP/IN/OUT), followed by a DATA0 or DATA1 packet (in the appropriate direction), and completed by an ACK, NAK or STALL. Either the last or the last two packets may be missing. If this sequence is not correct then an Invalid Transaction error is reported.

A SETUP transaction must contain a DATA0 packet. If this polarity is not correct then an Invalid Transaction error is reported.

### **3.20.6 Bit Stuffing Error**

In order to ensure adequate signal transitions, bit stuffing is employed by the transmitting device when sending a USB packet. A zero is inserted after every six consecutive ones in the data stream before the data is NRZI encoded. If more than six consecutive ones are detected a Bit Stuffing Error is reported.

### **3.20.7 Byte Error**

All packets must have an integral number of bytes. If this is not the case a Byte Error is reported.

### **3.20.8 Spurious Data**

If data is detected but doesn't begin with a synchronization pattern then the display will report Spurious Data.

### **3.20.9 Both Lines High**

The data encoding scheme is such that the D+ and D- lines should never both be high at the same time. If this condition is encountered an error is reported.

### **3.20.10 Spurious End of Packet**

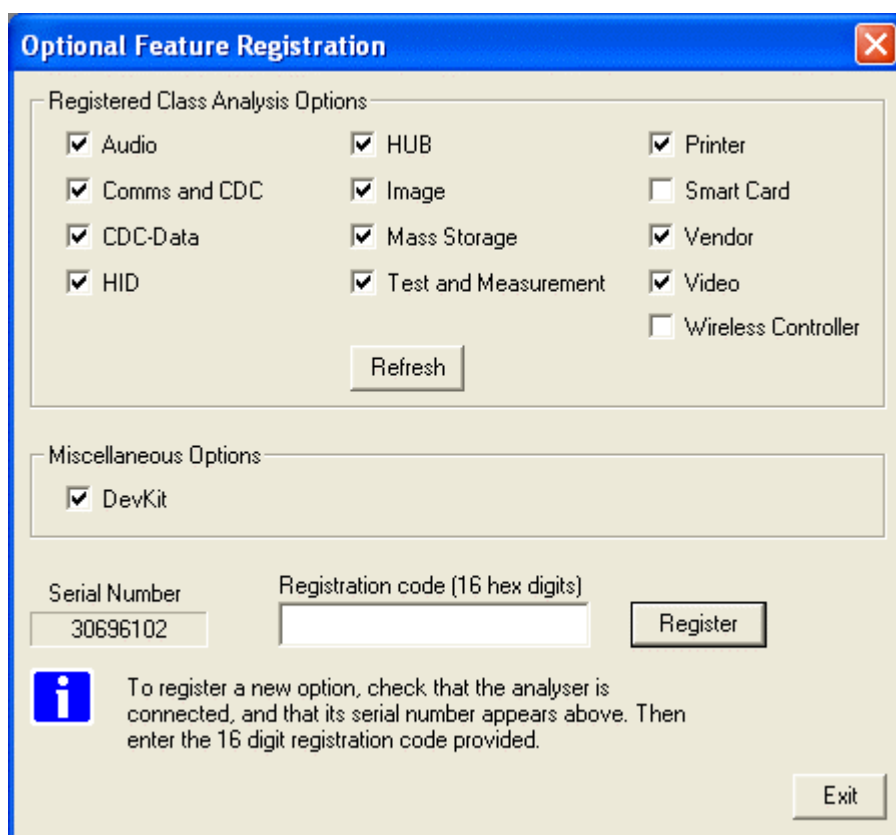
An End of Packet condition should only appear at the end of a data packet. If the condition appears at any other time it will be reported as an error.

## 3.21 Class Analysis Options

### 3.21.1 Registration

The Class Analysis Options are supplied as software add-ons for GraphicUSB. The options are available for individual classes, so you only need to purchase the functionality you actually require. The options are provided in the form of 16 digit hexadecimal registration codes.

To enable a particular option, first ensure the analyser is connected to the host, and then click in the menu bar on Edit...Class Analysis...Register... and the following dialog will appear.



The dialog box is titled "Optional Feature Registration" and contains the following elements:

- Registered Class Analysis Options:** A list of 12 options with checkboxes:
  - ☒ Audio
  - ☒ Comms and CDC
  - ☒ CDC-Data
  - ☒ HID
  - ☒ HUB
  - ☒ Image
  - ☒ Mass Storage
  - ☒ Test and Measurement
  - ☒ Printer
  - ☐ Smart Card
  - ☒ Vendor
  - ☒ Video
  - ☐ Wireless Controller
- Miscellaneous Options:** A section with one checked option:
  - ☒ DevKit
- Serial Number:** A text box containing "30696102".
- Registration code (16 hex digits):** An empty text box.
- Buttons:** "Refresh", "Register", and "Exit".
- Information:** An information icon (i) and a text block: "To register a new option, check that the analyser is connected, and that its serial number appears above. Then enter the 16 digit registration code provided."

Enter the registration code provided and the corresponding option should become checked. Please store the registration code carefully in case you need to install the option on another host computer.

### **3.21.2 Analysis Overview**

The Class Analysis option you have enabled will enhance all captures performed on the analyser in question. If you use the analyser on a different host, remember to register the option on that computer as well.

The option will not allow the analysis of classes on captures performed with the analyser before the option was registered. However the captured files can later be viewed in their analysed form on any computer with or without the analyser present.

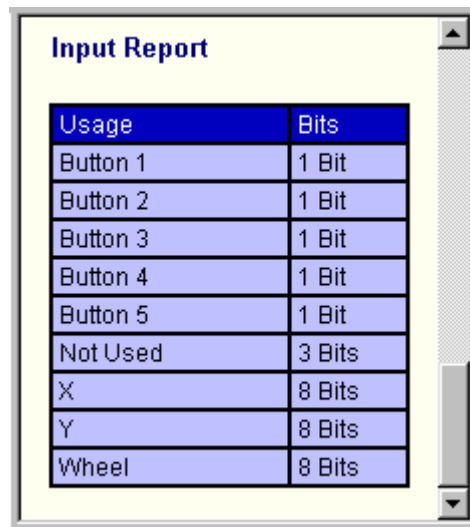
A typical class analysis example is shown below.

**i Control Transfer**

**Get HID Report Descriptor**

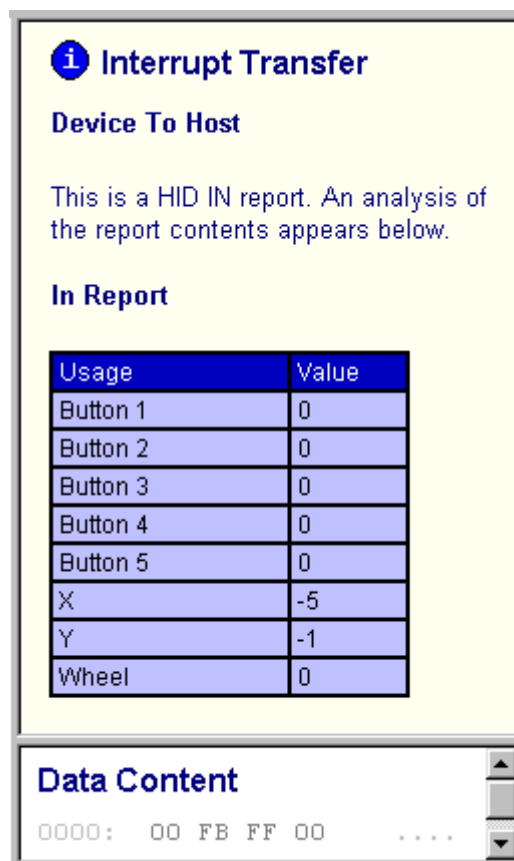
Meaning	Value
Usage Page (Generic Desktop Controls)	05 01
Usage (Mouse)	09 02
Collection (Application)	A1 01
Usage (Pointer)	09 01
Collection (Physical)	A1 00
Usage Page (Button)	05 09
Usage Minimum (1)	19 01
Usage Maximum (5)	29 05
Logical Minimum (0)	15 00
Logical Maximum (1)	25 01
Report Count (5)	95 05
Report Size (1)	75 01
Input (Data, Variable, Absolute, Bit Field)	81 02
Report Count (1)	95 01
Report Size (3)	75 03
Input (Constant, Array, Absolute, Bit Field)	81 01
Usage Page (Generic Desktop Controls)	05 01
Usage (X)	09 30
Usage (Y)	09 31
Usage (Wheel)	09 38
Logical Minimum (-127)	15 81
Logical Maximum (127)	25 7F
Report Size (8)	75 08
Report Count (3)	95 03
Input (Data, Variable, Relative, Bit Field)	81 06
End Collection	C0
End Collection	C0

This shows a HID Report Descriptor, and below is the result of parsing it.

A screenshot of a software window titled "Input Report". It contains a table with two columns: "Usage" and "Bits". The table lists various input elements and their corresponding bit sizes.

Usage	Bits
Button 1	1 Bit
Button 2	1 Bit
Button 3	1 Bit
Button 4	1 Bit
Button 5	1 Bit
Not Used	3 Bits
X	8 Bits
Y	8 Bits
Wheel	8 Bits

Each transfer of a HID report is also analysed, as follows.

A screenshot of a software window titled "i Interrupt Transfer". It contains information about a HID IN report, including a description, a table of report contents, and a section for data content.

**i Interrupt Transfer**

**Device To Host**

This is a HID IN report. An analysis of the report contents appears below.

**In Report**

Usage	Value
Button 1	0
Button 2	0
Button 3	0
Button 4	0
Button 5	0
X	-5
Y	-1
Wheel	0

**Data Content**

0000: 00 FB FF 00 ....

### 3.21.3 Vendor Class Analysis

#### 3.21.3.1 Introduction

Unlike other USB classes, Vendor Class does not have a predetermined specification. The class is made up of whatever control requests and data transfer types a vendor finds useful. For this reason, it is normally difficult to provide a useful analysis. MQP's vendor class analysis option attempts to overcome this problem, by allowing the user to specify characteristics of the vendor class, which can usefully be displayed on the capture document screens. To do this, the vendor class analysis option must be registered.

The user will need to provide a specification file for their device. The file will have a predetermined file name as follows:

**vendVVVVPPPP.mven**

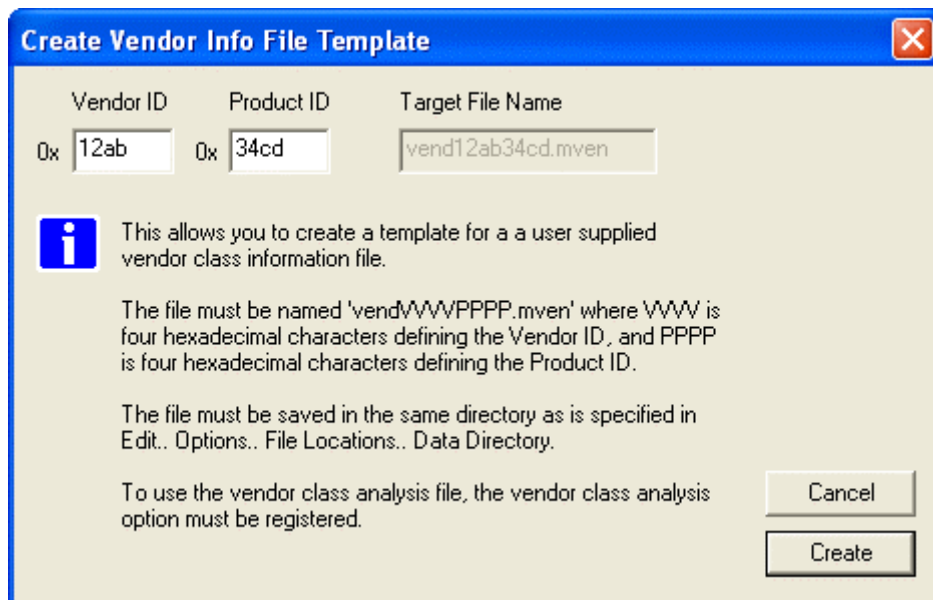
where VVVV is four hexadecimal characters defining the Vendor ID, and PPPP is four hexadecimal characters defining the Product ID.

So for example the file defining the vendor characteristics of a device with Vendor ID 0x12ab and Product ID 0x34cd would be called:

**vend12ac34cd.mven**

The file must be located in the same directory as is specified in: *Edit.. Options.. File Locations.. Data Directory..* (by default this is the standard location for application data defined by the operating system).

A template for this file can be quickly created (with the correct filename, in the correct folder), by: *Operations.. Create Vendor File Template..*



### Notes for users with files generated in previous versions

In previous versions of GraphicUSB this file had a .txt extension, however we now use an extension of .mven. Additionally a

```
FileType MQPVEN 1
```

command is now required at the start of the file.

Otherwise the format has not changed. The editor is now integrated with GraphicUSB, so it is no longer necessary to use an external text editor.

On running the new GraphicUSB for the first time, the application will offer to make these changes automatically for you.

The files can then be opened for editing if required, in GraphicUSB by selecting menu item File...Open... and choosing Files of Type: Vendor Files (\*.mven) in the file select dialog.

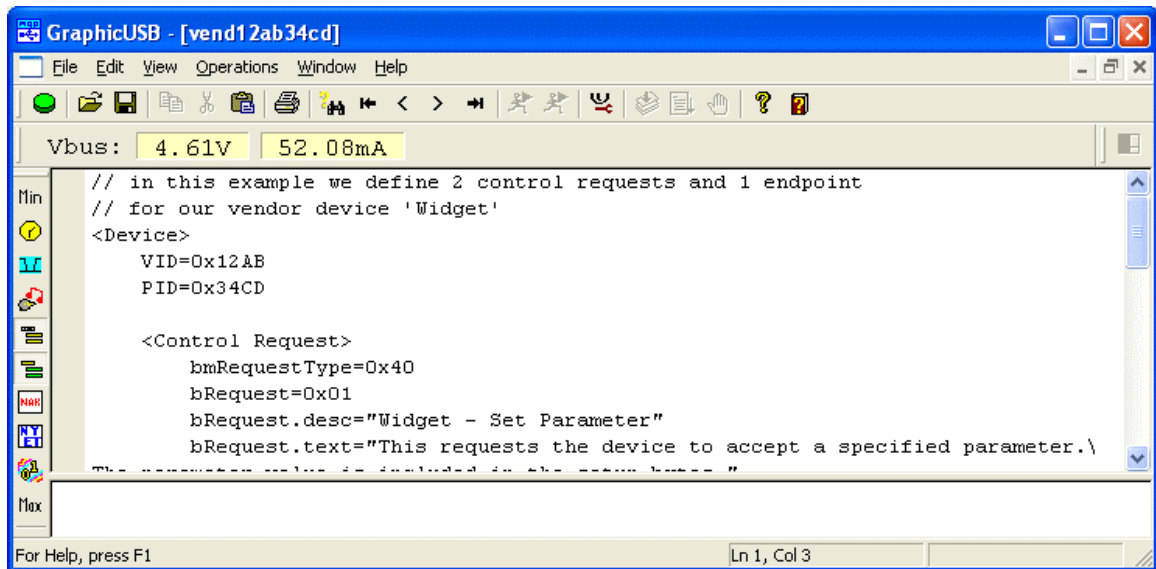
Note also that the validation function now works on the open file displayed in the editing window, rather than by locating the file on the





disk, so the validation option is only available in the menu when the file is displayed.

On clicking the Create button, the template file is generated:



The vendor info file opens in an editing window. Below it is an output pane, used by the built-in validation function.



The template file will have this typical appearance:

```
// in this example we define 2 control requests and 1 endpoint
// for our vendor device 'Widget'

FileType MQPVEN 1
<Device>
    VID=0x12AB    // the Vendor ID
    PID=0x34CD    // the Product ID

    <Control Request>
        bmRequestType=0x40
        bRequest=0x01
        bRequest.desc="Widget - Set Parameter"
        bRequest.text="This requests the device to accept a specified \
parameter.\n\nThe parameter value is included in the setup bytes."
        wIndex.desc="Parameter Number"
        wValue.desc="Parameter Value"
        wLength.Min=0
        wLength.Max=0
    </Control Request>

    <Control Request>
        bmRequestType=0xC0
        bRequest=0x01
        bRequest.desc="Widget - Get Parameter"
        bRequest.text="This requests the device to return a specified parameter.\n\n\
The value is 2 bytes sent in a data packet."
        wIndex.desc="Parameter Number"
        wValue.desc="Parameter Value"
        wLength.Min=0
        wLength.Max=0
    </Control Request>

    <Endpoint>
        ep.code=0x82
        ep.desc="Data Stream from Widget"
        ep.text="Responses in the 'Widget Protocol' are sent by the programmer. \
Typically each response is terminated with a 0x0d 0x0a character pair. Most commands \
are made up of ASCII characters."
    </Endpoint>

</Device>
```

### 3.21.3.2 File Syntax

#### 3.21.3.2.1 Comments

A comment is introduced by the pair of characters '//'. Everything to the right on the same line is part of the comment and ignored.

#### 3.21.3.2.2 Indentation

The example file uses (tabbed) indentation to emphasise the structure of the syntax, but it is not necessary to do this.

#### 3.21.3.2.3 Numbers

Numerical values may be expressed in decimal, or in hexadecimal introduced by the prefix 0x. So 10 and 0x0a represent the same value.

#### 3.21.3.2.4 Strings

String values must be enclosed in double quote marks, e.g. "this is a string".

A long string may span several lines of text as long as:

- each line which is not the end of the string is terminated as the last character with a '\'
- each subsequent line in the string cannot have any white space at the start of the string which is not part of that string
- a line which is part of a string cannot have a comment
- a line which is part of a string cannot be blank

A string may include a line break by including the symbol '\n' at the required point. To have the symbol '\' in the string you must include '\\' at the point required.

See the template file above for examples of long strings.

#### 3.21.3.2.5 <Device> </Device>

The whole file is the description of a device, and so must start with the '<Device>' tag, and end with '</Device>'.

#### 3.21.3.2.6 VID= PID=

Following the '<Device>' tag, the next two lines must define the Vendor ID and the Product ID of the device.

### 3.21.3.2.7 <Control Request> </Control Request>

Each defined control request must be introduced by the '<Control Request>' tag, and ended with '</Control Request>'. Between the tags you should define the parameters of the request by specifying the following:

Parameter	Status	Value type	Purpose
FileType	Mandatory	MQPVEN 1	<b>Must come first.</b> Identifies the filetype and version
bmRequestType=	Mandatory	Number from 0x00 - 0xff	Specifies the Setup packet field which identifies this request
bRequest=	Mandatory	Number from 0x00 - 0xff	Specifies the Setup packet field which identifies this request
bRequest.desc=	Mandatory	String to use as the name of this request. Must be single line and preferably kept short.	Appears in the Control transfer header of the event pane to name this request. Also appears in the Setup transaction header table, in the detail pane, and used as a sub-title in the detail pane for the Control transfer header.
bRequest.text=	Desirable	String to use as the explanation of this request. May be several lines of text.	Appears in the detail pane for the Control transfer header.
wIndex.desc= wIndexH.desc= wIndexL.desc= wValue.desc= wValueH.desc= wValueL.desc=	As required	String to use to name the purpose for this setup packet parameter field. Preferably kept short. Note that e.g. wIndexH is the high byte of wIndex to be used	Appears in the detail pane table when a Setup transaction header is selected in the event pane.

		instead of it when the parameter only occupies a single byte.	
wLength.Min=	Optional	Minimum number of bytes which may be transferred in the Data Stage of the control transfer. Defaults to 0.	Used to validate the number of bytes transferred in the Data Stage.
wLength.Max=	Optional	Maximum number of bytes which may be transferred in the Data Stage of the control transfer. Defaults to 65535.	Used to validate the number of bytes transferred in the Data Stage.

### 3.21.3.2.8 *<Endpoint>* *</Endpoint>*

Each defined endpoint must be introduced by the '*<Endpoint>*' tag, and ended with '*</Endpoint>*'. Between the tags you should define the parameters of the endpoint by specifying the following:

Parameter	Status	Value type	Purpose
ep.code=	Mandatory	Number from 0x01 - 0x0f for OUT endpoints, or from 0x81 - 0x8f for IN endpoints	Specifies the endpoint being described
ep.desc=	Mandatory	String to use as the name of this data transfer. Must be single line and preferably kept short.	Appears in the Data transfer header of the event pane to name this request. Also used as a sub-title in the detail pane for the Control transfer header.
ep.text=	Desirable	String to use as the explanation of this request. May be several lines of text.	Appears in the detail pane for the Data transfer header.

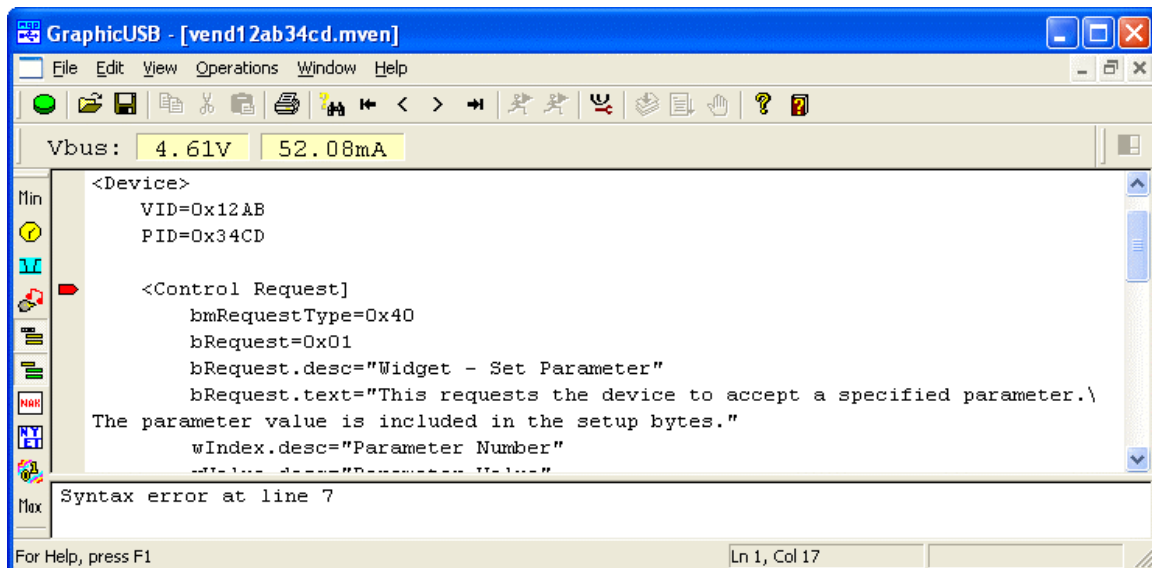
#### 3.21.3.2.9 *Syntax Checking*

It would be irritating to have syntax error messages popping up whenever a syntax error in the user file is encountered, so during display of a capture file the parsing will fail silently, putting up the best interpretation it can. We have provided a separate function to check the legality of the file before attempting to use it.

The syntax of the file can be validated by:  
*Operations.. Validate Vendor File..*

The file must be open in GraphicUSB for this option to be available.

The file will be validated and the results will be shown in the lower output pane.



Double-click on the error message in the output pane, to locate the line in the edit pane (in this case the wrong type of closing bracket was used).

Only the first syntax error found may be shown each time, so run the checker until no errors are flagged.

When satisfied, save the file, using the default filename, and ensuring that it is being saved in the data folder specified for the application.

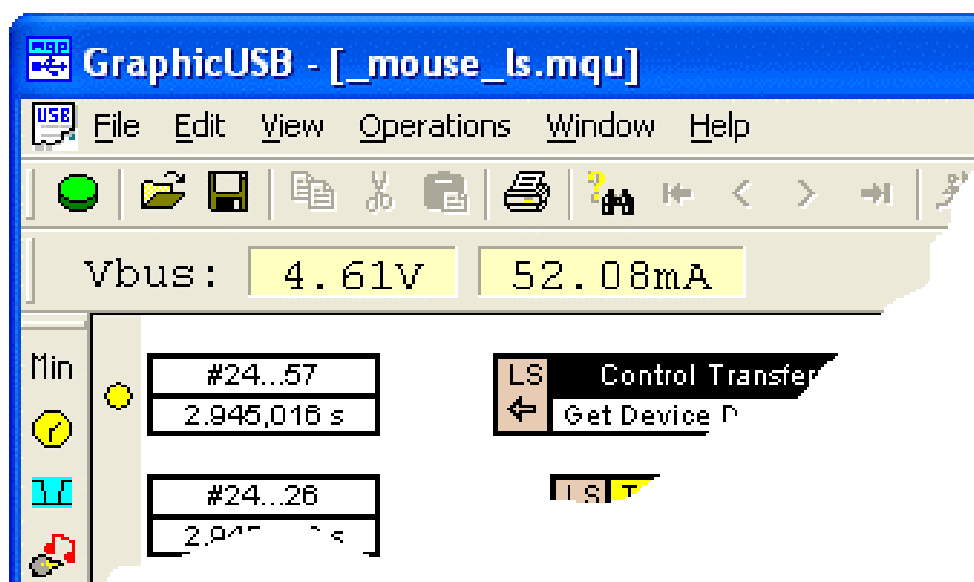


### 3.22 $V_{BUS}$ Current and Voltage Measurement (USB500 AG)

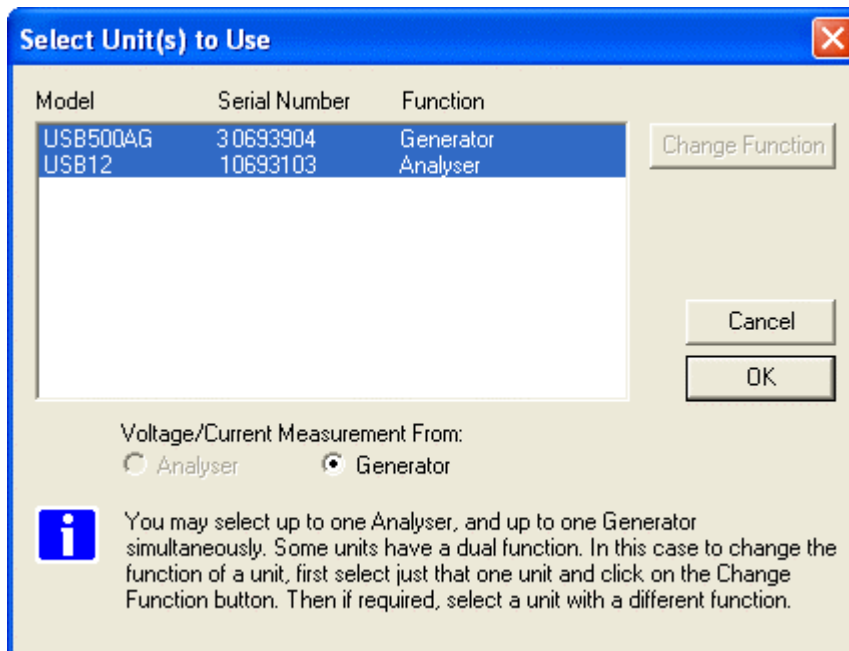
The Packet-Master USB500 AG has built-in  $V_{BUS}$  voltage and current measurement circuitry. This is useful to give an early indication of hardware or software problems related to USB power supply, and to indicate correct response to suspend conditions.

See Technical Data section for accuracy information.

GraphicUSB will continuously display the voltage and current values on its toolbar.



As GraphicUSB may operate two USB500 AG units at the same time, one as a Generator and one as an Analyser, it is necessary to specify which unit should provide the measurement. This is done via the menu item Operations...Select Analyser or Generator...



If there is a possible choice to make, this may be selected under 'Voltage/Current Measurement From'.

### 3.23 Firmware Updates

It is occasionally necessary to modify the firmware within the analyser unit. GraphicUSB has the capability of performing this function in the field.

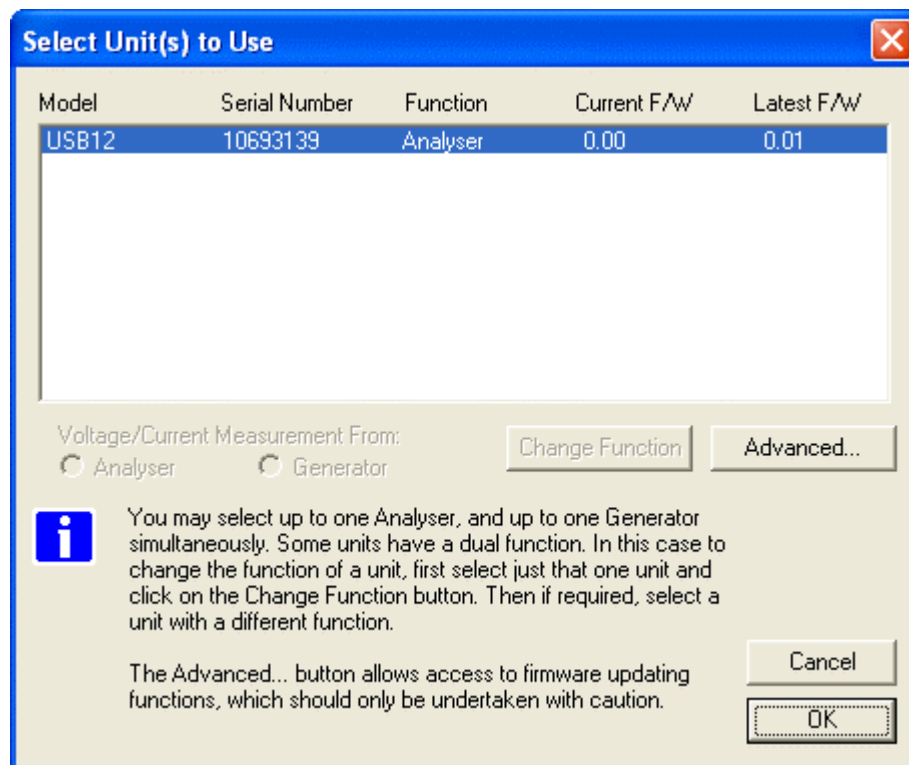
**Caution**

Updating firmware is not without its risks. If the update process is interrupted by a power failure, USB cable disconnection or any other similar problem, then it is possible to leave the analyser unit in a non-working state. So the firmware should only be updated for a valid reason.

The website [www.mqp.com](http://www.mqp.com) contains software revision information, which includes details on firmware revisions, and the reasons for them. Please check there before attempting an update, and contact us beforehand if uncertain.

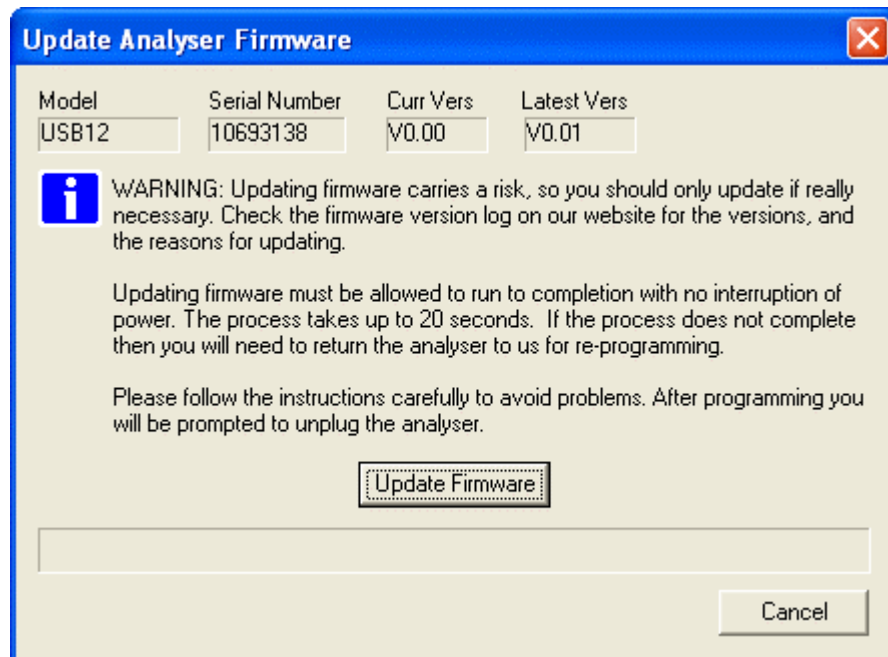
If the process does fail, for one of the above reasons, then you will have to return the analyser to us for re-programming. Please contact us in advance for a returns number in this case.

Firmware updates are controlled from the Operations...Select Analyser or Generator... menu item.

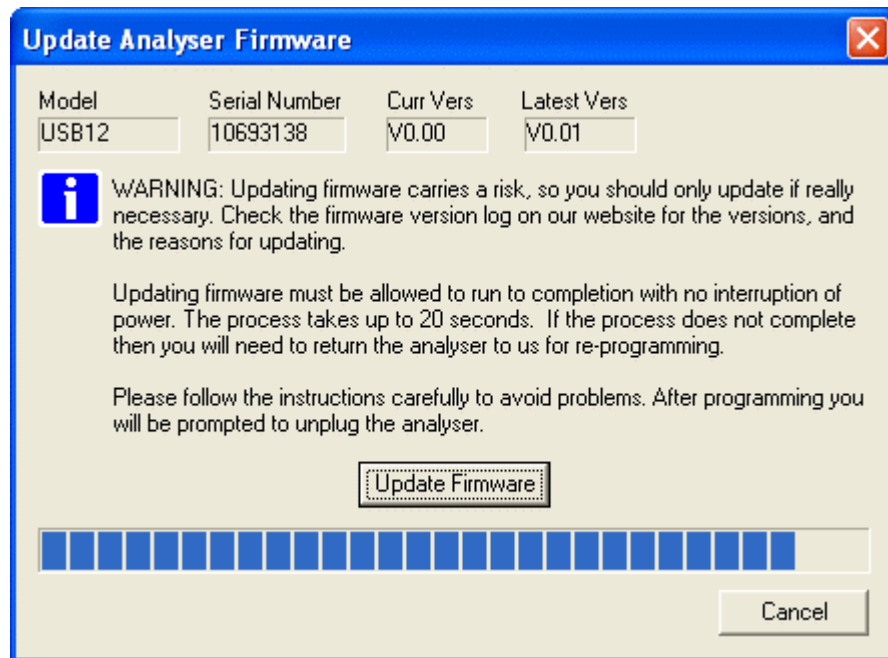


Each connected unit will be displayed. In the example above, the current version (in the unit) is 0.00 and the latest available version is 0.01. Checking on our website will reveal that this update is required from GraphicUSB V3.00 onwards, so we advise proceeding with the update.

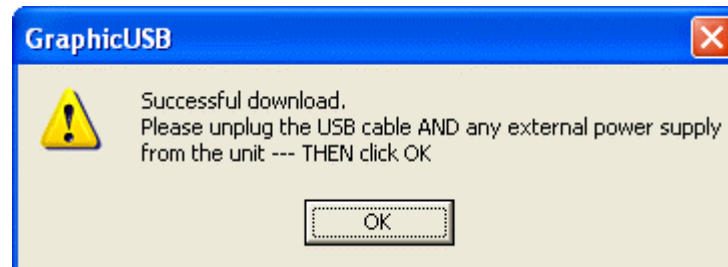
Ensure that only the unit you wish to update is selected, then click on the Advanced... button to see the firmware update dialog:



It is important to follow the instructions very carefully, step by step. First click on the Update Firmware button. The progress of the update process, which takes around 20 seconds, will be indicated on the progress bar.



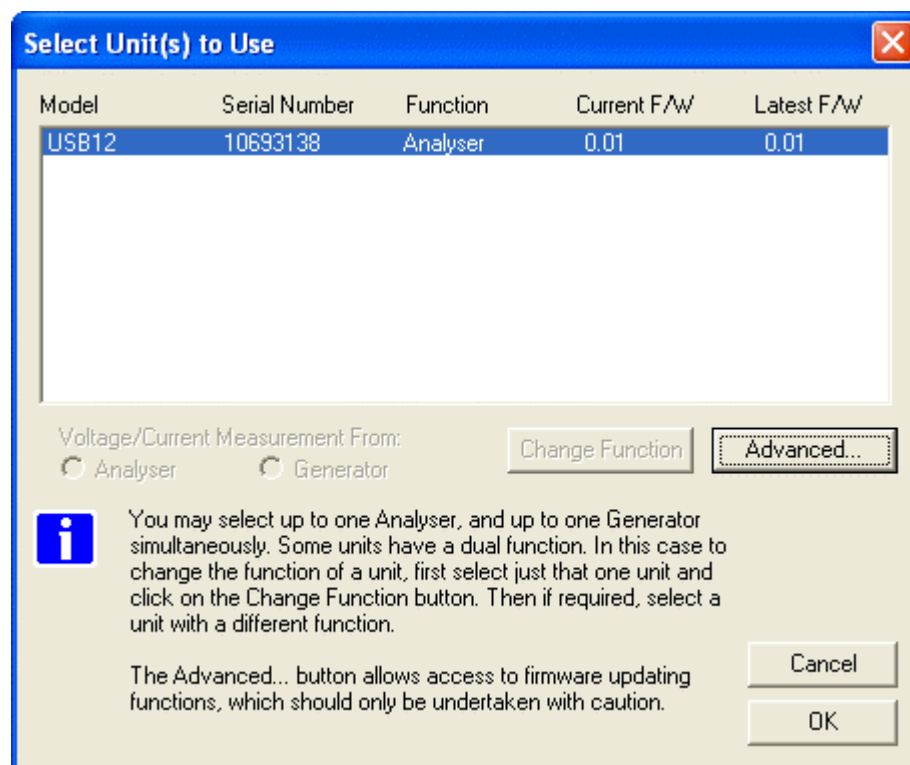
When updating is complete, you will be requested to unplug any connections to the analyser, and **then** click OK:



You will then be asked to reconnect the USB cable, and then click OK:



On clicking OK, the 'Update Analyser Firmware' dialog will close, leaving the Select dialog, which should now show the new firmware version.



## 4 GRAPHICUSB SOFTWARE – GENERATOR



### 4.1 Introduction

The USB500 AG is unique in the Packet-Master series in that it can function as a USB Generator, or Exerciser. In this mode it can emulate host or device hardware, or both alternately when developing for On-The-Go (OTG).

The Generator is controlled by a user-written script, which specifies each event in the interaction, and the expected responses. To simplify writing this script, a capture file from any Packet-Master analyser can be used to export a generator script.

Some feedback is available from the generator, in the form of error messages when something unexpected is encountered. However, in general, a separate USB analyser is required to examine precisely how the device responded to the emulated host commands.

There are a number of choices to be made in respect of generator operation.

### 4.2 Generator Operation – Host / Device Emulation

The generator can operate in host Control Mode or device Control Mode. For On-the-Go devices it can alternate as required.



### ***4.3 Generator Operation - Retry***

When the generator acts as a host, it can be operated in retry or non-retry modes.

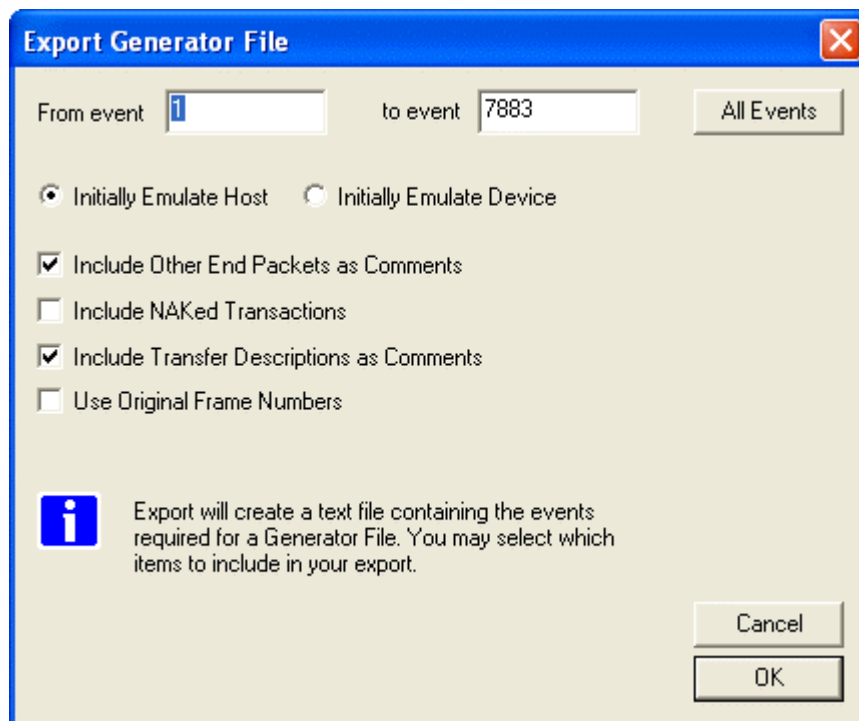
In non-retry mode it can be instructed to output particular packets and wait for particular reply packets, including NAK packets. This can be useful when investigate detailed timings relating to NAKed transactions.

In retry mode, waiting for NAK packets is not specified in the script. Instead the finally expected packet (ACK or DATAx) is waited for, and if the generator receives a NAK packet then it will automatically retry the transaction. This mode is very useful because it relieves you from specifying the exact timing and quantity of NAKed operations, which with some devices may vary from occasion to occasion.

### ***4.4 Creating a Generator Script from a Capture file.***

The quickest way to produce a valid generator script is to start from a Capture file (\*.mqu) previously created by a Packet-Master analyser, and which represents the sequence of operations which you want to test. It might for example have been captured from a working example of a device whose performance you wish to emulate in a device under development.

With the capture file displayed in GraphicUSB, select the menu item File...Create Generator File...



You have the option here of selecting the range of events which you want to turn into a generator script file, or simply (by default) select all events from the capture file.

Next you should specify which side of the link you wish to emulate, host or device.

There is an option to include, or not, the events which occurred at the other end of the link as comments, which may be of assistance in showing the expected response. These comments will not affect the operation of the script.

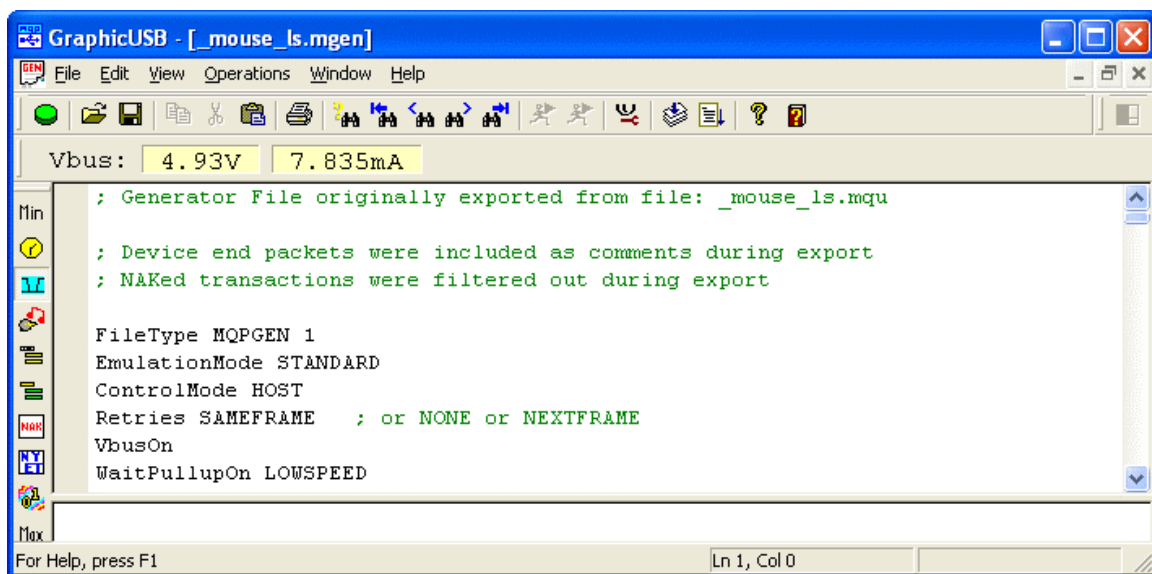
You need to decide whether to include NAKed transactions in the output. This is appropriate if you are trying to simulate an exact exchange of packets including NAKs. It is usually more appropriate to exclude the NAKed transactions, and use the Retries SAMEFRAME or Retries NEXTFRAME command. This allows emulation of the host to automatically retry any NAKed transaction.


If including NAKed transactions it is appropriate to use Retries NONE in your script. The automatically created script will contain the appropriate setting for Retries.

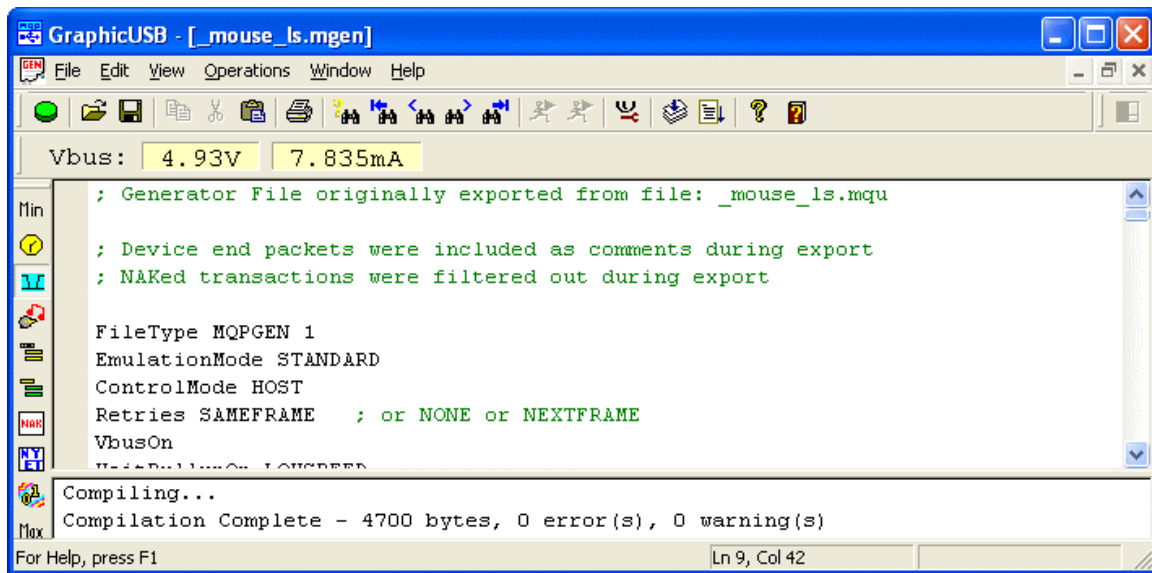
'Include Transfer Descriptions as Comments' is selected by default. This is extremely useful in interpreting your script later on.

Finally you have the option to use the original frame number from the SOF packets, or (by default) to generate them automatically starting at 0.

When happy with your selections, click on 'OK' and your generator script file will appear in its own edit window. Below it will be an empty output pane. This is for compiler output information.



It is a good idea to save this file at this point. You may now try compiling the file using menu item File...Compile, or click on the 'Compile' icon in the toolbar .

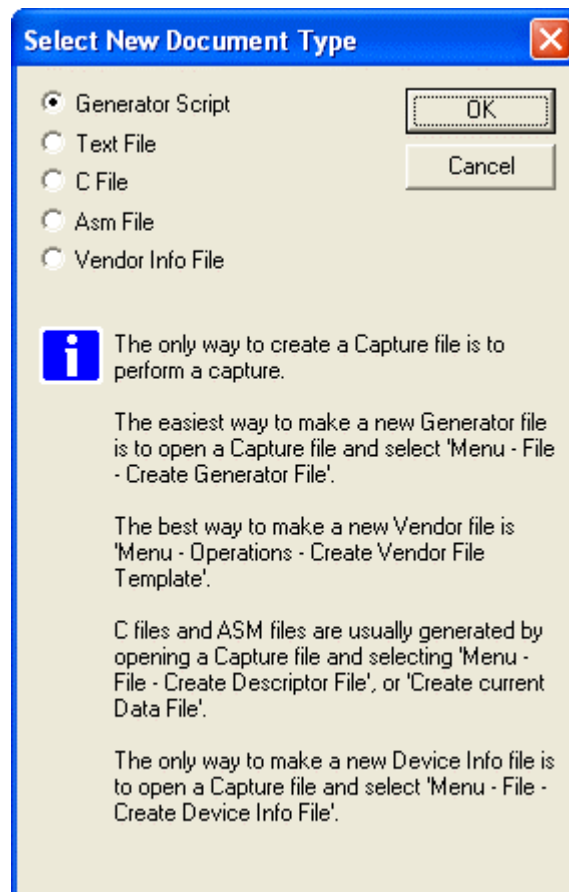


Edit your file as you require, checking its validity with frequent compilations. Then you are ready to run your generator script.

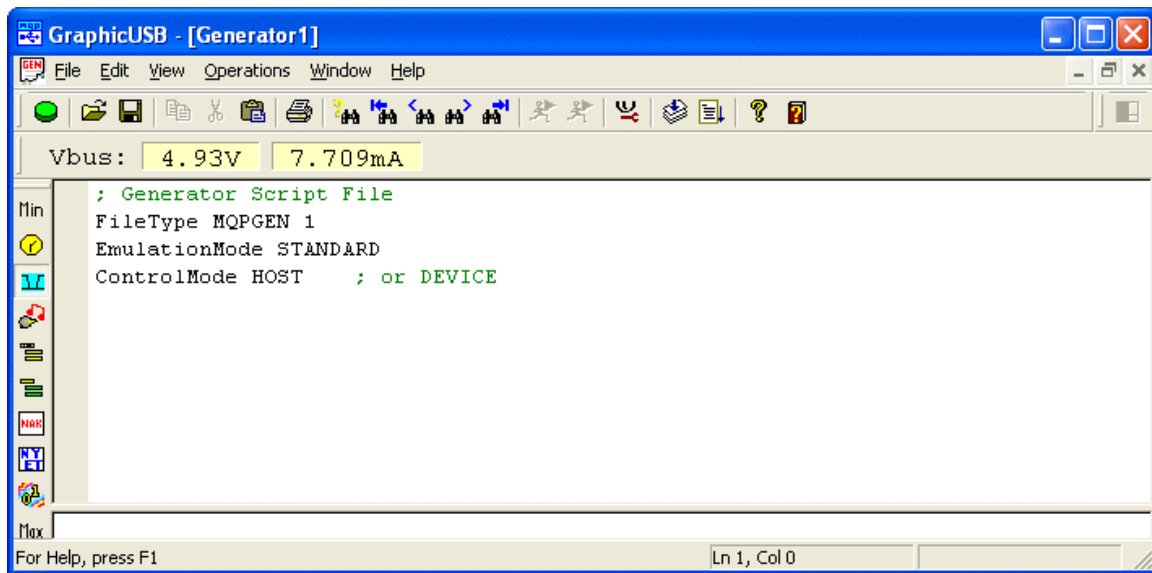
### 4.5 Creating a Generator Script from Scratch.

An alternative to creating a generator file from a capture file is to start with a blank page and insert the script commands yourself. This is made simpler by the availability of two command insertion dialogs.

Before entering commands you need to make a new blank script file. Select menu item File...New...

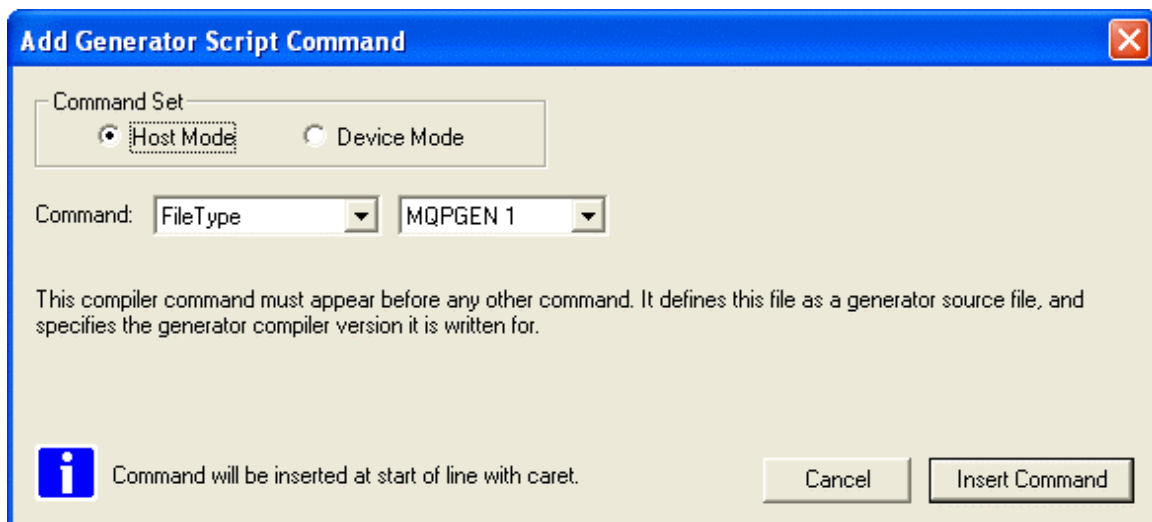


Select 'Generator Script' and click on 'OK'. A new blank generator script file will open.



Notice that the document has a separate output pane beneath for compiler information to be displayed in.

The first of the two command insertion dialogs is 'Insert Command...'. It is found in the Edit menu, or by typing Ctrl+I.

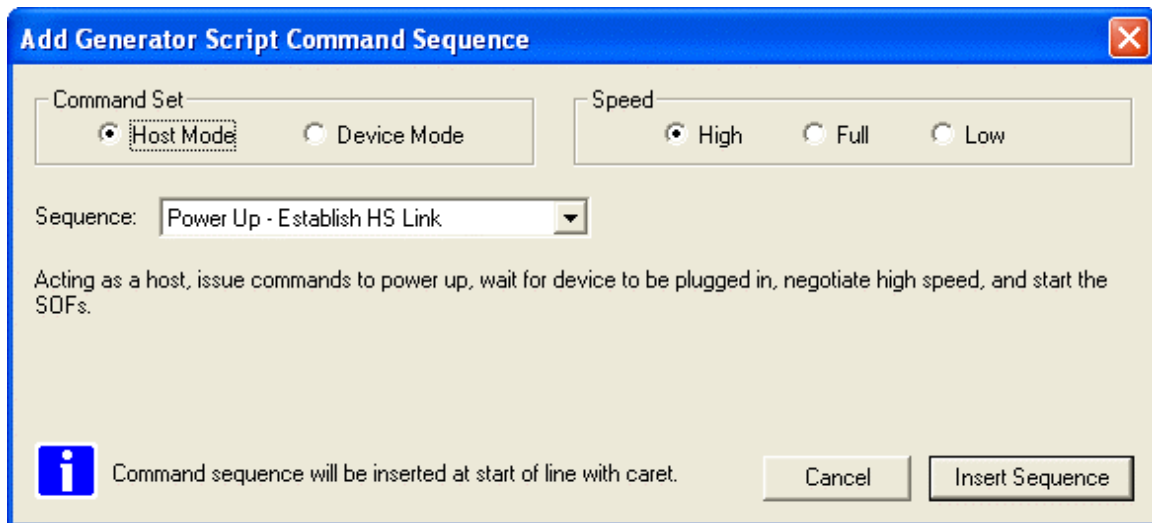


This dialog will insert a single, correctly constructed, command of your choice in front of the line containing the caret. It has a variable

number of selection and edit boxes, depending on the command you select. The procedure is:

- Before entering the dialog, ensure that your caret is located at the start of the line you want to insert the command in front of.
- First select the command set, host or device, depending on which you are currently emulating. The command box will be filled with only those commands appropriate to the mode.
- Then select the command required from the command selection box. The purpose of the command is explained underneath.
- Now make appropriate selections in the other boxes to the right of the command. When happy with your selection, click on 'Insert Command'.

The second way of inserting commands is by using the 'Insert Sequence...'. This is also found in the Edit menu, or by typing Ctrl+Shift+I.



This is a more powerful approach as it allows you to enter all the required commands for a particular operation in one go. Typical examples are a complete 'Power-up' or 'Get Device Descriptor' sequence. The procedure here is:

- Before entering the dialog, ensure that your caret is located at the start of the line you want to insert the sequence in front of.
- First select the command set, host or device, depending on which you are currently emulating.
- Select the required link speed.
- The command box will be filled with only those command sequences appropriate to the mode and the link speed you have selected.
- Select the sequence you want to insert; the purpose of the sequence is explained below the selection box.
- Click on Insert Sequence.



Using this system you can build up a script containing correct sequences of commands in no time. You can then edit the commands in the script as you require, taking into account your special needs. Then you are ready to run your generator script.

## 4.6 Generator Script Language Commands

Command	Remarks
FileType MQPGEN 2	<p>Must come first in script.</p> <p>Identifies the file type and version. Version must be 2.</p>
EmulationMode STANDARD EmulationMode ADEVICE EmulationMode BDEVICE	<p>This must come after FileType and before any other command.</p> <p>STANDARD is used for standard USB, while ADEVICE or BDEVICE are used for OTG devices as appropriate.</p>
ControlMode HOST ControlMode DEVICE	<p>This determines whether the generator should emulate a host or a device. This command must appear at least once in the file, immediately following EmulationMode.</p> <p>ControlMode HOST results in a 15k pull-down resistor being applied to each line.</p> <p>ControlMode DEVICE results in the data lines being floated, which they will continue to do until the PullupOn command is encountered.</p> <p><b><u>On-the-Go Usage</u></b></p> <p>This command can also be used during an OTG emulation sequence, as part of a Host Negotiation Protocol (HNP) exchange of control.</p> <p><u>ControlMode HOST</u> must only appear:</p> <ul style="list-style-type: none"> <li>when acting as a device</li> </ul>

	<ul style="list-style-type: none"> <li>• after being suspended, and</li> <li>• after having floated the data bus, using PullupOff.</li> </ul> <p>ControlMode DEVICE must only appear:</p> <ul style="list-style-type: none"> <li>• when acting as a host</li> <li>• after a SuspendHnp command.</li> </ul> <p>See sample OTG script fragments in a following section for further information.</p>
Retries NONE Retries SAMEFRAME Retries NEXTFRAME	<p><u>Host Only</u></p> <p>The Retries command determines the method used to do transaction retries, after a packet with an expected PID fails to arrive.</p> <p>If not included in the script, the compiler will assume 'NONE'; i.e. that retries should not be attempted. In this case execution of the script will hang at this point, with a suitable error message.</p> <p>The 'SAMEFRAME' setting will attempt up to four retries per frame.</p> <p>The 'NEXTFRAME' setting will attempt one retry per frame.</p>
Reset <n>	<p><u>Host Only</u></p> <p>Apply SE0 for &lt;n&gt; us.</p> <p>A value of 0 for &lt;n&gt; is an instruction only to start applying the reset condition. This form is used when a high speed handshake chirp sequence is involved.</p>

Suspend <n>	<p><u>Host Only</u></p> <p>Idles for &lt;n&gt; ms. This means it stops SOFs being automatically sent. It then waits for &lt;n&gt; ms before proceeding to the next command.</p> <p>If in high speed, it also removes the high speed termination, and connects the 15k pull-down resistors, but the generator remembers that it was in high speed, so that it can automatically restore the high speed termination, if the next command is a Resume.</p>
SuspendHnp	<p><u>OTG Host Only</u></p> <p>This has the effect of stopping sending SOFs, If in high speed, it also removes the high-speed termination, and connects the 15k pull-down resistors.</p> <p>Having done this, it proceeds immediately to the next command.</p> <p>The command should be used at the start of an exchange of control using Host Negotiation Protocol.</p>
WaitResume	<p><u>Host</u></p> <p>Use before Resume command if device has remote wakeup.</p> <p><u>Device</u></p> <p>Waits for host to perform a resume.</p>
Resume n	<p><u>Host or Device</u></p> <p>Applies K for n ms followed by the appropriate condition for the speed and whether it is a host or a device.</p>

	<p>A host resume should have a period of at least 20 ms.</p> <p>A device resume should have a period of 1 to 15 ms,</p>
WaitSuspend	<p><u>Device Only</u></p> <p>Wait for activity to cease for 3 ms.</p> <p>If originally in high speed, it then removes the high speed termination, and connects the 1.5k full speed termination resistor on D+.</p> <p>If the next command is Resume, or WaitResume, the original high speed termination will automatically be re-connected.</p> <p>If the suspend occurs at the start of a transfer of control, which is part of the Host Negotiation Protocol, then the next command should be</p> <p>‘PullupOn FULLSPEED’</p> <p>which has the effect of canceling the need to automatically return to high speed termination.</p>
VbusOn VbusOff	<p><u>Host</u></p> <p>Used for power control.</p> <p><u>Device</u></p> <p>Used for SRP <math>V_{BUS}</math> pulsing, by OTG B-device.</p>
WaitVbusOn WaitVbusOff	<p><u>Host</u></p> <p>Used by OTG A-device to detect SRP <math>V_{BUS}</math> pulsing.</p> <p><u>Device</u></p> <p>Used to detect when <math>V_{BUS}</math> is</p>

	available. The PullupOn command may then be used.
WaitReset	<u>Device Only</u> Waits for Reset persistence check period of 2.5us for full speed or low speed, and 3.4ms for high speed.
PullupOn LOWSPEED PullupOn FULLSPEED PullupOff	<u>Device Only</u> Connects or disconnects 1.5k pullup resistor. <b>PullupOff</b> (usually only used for OTG) results in the data lines being floated, which they will continue to do until the <b>PullupOn</b> command is encountered. When used after <b>WaitSuspend</b> , <b>PullupOn FULLSPEED</b> also has the effect of canceling the need to automatically return to high speed termination, and is used as part of the Host Negotiation protocol.
WaitChirp	<u>Host or Device</u> If host, wait for end of K chirp. If device, wait for end of KJ pairs. If a device, it switches to high speed terminations after detecting 3 KJ pairs.
SendChirp <n> <k> <j>	<u>Host or Device</u> This sends a series of <n> KJ chirp pairs, where each K chirp lasts <k> $\mu$ s, and each J chirp lasts <j> $\mu$ s. If emulating a device chirp, specify

	<n> as 1 and <j> as 0.
WaitPullupOn LOWSPEED WaitPullupOn FULLSPEED WaitPullupOff	<u>Host Only</u> If the wrong polarity is detected, execution is halted with an error code.
SOFOn [n]	<u>Host Only</u> n is number of first frame. Start to transmit regular SOFs or Keep Alives as appropriate. If re-used it starts next SOF immediately, without waiting for correct time. If n is omitted, frame number of next SOF is not reset. The default starting frame number is 0 if this is first SOFOn. Note that Suspend is one way to turn SOFs off. There are other commands which will have the same result, such as Reset, VbusOn, VbusOff.
SendPacketLs ( ) SendPacketFs ( ) SendPacketHs ( )	<u>Host or Device</u> Sends a packet containing the data bytes specified within the brackets. The PID byte and any CRC bytes must also be specified. See the predefined value table below for special values: <ul style="list-style-type: none"> <li>• ACK, NAK, DATA0 etc.</li> <li>• CRC16L</li> <li>• CRC16H</li> <li>• AD_EP_C5</li> </ul>

	<ul style="list-style-type: none"> <li>• AD_SC_P_S_E_ET_C5 which are of great assistance in defining packet data.</li> </ul> <p>If the packet cannot be sent given the current link speed, execution is halted with an error code.</p> <p>If a LS packet is sent on a FS link, the preamble will automatically be sent first.</p>
Idle <n>	<p><u>Host or Device</u></p> <p>Idle for specified number of 60MHz clock cycles.</p> <p>Idle is also used for waiting in other states than the standard USB idle state, if required. Examples are after WaitVbusOn, or before PullupOn.</p> <p>If the time specified overlaps the next SOF start time, then delay the next SOF.</p> <p>Note that after sending a packet (or waiting for a packet), the Idle time specified starts after the minimum inter-packet delay time. So a command of Idle 0 is valid between two packets.</p>
SOFs <n> [<m>]	<p><u>Host Only</u></p> <p>Do nothing while the specified number &lt;n&gt; of SOFs is automatically generated.</p> <p>If &lt;m&gt; is specified then the next frame number will be forced to &lt;m&gt;, otherwise the next automatically generated frame number is used.</p>



WaitPacketLs (<pid>) WaitPacketFs (<pid>) WaitPacketHs (<pid>)	<p><u>Host or Device</u></p> <p>Wait till the end of the next packet received, and compare the PID with the one specified. If no match – the generator stops and displays a warning message. If device – ignore SOFs.</p> <p>If you wish to wait a number of alternative PIDs, then the PIDs in question are listed between ‘ ’ (or) symbols, e.g:</p> <p>WaitPacketLs (DATA0   DATA1)</p> <p>If RETRIES are enabled, then if NAK is received instead of the awaited PID, the transaction will automatically be retried from the associated SETUP IN or OUT.</p> <p>In appropriate cases (only) this also applies to received NYET packets.</p>
Unplug	<p><u>Device Only</u></p> <p>Simulates unplugging by disconnecting any termination conditions.</p>
WaitUnplug	<p><u>Host Only</u></p> <p>Waits for device to be unplugged.</p> <p>If the link is full speed this command will stop the automatic sending of SOFs, and if low speed stops the automatic sending of Keep Alives.</p> <p>At high speed, SOFs are not stopped, as they are used as the mechanism for detecting unplugging.</p>

SetCount0 <n> SetCount1 <n> SetCount2 <n> SetCount3 <n>	<u>Host or Device</u>  Counters are available to perform loops. Each counter may be set to a value of 1 to 65535, and used in conjunction with the DJNZCountn commands.
SetCount <n>	<u>Host or Device</u>  This command is now deprecated, but is still valid, having exactly the same meaning as SetCount0 <n>
DJNZCount0 <label> DJNZCount1 <label> DJNZCount2 <label> DJNZCount3 <label>	<u>Host or Device</u>  Decrement the count number, and if it is not zero, goto the specified label.
DJNZ <label>	<u>Host or Device</u>  This command is now deprecated, but is still valid, having exactly the same meaning as DJNZCount0 <label>
Goto <label>	<u>Host or Device</u>  This is an unconditional Goto to allow the last part of a script to be continuously repeated.
Halt	<u>Host or Device</u>  An option command to terminate execution of the script. This can be useful to limit the extent of the script during script development.

## 4.7 Generator Script Language Pre-defined Values

Parameter	Usage	Meaning
STANDARD ADEVICE BDEVICE	EmulationMode	Mandatory setting. The ADEVICE and BDEVICE settings are for On-the-Go (OTG) devices.
HOST	ControlMode	The generator should behave as a host.
DEVICE	ControlMode	The generator should behave as a device.
LOWSPEED	PullupOn WaitPullupOn	The resistor on D- is specified.
FULLSPEED	PullupOn WaitPullupOn	The resistor on D+ is specified.
NONE	Retries	Automatic retries of NAKed transactions are not enabled
SAMEFRAME	Retries	Automatic retries of NAKed transactions will be retried a number of times in the current and later frames.
NEXTFRAME	Retries	Automatic retries of NAKed transactions will be retried only once in the next and later frames.
ACK NAK STALL NYET DATA0 DATA1	SendPacketLs SendPacketFs SendPacketHs	Specifies the binary value of the PID in question. This includes the PID invert (check) bits, ready for transmission in a packet.

DATA2 MDATA IN OUT SETUP ERR SPLIT PING		
ACK NAK STALL NYET DATA0 DATA1 DATA2 MDATA IN OUT SETUP ERR SPLIT PING	WaitPacketLs WaitPacketFs WaitPacketHs	<p>Specifies the particular PID to wait for. Multiple alternative PIDs may be specified, separated by ‘ ’ symbols.</p>
NORETRY	WaitPacketLs WaitPacketFs WaitPacketHs	<p>An optional parameter in a WaitPacketLs / WaitPacketFs / WaitPacketHs command. Must appear between the command keyword and the PID or PIDs.</p> <p>It prevents retries for this one command, even though retries are enabled.</p>

CRC16L CRC16H	SendPacketLs SendPacketFs SendPacketHs	<p>These specify the two bytes of a 16 bit CRC. They are included in data packets to allow the compiler to insert the correct crc values rather than the user having to calculate them. They are only guaranteed to contain the correct byte values if they are included at the end of the packet, and in the correct order: CRC16L CRC16H.</p>
AD_EP_C5 (<a> <e>)	SendPacketLs SendPacketFs SendPacketHs	<p>This special value represents the two bytes required in a token packet (IN OUT SETUP or PING). The tow values in parenthesis are:</p> <p>&lt;a&gt; = the address</p> <p>&lt;e&gt; = the endpoint</p> <p>The two bytes added into the packet will include those two values correctly positioned, plus the 5 bit crc.</p> <p>E.g.</p> <pre>SendPacketHs ( IN AD_EP_C5(0 0) )</pre>
AD_SC_P_S_E_ET_C5 (<a> <sc> <p> <s> <e> <et>)	SendPacketLs SendPacketFs SendPacketHs	<p>This is similar to the AD_EP_C5 value but relates instead to the bytes required in a SPLIT packet.</p> <p>&lt;a&gt; = the address</p> <p>&lt;sc&gt; = start/complete</p>

		<p>&lt;p&gt; = port</p> <p>&lt;s&gt; = speed</p> <p>&lt;e&gt; = end</p> <p>&lt;et&gt; = endpoint type</p> <p>The 5 bit crc is automatically calculated and positioned correctly within the 3 bytes which are inserted into the packet.</p>
--	--	--

## 4.8 Generator Script Language Syntax Rules

### 4.8.1 Command Sequence

The first three commands must appear as follows, in this order:

FileType MQPGEN 1

EmulationMode STANDARD ; (or ADEVICE or BDEVICE)

ControlMode HOST ; (or DEVICE)

...

### 4.8.2 Case Sensitivity

All commands and parameters are case insensitive. Thus

WaitPacketHs is the same as WAITPACKETHS.

DATA0 is the same as daTa0.

We use mixed case for commands and all upper case for pre-defined values, for clarity.

### 4.8.3 Command Lines

All commands must start on a new line. Commands:

SendPacketLs

SendPacketFs

SendPacketHs

may take up as many lines as are necessary to specify all the data contained in the packet to be sent.

Blank lines are allowed.

#### 4.8.4 Labels

A label is defined as a sequence of alphanumeric characters finishing with a ':' It must appear on its own line of text, and should not match any keywords used by the compiler. A maximum of 16 labels is allowed in a script. A label is used as the target of a GOTO command or a DJNZCount*n* command.

An example of a label is:

```
Label12:
```

#### 4.8.5 Comments

Comments are introduced by a ';' character and continue till the end of the current line. A comment may appear to the right of any command or part command. e.g.

```
SendPacketLs (DATA0      ; this is a comment
              0x80 0x06 0x00 0x01 0x00 0x00 0x40 0x00
              CRC16L CRC16H)
```

Comments are completely ignored by the compiler.

#### 4.8.6 Tabs

Tab characters may be used to make the script tidier, a tab will be interpreted as white space.

#### 4.8.7 Data Values

Data values may be expressed in decimal or hexadecimal, or by a pre-defined value.

A hexadecimal number is prefixed with '0x'.

e.g.

```
45, 0x2d and SETUP
```

all represent the same value.

#### **4.8.8 Execution**

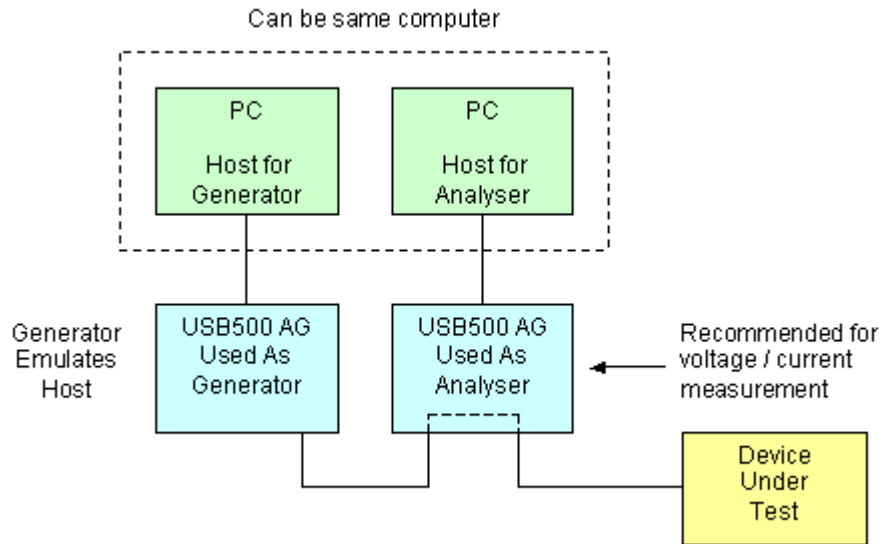
The script instructions are executed in turn starting at the beginning and continuing till the end.

In the case of a user specified loop, execution will continue till the user halts it from the application.

If a problem is encountered, execution halts and an error message is displayed.



## 4.9 Test Configurations



This is how to connect the generator when emulating a host. The analyser shown is marked USB500 AG, but can in fact be any suitable analyser. The PC which is hosting the generator can be the same as that hosting the analyser, provided both are from the Packet-Master series.

If the analyser in question is from a third party manufacturer, then you should be cautious about performance when using the same PC; we would recommend separate PCs in this case.

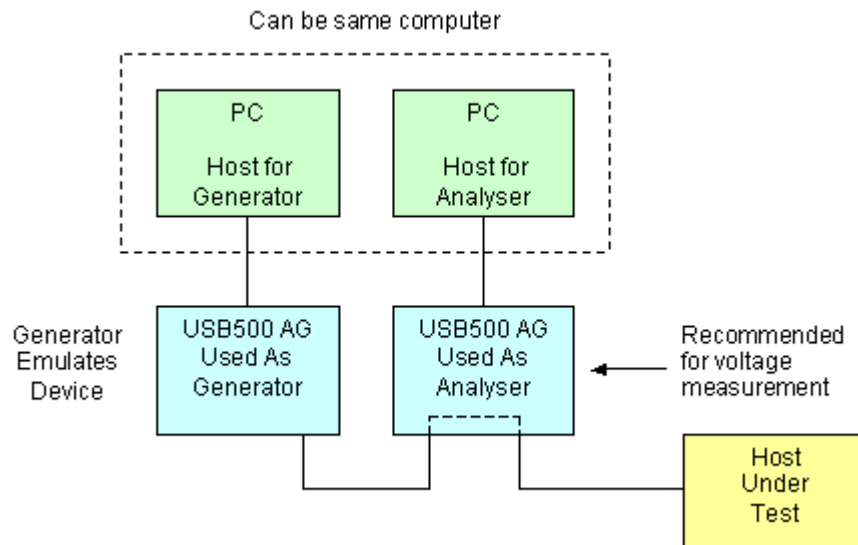
Packet-Master USB500 AG has a built-in  $V_{BUS}$  measuring option.

If two USB500 AG units are used, we recommend that the  $V_{BUS}$  voltage and current measurements are specified as being read by the unit used as an analyser. This eliminates any current drawn by downstream analyser units being included in the measurement.

If a USB500 AG is sometimes used as an analyser and sometimes as a generator, there is the possibility that the host lead is left plugged into the front of the generator, which is a bad idea as both the connected host and the generator will be attempting to source  $V_{BUS}$ . To prevent this occurring, the generator senses whether an external  $V_{BUS}$  is connected when it executes a ControlMode HOST



command, and will alternately flash the 'Generator' and 'Host' indicators, until the host connection is removed.




A similar arrangement is used when the generator is to emulate a device.

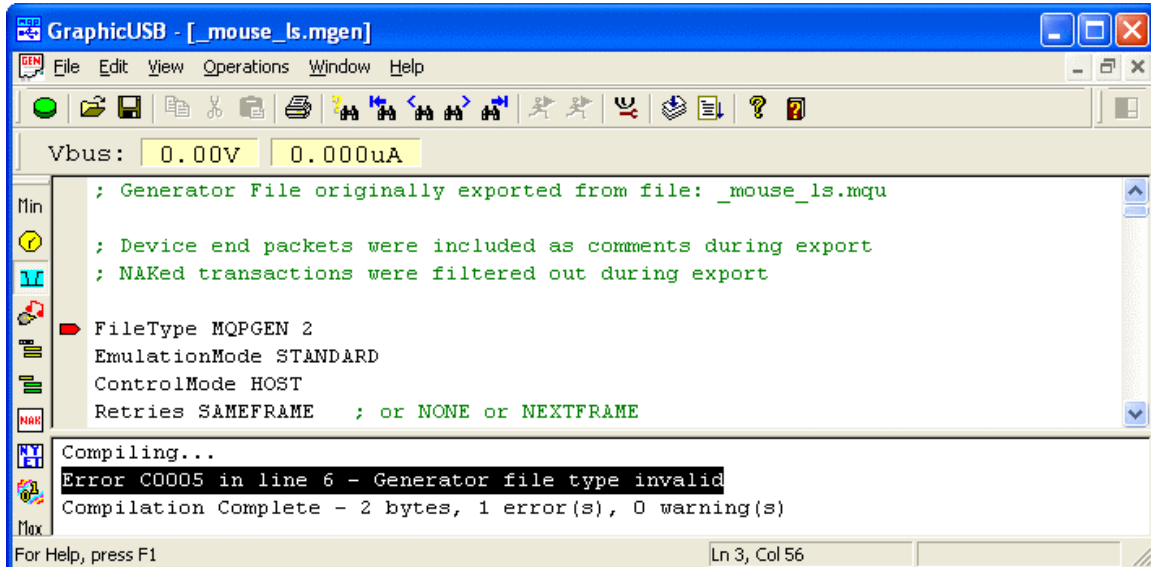
If you are using the USB500 AG built-in  $V_{BUS}$  measuring option, note that only the voltage is meaningful as the current would be that drawn by the generator, which is in any case an insignificantly small amount.

The Host under test could of course be a PC. We do not recommend using the same PC as is hosting either or both, generator or analyser. If you attempt this, then at the very least, use a separate USB controller card for the host under test.

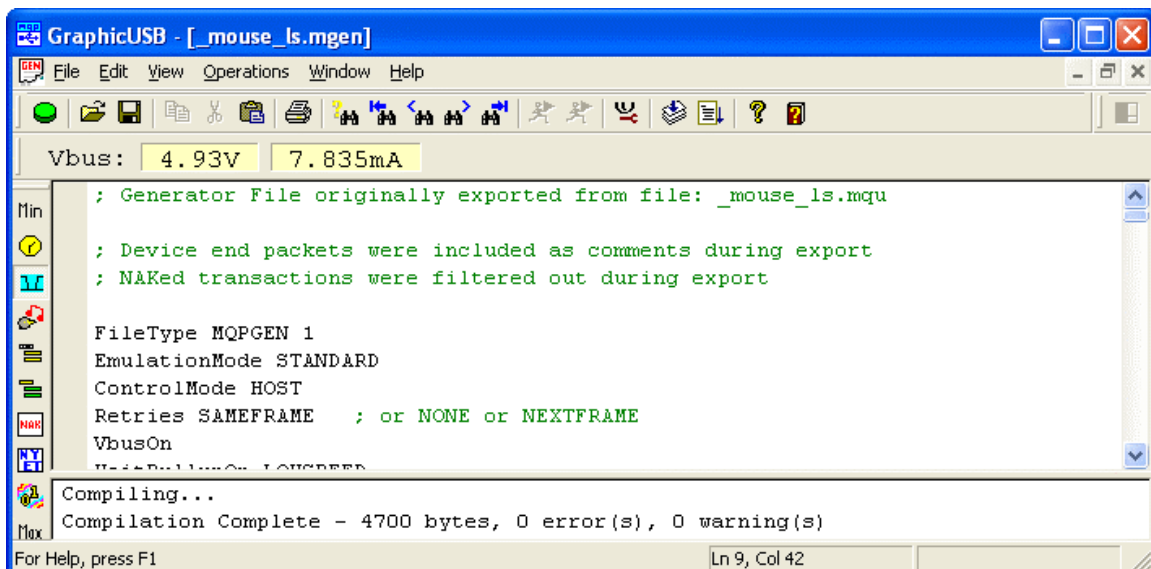
You will have noticed the similarity in the configuration for emulating a host, or emulating a device. Clearly the same arrangement therefore applies for testing On-the-Go devices.

## 4.10 Compiling the Generator Script

Compile the file using menu item File...Compile, or click on the 'Compile' icon in the toolbar . If there is an error, it will be announced in the lower pane. Double click on the error message to point at the line in the script containing the error. Alternatively press the F4 key to highlight the errors one at a time.



Correct any errors until the compilation is successful.




## 4.11 Running the Generator Script

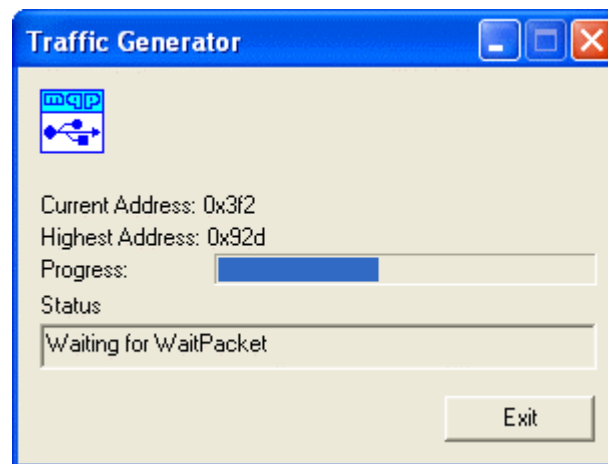
Once you have successfully compiled the generator script with no errors, it is time to run it in the generator.

### 4.11.1 Different Computer Hosting Analyser

Before running the generator script, ensure that you have started capturing with the chosen analyser.


With the compiled script showing in the application window, select menu item File...Run..., or click on the 'Run' icon in the toolbar . A dialog will appear showing progress of the emulation. The Status window will describe the condition of the emulation, which may have halted owing to an error, or may be waiting for an event which does not occur. Typically the run will be over very quickly. The script may have hung up waiting for a particular event, or may have completed. After clicking on Exit, you will see the script displayed, showing the location reached in the emulation.

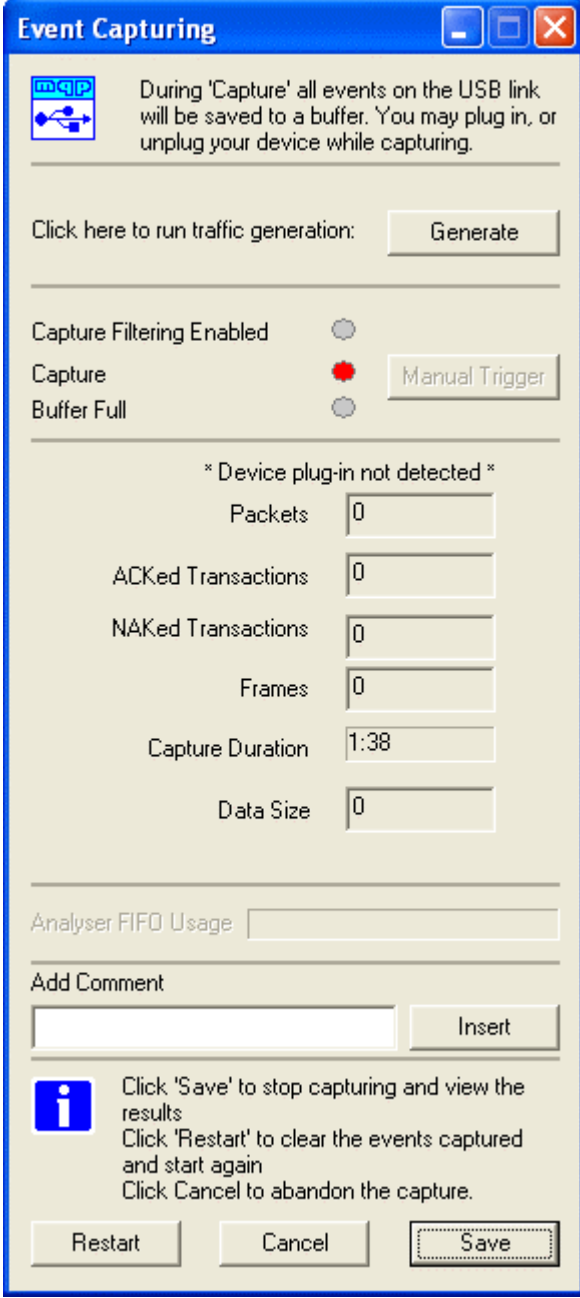
After a run, you should stop the capture on the associated analyser.



#### 4.11.2 Same Computer Hosting Analyser

It is possible for GraphicUSB to control both generator and analyser from the same computer. You should proceed as described above for separate computers.

This time however before starting to run the script, click on the Capture icon . The capture progress dialog will open.



The 'Event Capturing' dialog box is a standard Windows-style window with a blue title bar. It contains a status bar at the top with a message and a 'Generate' button. Below this are checkboxes for 'Capture Filtering Enabled', 'Capture', and 'Buffer Full', with a 'Manual Trigger' button. A section titled '\* Device plug-in not detected \*' contains several input fields for 'Packets', 'ACKed Transactions', 'NAKed Transactions', 'Frames', 'Capture Duration', and 'Data Size'. At the bottom, there is an 'Analyser FIFO Usage' field, an 'Add Comment' section with an 'Insert' button, and an information icon with instructions. At the very bottom are 'Restart', 'Cancel', and 'Save' buttons.

**Event Capturing**

During 'Capture' all events on the USB link will be saved to a buffer. You may plug in, or unplug your device while capturing.

Click here to run traffic generation:

Capture Filtering Enabled ☐

Capture ☒

Buffer Full ☐

\* Device plug-in not detected \*

Packets

ACKed Transactions

NAKed Transactions


Frames

Capture Duration

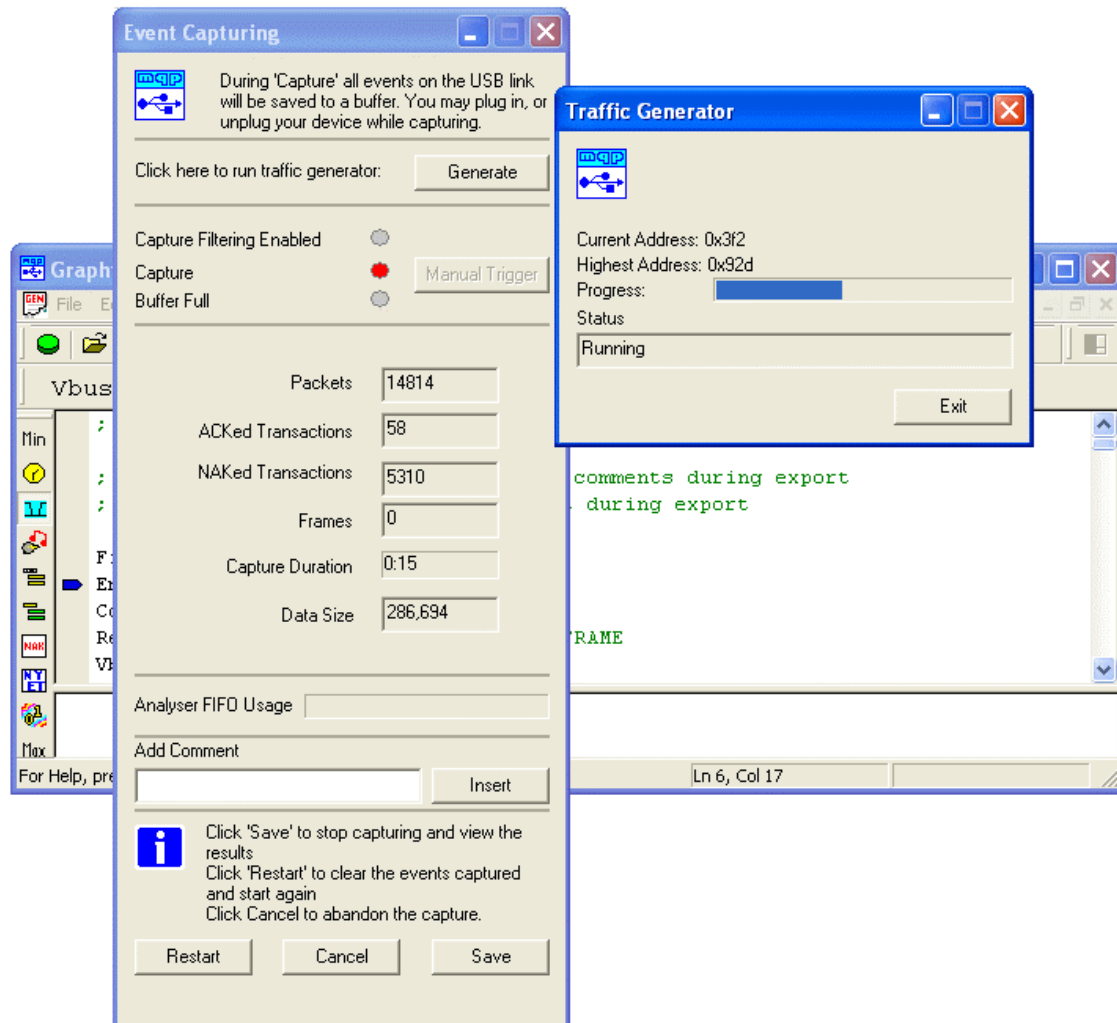
Data Size

Analyser FIFO Usage

Add Comment

 Click 'Save' to stop capturing and view the results  
Click 'Restart' to clear the events captured and start again  
Click Cancel to abandon the capture.

The capture dialog has its own button to start the generator. Click this and the script run will be started. The script dialog will arrange itself on the screen so that both dialogs are visible.



When the generator run has finished, click on 'Exit', and the generator dialog will close, allowing you to use the capture dialog to save the resulting capture.

## 4.12 Generator Error Messages

Generator error messages are usually self-explanatory.

A message can accompany a halt in execution of your script, perhaps because a particular command is illegal in the current link state.

It is also possible to receive a message that a particular event is being waited for. This would not normally result in halting execution, merely pausing it while the wait takes place, and in the case that the event in question finally occurs, execution will continue.



## **4.13 Generator Script Suggestions**

A generator is not the same thing as a real host. It follows a sequence of commands with the minimum of interaction with the driven device. The main object is to stimulate a device in a fairly precise and simple way, in order to capture the response.

In order to allow flexible operation, the generator has the capability of automatic retries on NAKs, and where appropriate, also NYETs.

It also automatically outputs, accurately timed SOFs or Keep Alives without having to define this specifically in the script timing.

There are certain limitations to a generator, which it is as well to understand. The generator script makes the assumption that the device is going to behave in a certain way in response to each request.

Let us take a USB Mouse as an example device. In normal operation, after enumeration, a real host will send an IN packet at regular intervals (perhaps every 8 ms), and will receive either a NAK or a DATAx packet in return, depending on whether the mouse was moved or clicked, or left alone.

To reproduce this action exactly from a script, the mouse would have to be moved in exactly the same way each time the script was run, which is clearly unlikely.

If we start by doing a capture of the enumeration and for a few seconds after, we can create a generator script by selecting from the menu bar:

File... Create Generator Script...

In the 'Export generator Script File' dialog, we will typically not select the option: 'Include NAKed Transactions'. The result of this is that all the NAKed interrupt transfers will be missing from the script, and that the Retries SAMEFRAME command will be included in the script.

The result of running the script will be that the INs that were originally NAKed will not be generated at all, and that the first IN

transaction which is generated will be retried about 4 times per frame until a non-NAKed response is received.

So the generator will proceed through its script, as long as you move the mouse around, but the actual frequency of the IN transactions will sometimes be multiples of the device specified blinterval time, and sometimes 4 times per frame. (This could be reduced to once per frame by changing the 'Retries SAMEFRAME' command to 'Retries NEXTFRAME'.

The alternative type of Generator Script has 'Retries NONE', and includes all the NAKs precisely. Running this kind of script will depend on an exact reproduction of the original responses from the device, and in the case of a manually moved mouse, would not be very useful. It would be more appropriate if the device behaviour was entirely self-determined.

The message is to use scripts with care, and not depend on lengthy automatically generated scripts, but rather to use those as a guide and build the script you use section by section, treating each device on its merits.

## 4.14 Generator Scripts For On-the-Go

The automatic generation of Generator Scripts from capture files, may not produce the desired results when OTG is involved.

We offer the following sequences for the testing of OTG protocols:

### 4.14.1 Host Negotiation Protocol – Full Speed

#### 4.14.1.1 A-device transfers control to B-device (A-device script)

```
===== Host Negotiation Protocol =====
; A-device has sent Set Feature command to enable the B-device for HNP

Suspendhnp                ; stop sending SOFs, apply 15k pulldowns
WaitPullupOff             ; wait for B-device to remove its
                           ; pullup resistor
ControlMode DEVICE        ; A becomes the device (floats data lines)
Idle 90000                ; (1.5ms - must be less than 3ms)
PullupOn FULLSPEED        ; ...and connects its pullup
WaitReset                 ; B-device has become Host, wait for it
                           ; to reset the bus
```

#### 4.14.1.2 A-device transfers control to B-device (B-device script)

```
===== Host Negotiation Protocol =====

WaitSuspend               ; wait for A-device to suspends the bus
Idle 12000000
PullupOff                 ; B device removes pullup resistor
ControlMode HOST          ; and becomes host, applying 15k pulldowns
WaitPullupOn FULLSPEED    ; wait for A-device to connect pullup and
                           ; become device

Idle 12000000
Reset 20000
SOFOn 0
SOFs 20
```

#### 4.14.1.3 B-device transfers control back to A-device (A-device script)

```
WaitSuspend          ; results in FS pullup being on
PullupOff             ; A removes its pullup (bus floats)
ControlMode HOST      ; and returns to being host
                     ; (applies 15k pulldowns)
Idle 6000             ; allow time for D+ to go low
WaitPullupOn FULLSPEED ; B connects pullup and returns to
                     ; being device
```

```
;=====
```

```
Reset 20000
```

#### 4.14.1.4 B-device transfers control back to A-device (B-device script)

```
SuspendHnp           ; stop sending SOFs, apply 15k pulldowns
ControlMode DEVICE    ; B returns to being the Device
                     ; (floats data lines)
PullupOn FULLSPEED    ; ...and connects its pullup
```

## 4.14.2 Host Negotiation Protocol – High Speed

### 4.14.2.1 A-device transfers control to B-device (A-device script)

```
===== Host Negotiation Protocol =====
; A-device has sent Set Feature command to enable the B-device for HNP

Suspendhnp                ; stop sending SOFs, release high speed
                           ; termination, and apply 15k pulldowns
WaitPullupOn FULLSPEED    ; B-device reverts to FS
WaitPullupOff             ; Waiting for B-device to remove
                           ; its pullup resistor
ControlMode DEVICE        ; A becomes the device (floats data lines)
Idle 90000                ; (1.5ms) must be less than 3ms
PullupOn FULLSPEED        ; ...and connects its pullup
WaitReset                 ; B-device has become Host
SendChirp 1 2000 0        ; start HS detection handshake
WaitChirp
```

### 4.14.2.2 A-device transfers control to B-device (B-device script)

```
===== Host Negotiation Protocol =====

WaitSuspend               ; wait 3ms of idle, then remove
                           ; termination, and apply FS pullup
PullupOn FULLSPEED        ; (cancels HS)
Idle 600000               ; 1-146ms
PullupOff                 ; B-device removes pullup resistor
                           ; (floats data lines)
ControlMode HOST          ; applies 15k pulldowns
WaitPullupOn FULLSPEED    ; A-device connects pullup and
                           ; becomes device

Idle 12000000
Reset 0                   ; reset bus
WaitChirp                 ; and start HS detection handshake
SendChirp 750 50 50
Idle 180
SOFOOn 0
```

#### 4.14.2.3 B-device transfers control back to A-device (A-device script)

```
WaitSuspend                ; results in FS pullup being on
Idle 60000
PullupOff                  ; A removes this pullup (bus floats)
ControlMode HOST           ; and returns to being host
                           ; (applies 15k pulldowns)
Idle 6000                  ; time for bus to discharge
WaitPullupOn FULLSPEED    ; B connects pullup and
                           ; returns to being device
Idle 120000                ; minimum of 4ms after WaitSuspend
                           ; to allow correct report from analyser
Reset 0                    ; reset bus
WaitChirp                  ; and start HS detection handshake
SendChirp 750 50 50
Idle 180
SOFOn 0
```

#### 4.14.2.4 B-device transfers control back to A-device (B-device script)

```
SuspendHnp                 ; stop sending SOFs, apply 15k pulldowns
ControlMode DEVICE         ; B returns to being the Device
                           ; (floats data lines)
PullupOn FULLSPEED        ; ...and connects its pullup
Idle 200000
WaitReset                  ; wait for A-device to reset bus
SendChirp 1 2000 0        ; start HS detection handshake
WaitChirp
```

### 4.14.3 Session Request Protocol – Full Speed

#### 4.14.3.1 B-device requests a session from A-device (A-device script)

```
;===== Ending original session =====
Suspend 10                ; stop sending SOFs
VbusOff                   ; end session
WaitPullupOff             ; wait for B device to
                          ; remove its pullup resistor

;===== Session Request Protocol =====
WaitPullupOn FULLSPEED    ; data line pulse from B
WaitPullupOff

WaitVbusOn                ; start of VBUS pulse from B
VbusOn                    ; A applies Vbus
WaitPullupOn FULLSPEED    ; B applies pullup resistor,
                          ; Session starts

Idle 6000000
Reset 20000
Idle 5000
SOFOOn 0
```

#### 4.14.3.2 B-device requests a session from A-device (B-device script)

```
;===== Ending original session =====
WaitSuspend
WaitVbusOff               ; detect end of session
PullupOff                 ; remove pullup resistor
Idle 600000

;===== Session Request Protocol =====

PullupOn FULLSPEED        ; 5ms data line pulse
Idle 300000
PullupOff
Idle 300000

VbusOn                    ; 5ms Vbus pulse
Idle 300000
VbusOff
Idle 6000

WaitVbusOn                ; wait for Vbus from A-device
PullupOn FULLSPEED        ; apply pullup resistor
                          ; session starts

WaitReset
```

#### 4.14.4 Session Request Protocol – High Speed

##### 4.14.4.1 B-device requests a session from A-device (A-device script)

```
;===== Ending original session =====
Suspend 10                ; stop sending SOFs, apply 15k pulldowns
VbusOff                   ; end session
WaitPullupOff             ; wait for B device to
                          ; remove its pullup resistor

;===== Session Request Protocol =====
WaitPullupOn FULLSPEED    ; data line pulse from B
WaitPullupOff
WaitVbusOn                ; start of VBUS pulse from B
VbusOn                   ; A applies Vbus
WaitPullupOn FULLSPEED    ; B applies pullup resistor
                          ; Session starts

Idle 6000000

Reset 0
WaitChirp
SendChirp 750 50 50
Idle 180
SOFOff 0
SOFs 875
```

##### 4.14.4.2 B-device requests a session from A-device (B-device script)

```
;===== Ending original session =====
WaitSuspend
WaitVbusOff               ; detect end of session
PullupOff                 ; remove pullup resistor
Idle 600000

;===== Session Request Protocol =====
PullupOn FULLSPEED
Idle 300000                ; 5ms data line pulse
PullupOff
VbusOn
Idle 300000                ; 5ms Vbus pulse
VbusOff
Idle 6000
WaitVbusOn                ; wait for Vbus from A-device
PullupOn FULLSPEED        ; apply pullup resistor

WaitReset                 ; wait for A-device to reset bus
SendChirp 1 2000 0        ; start HS detection handshake
WaitChirp
```



## **4.15 Generator Limitations**

The generator can be programmed to behave as:

- A high speed, full speed or low speed host
- A high speed, full speed or low speed device
- The up- or downstream facing port of a USB2.0
- The downstream facing port of a USB1.1 hub

*It cannot, however, be programmed to behave as the upstream facing port of a USB 1.1 hub.*

## 5 TROUBLESHOOTING

### **During capture a Data Overrun message appears.**

This happens when the device under test generates more traffic than the Host computer can handle. Check that the Host computer has a High Speed USB connection and is sufficiently powerful. A test set up using two computers is preferable.

### **The data captured contains a large number of CRC or other errors.**

Check the cabling between the Packet-Master and the device under test and to the Host under test. The cabling should be kept as short as possible with the total length of cable not exceeding 4 metres.

### **The data captured just contains a Plugged In message and a Start of Reset message.**

This may happen if, after starting capture, a high speed device is plugged into the Packet-Master USB12, which does not handle high speed.

### **The data captured contains a large number of “Spurious Data” or “Both Lines High” errors.**

This may be the result of using excessively long cables in the test setup or perhaps trying to analyse a high speed device with the Packet-Master USB12.

An alternative possibility, which we have seen on some (non-approved) devices, is that the designer has incorporated reactive elements into the data lines such that on the bus itself the voltage waveform is not readable. A quick check with an oscilloscope will confirm this situation as the data lines will not show a clean square appearance, but rather mostly ringing shapes. One solution to devices of this type is to view their data upstream of a Full Speed Hub

**My capture buffer fills up too quickly to collect any useful events.**

Some devices can continuously NAK transfers, which leads to a very high bandwidth of not-very-useful data. We suggest that you disable the capture of NAKs in the Edit...Options...Capture dialog, which will reduce the amount of data captured, limiting it to transfers which are not NAKed. NYETed split transactions to a high-speed hub can also be omitted.

If necessary, a further reduction in captured data can be achieved by disabling the capture of SOFs or Keep Alive events.

You can also increase the buffer size in the same dialog. If doing this causes system slowdown problems (caused by the system using virtual memory), then consider adding RAM to your computer.

**The analyser is not showing any events although the device is functioning correctly.**

One possibility is that you are trying to capture high speed events with a USB12 analyser which can only capture low speed and full speed events.

A further possibility is that the conditions for the presence of a device are not being seen by the analyser, which expects to see VBUS appear and then a stable state with one data line high. For example, in a test set-up using flying leads soldered to a development board, the VBUS must be connected and working.

A host, whose connect timing is out of specification can cause problems with capturing if it does not wait 100ms before resetting the device as required. Some chips have been known to wait only 50us.

To avoid such problems we recommend always plugging the host connection into the analyser first, and using the other connection to the device to control the plugging-in and plugging-out events.

If the message “\*Device plug-in not detected\*” is showing in the capture window, then you should remember that the analyser has to have seen a high speed device being plugged in (even if it is before a

capture is started) in order to follow the transition to high speed. If this does not happen, the analyser will still be monitoring for low/full speed signals and not see the high speed traffic. Simply unplug and replug the device, and remember in future not to plug the device into the analyser until the analyser is powered up.

**The generator is alternately flashing the 'Generator' and 'Host' indicators.**

You have a cable plugged in the front of the generator from a host supplying  $V_{BUS}$ . The flashing is warning you not to continue until you unplug this connection as the generator will be providing its own  $V_{BUS}$  supply on the same wire, which could result in damage.

## **6 WARRANTY**

### ***6.1 Warranty***

MQP Electronics guarantees that its products are free from defects in materials and workmanship for the warranty period, subject to the limitations below. MQP Electronics will at its discretion either repair or replace any part that proves defective because of faulty materials or workmanship.

### ***6.2 Limitations***

This warranty does not cover any damage that results from any accident, misuse or unauthorized disassembly or repair. This product is not authorized for use as a critical component in life support equipment or any application where failure would result in any loss, injury or damage to persons or property.

### ***6.3 Warranty Period***

The warranty starts on the day of purchase and covers a period of one year.

### ***6.4 Obtaining Service***

Defective product may be returned to the authorized distributor from whom you purchased the product. Defective product may be returned direct to MQP Electronics. Please call +44 (0)1666 825 666 and request a Return Material Authorization (RMA) number from customer services.