Packet-Master USB-PET
User Manual

MQP Electronics Ltd
Unit 2, Park Road Centre
Malmesbury
Wiltshire SN16 0BX
United Kingdom
e-mail: sales@mqp.com Website: www.mqp.com

Contents

1 INTRODUCTION - PET Architecture

The PET (Protocol and Electrical Tester) is a comprehensive USB tool, capable of performing compliance testing or assisting in development work leading towards compliance testing on On-the-Go, Battery Charging and other general USB applications. It can best be described in terms of a number of functional blocks, as follows:

1.1 Serial Interface Engine (SIE)

A fully functional SIE, with both host and peripheral capabilities, connected via a PHY to the UUT micro-AB receptacle on the front panel. It is under the control of the Script Processor.

1.2 Electrical Test Board (ETB)

This contains circuitry to allow control and measurement of the electrical parameters for USB, OTG and BC specifications. It includes VBUS Generator, ID pin circuitry, data line test mode circuitry, VBUS current and voltage loads, and a variety of voltage and current measuring blocks. Extra connections are provided on the front panel to enable the testing of Accessory Charger Adapters (ACAs).

1.3 Script Processor

Scripts are downloaded to this processor to control the sequence of operations required for a particular test. The processor controls the SIE and ETB as required by the operator. Scripts for all the OTG and BC compliance tests are provided by the GraphicUSB application which accompanies the PET. GraphicUSB also supports user-written scripts, to allow particular test situations to be set up.

1.4 High Speed / Full Speed/ Low Speed USB Analyser

The PET also provides full USB analyser functionality. Of particular interest is the fact that this analyser has zero impact on the data line transmission quality. Messages generated by a running script can be inserted into the analyser capture. The analyser also displays a continuous accurate monitoring of the VBUS voltage value.

2 GETTING STARTED

2.1 Installing the Software and Driver

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

2.1.1 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.

2.1.2 Installing the Driver

- The first time you plug in the USB cable from your Packet-Master USB-PET, 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 or Windows 7, the process should 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”.

2.1.3 Updating the Driver

- If there is a requirement to update the USB Driver for the Packet-Master USB-PET, 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\”

2.1.4 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.

2.2 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 analyzer capabilities of GraphicUSB.

2.3 Front and Back Panels – Packet-Master USB-PET

- The Power indicator illuminates red when the PET is powered, it turns green when the PET has been configured by the driver.
- The Running indicator illuminates yellow while a script is running.
- The Pass/Fail indicator illuminates steady green for a pass, slow flashing red for a non-fatal test failure, and fast flashing red for a fatal test failure.
- The micro-AB USB connector is used for connection to the unit-under-test (UUT).
- The 9-pin D-Type connector is used for the additional connections required to test an Accessory Charger Adapter (ACA).

- The Packet-Master USB-PET must be powered externally by the external power supply, included with the unit, so that sufficient voltage and current are available to the device under test.
- A High Speed USB (480 Mbit/s) provides the connection to the Host PC.
- A trigger output for external equipment is provided on a BNC connector.
- A further BNC connector provides the ability to monitor one of the following; VBUS OTG, VBUS ACC, VBUS CHG, D+ (test mode), D- (test mode).

2.4 Test Cables

The following test cables are specified for use with the PET. Special Cables A, B and E are provided as standard. Cables C and D are available as optional extras.

Special Test Cable A - Micro-B plug to Micro-B plug		
Micro-B plug (PET)	**Micro-B plug (UUT)**	**Purpose**
1	1	VBUS
2	2	D-
3	3	D+
4	4	ID
5	5	GND

(This cable is supplied as standard.)

This cable has been specially manufactured to allow control of the ID pin of the unit-under-test. It is important to use this cable when the test specifies it. The particular resistance of the cable has also been allowed for in the test suite.

Note: The original version of this cable was 1m in length, and was not suitable for testing PDs in situations where the PET was acting as an ACA, and the PD was drawing more than I_{CFG_MAX}. The cable now supplied is 200mm in length and has a GND resistance of less than 100mΩ.

Special Test Cable B - Micro-B plug to Standard-A plug		
Micro-B plug (PET)	**Standard-A plug (UUT)**	**Purpose**
1	1	VBUS
2	2	D-
3	3	D+
nc		
5	4	GND

(This cable is supplied as standard.)

Although this is a standard cable configuration, it is important to use this cable when the test specifies it, as the particular resistance of the cable has been allowed for in the test suite.

Special Test Cable C – 9-pin D-type Assembly			
D-type (PET)	**Standard-A Receptacle (ACA Charger Port)**	**Micro-B Plug (ACA Accessory Port)**	**Purpose**
1 (linked to 4)			Cable Sense
2		1	Accessory VBUS
3		4	Accessory ID
4	4	5	GND
5	1		Charger VBUS
6	2		Charger D-
7	3		Charger D+
8			OTG VBUS
9			OTG ID

(Available as an optional extra.)

This cable is used when testing a Micro-ACA.

Note: This is correct for a Micro-ACA with a captive charger port cable. For a Micro-ACA with a Micro-B receptacle as the charger port, connect Special Test Cable B to the Standard-A receptacle of Special Test Cable C, and plug the other end into the charger port of the ACA.

Special Test Cable D – 9-pin D-type Assembly			
D-type (PET)	**Standard-A Receptacle (ACA Charger Port)**	**Standard-A Plug (ACA Accessory Port)**	**Purpose**
1 (linked to 4)			Cable Sense
2		1	Accessory VBUS
3			Accessory ID
4	4	4	GND
5	1		Charger VBUS
6	2		Charger D-
7	3		Charger D+
8			OTG VBUS
9			OTG ID

(Available as an optional extra.)

This cable is used when testing a Standard-ACA.

Note: This is correct for a Standard-ACA with a captive charger port cable. For a Standard-ACA with a Micro-B receptacle as the charger port, connect Special Test Cable B to the Standard-A receptacle of Special Test Cable C, and plug the other end into the charger port of the ACA.

Special Test Cable E - Micro-A plug to Standard-A receptacle		
Micro-A plug (PET)	**Standard-A receptacle (UUT)**	**Purpose**
1	1	VBUS
2	2	D-
3	3	D+
4 - Connected to pin 5		
5	4	GND

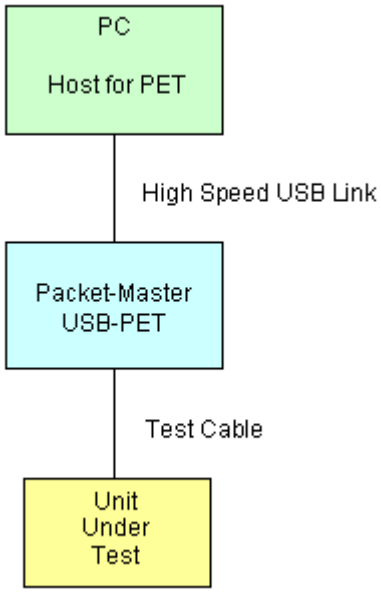
(This cable is now supplied as standard.)

This is a short adapter cable, which may be used to connect a PD with a standard-A plug to the PET.

2.5 Test Set Up

2.5.1 PET Host

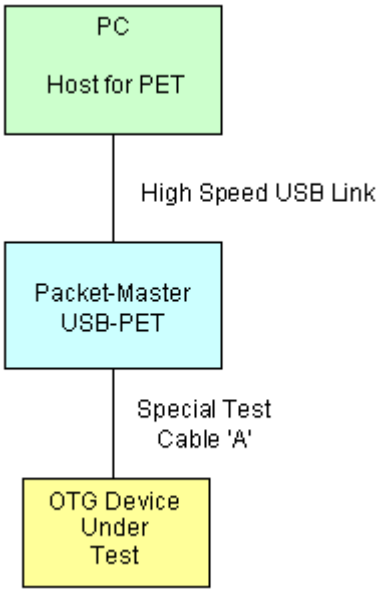
To achieve a good capture rate it is important to provide a suitable test environment. The Packet-Master unit should preferably be hosted by a good performance PC with a high speed USB connection. The USB host controller should not be shared by any other high data throughput USB device while doing the testing.



2.5.2 OTG Device as Unit-Under-Test (Setup no. 1)

When running a test-suite relating to an OTG device, the first test will prompt you to connect it to the PET using 'Special Test Cable A'. This ***Micro-B plug*** to ***Micro-B plug*** cable is provided with the PET unit and it is essential that this particular cable is used, for the following reasons:

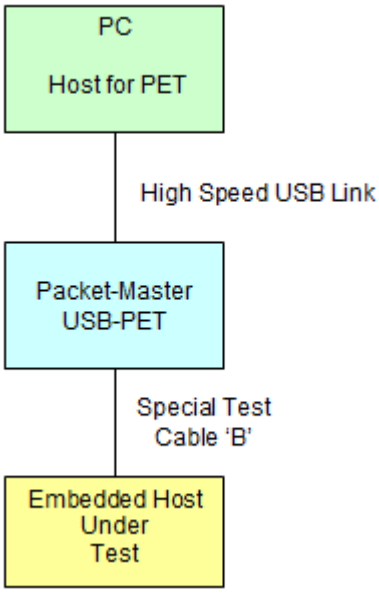
- It has 5 cores, instead of the usual 4. This allows the PET to control the ID pin of the UUT.
- The resistance of this cable has been allowed for in tests involving large VBUS currents with measurements on VBUS current and voltage.



2.5.3 Embedded Host as Unit-Under-Test (Setup no. 2)

When running a test-suite relating to an Embedded Host, the first test will prompt you to connect it to the PET using 'Special Test Cable B'. This ***Micro-B plug*** to ***Standard-A plug*** cable is provided with the PET unit and it is essential that this particular cable is used, for the following reason:

- The resistance of this cable has been allowed for in tests involving large VBUS currents with measurements on VBUS current and voltage.



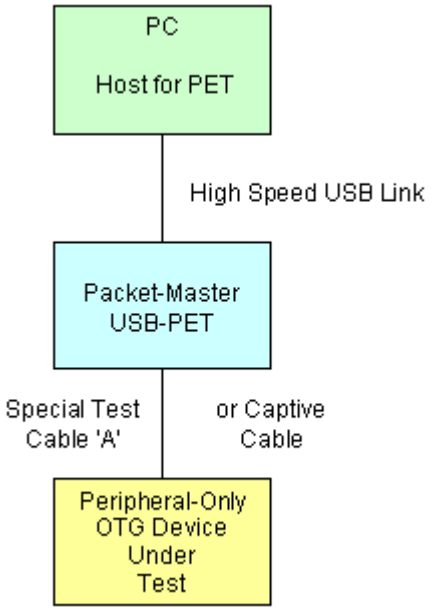
2.5.4 Peripheral Only as Unit-Under-Test (Setup no. 3)

When running a test-suite relating to a Peripheral-Only OTG device, the first test will prompt you to connect it to the PET using 'Special Test Cable A'. This ***Micro-B plug*** to ***Micro-B plug*** cable is provided with the PET unit and it is essential that this particular cable is used, for the following reason:

- The resistance of this cable has been allowed for in tests involving large VBUS currents with measurements on VBUS current and voltage.

Another possibility is that the device has a captive cable with a **Micro-A plug**. In this case use this, and check the 'Captive Cable' check box, in the 'PET Test Suites' Dialog.

Finally, the device may have a captive cable with a **Standard-A plug**. In this case, use a suitable adapter to attach the **Standard-A plug** to the **Micro-AB** receptacle of the PET, and check the 'Captive Cable' check box, in the 'PET Test Suites' Dialog.



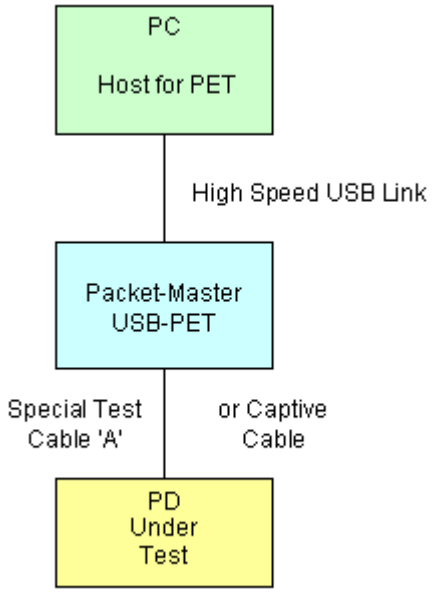
2.5.5 PD as Unit-Under-Test (Setup no. 4)

A Portable Device (PD) is assumed to have a Micro-B receptacle or a Micro-AB receptacle. A PD that mates with an ACA-Dock or ACA is required to have a Micro-AB receptacle.

When running a test-suite relating to a PD, the first test will prompt you to connect it to the PET using 'Special Test Cable A'. This ***Micro-B plug*** to ***Micro-B plug*** cable is provided with the PET unit and it is essential that this particular cable is used, for the following reason:

- The resistance of this cable has been allowed for in tests involving large VBUS currents with measurements on VBUS current and voltage.

The other possibility is that the device has a captive cable with a ***micro-A plug***. In this case use this, and check the 'Captive Cable' check box, in the 'USB-PET Test Suites' Dialog.

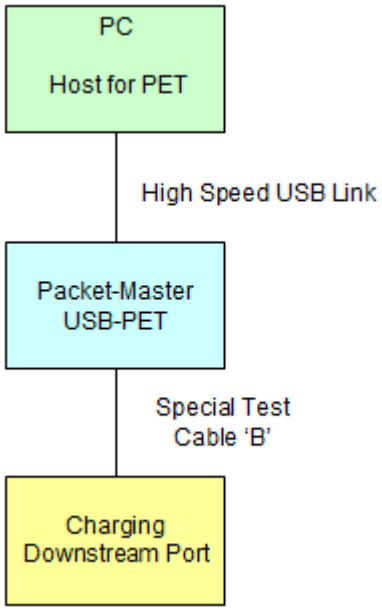


2.5.6 CDP as Unit-Under-Test (Setup no. 5)

A Charging Downstream Port (CDP) must be equipped with a Standard-A receptacle.

When running a test-suite relating to a CDP, the first test will prompt you to connect it to the PET using 'Special Test Cable B'. This ***Micro-B plug*** to ***Standard-A plug*** cable is provided with the PET unit and it is essential that this particular cable is used, for the following reason:

- The resistance of this cable has been allowed for in tests involving large VBUS currents with measurements on VBUS current and voltage.



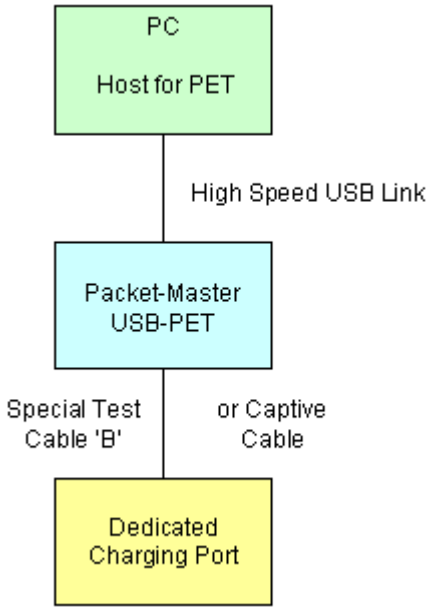
2.5.7 DCP as Unit-Under-Test (Setup no. 6)

A Dedicated Charging Port (DCP) must be equipped with a Standard-A receptacle, or a captive cable terminated with a Micro-B plug.

When running a test-suite relating to a DCP, the first test will prompt you to connect it to the PET using ‘Special Test Cable B’. This ***Micro-B plug*** to ***Standard-A plug*** cable is provided with the PET unit and it is essential that this particular cable is used, for the following reason:

- The resistance of this cable has been allowed for in tests involving large VBUS currents with measurements on VBUS current and voltage.

The other possibility is that the device has a captive cable with a ***micro-B plug***. In this case use this, and check the 'Captive Cable' check box, in the 'USB-PET Test Suites' Dialog.

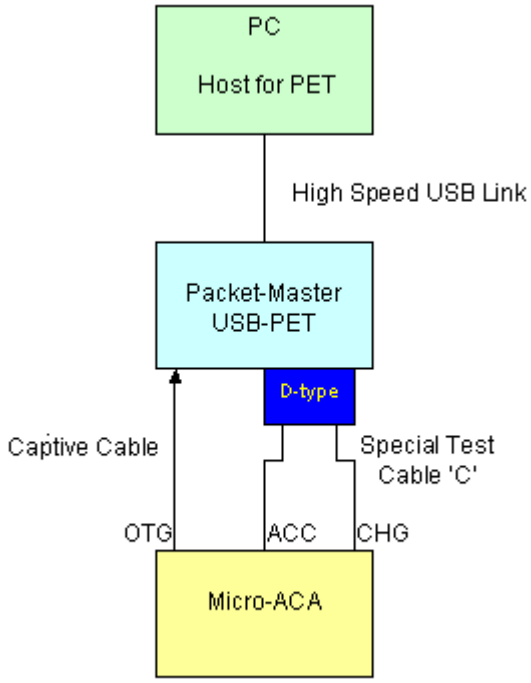


2.5.8 Micro-ACA (Separate Charger) as Unit-Under-Test (Setup no. 7)

An Accessory Charger Adapter having a Micro-AB receptacle for its accessory port (Micro-ACA) must be equipped with:

- a captive cable terminated with a Micro-A plug for its OTG port
- a Micro-B receptacle, or a captive cable with Standard-A plug for its charger port, and of course
- a Micro-AB receptacle for its accessory port

When running a test-suite relating to a Micro-ACA, the first test will prompt you to connect it to the PET using ‘Special Test Cable C’.



2.5.9 Micro-ACA (Combined Charger) as Unit-Under-Test (Setup no. 7b)

This is the same as Setup 7 except that there is no charger cable coming from the UUT.

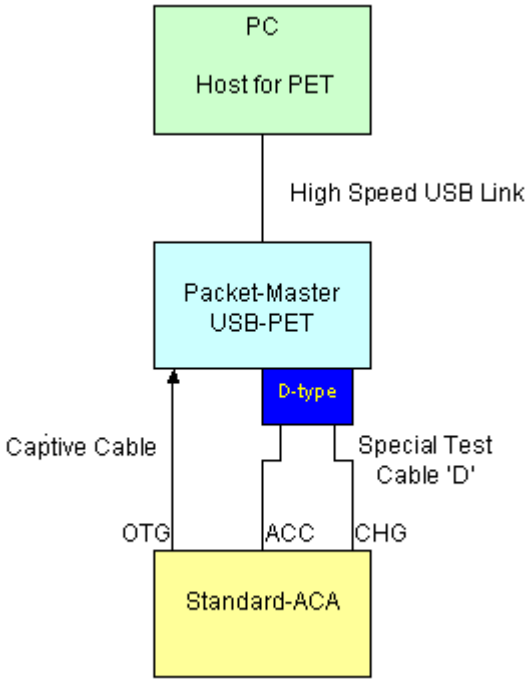
2.5.10 Standard-ACA (Separate Charger) as Unit-Under-Test (Setup no. 8)

An Accessory Charger Adapter having a Standard-A receptacle for its accessory port (Standard-ACA), must be equipped with:

- a captive cable terminated with a Micro-A plug for its OTG port
- a Micro-B receptacle, or a captive cable with Standard-A plug for its charger port, and of course
- a Standard-A receptacle for its accessory port

When running a test-suite relating to a Standard-ACA, the first test will prompt you to connect it to the PET using ‘Special Test Cable D’.

(Note: This cable is currently an optional extra. An alternative is to use ‘Special Test Cable C’ and to use a suitable adapter to convert the Micro-B plug on the end of the cable marked ‘Accessory’ to a Standard-A plug.)



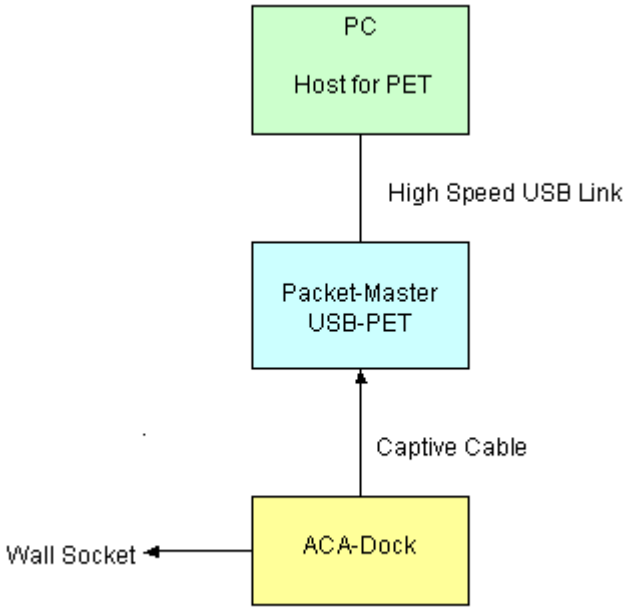
2.5.11 Standard-ACA (Combined Charger) as Unit-Under-Test (Setup no. 8b)

This is the same as Setup 8 except that there is no charger cable coming from the UUT.

2.5.12 ACA-Dock as Unit-Under-Test (Setup no. 9)

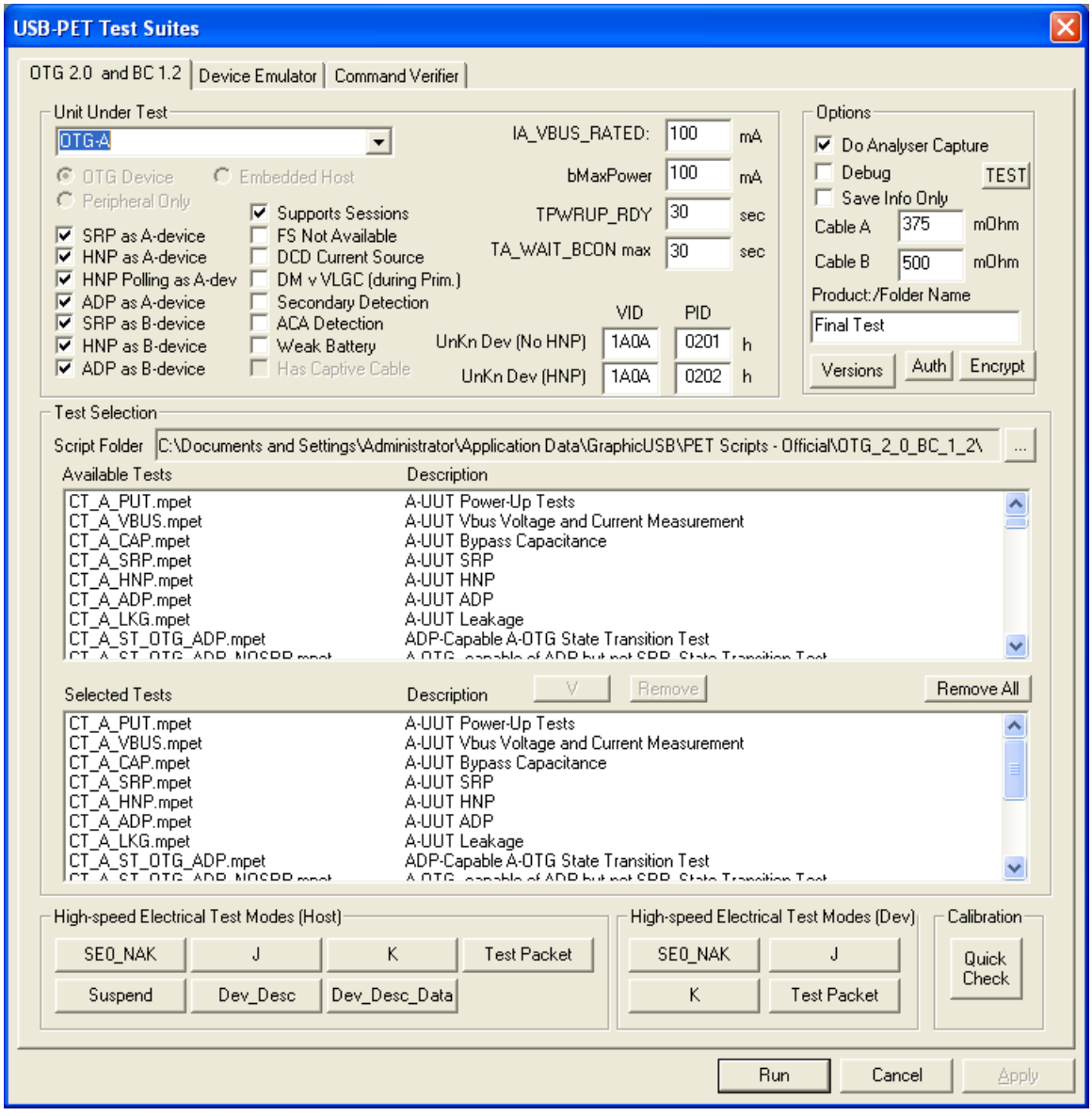
An ACA-Dock must be equipped with a Micro-A plug, for connecting to the Micro-AB receptacle of a PD. It is represented here as a captive cable. In practice it may comprise part of a fixture, which may be difficult to connect to the PET front panel. In this case it is the responsibility of the vendor to provide a suitable means to connect the ACA-Dock to the Micro-AB receptacle of the PET.

When running a test-suite relating to an ACA-Dock, the first test will prompt you to connect it to the PET using its captive cable.



2.6 Running Standard Compliance Tests

To run the standard tests, click on “Operations…Pet Compliance Tester” on the menu bar, or on the “PET” icon on the Tool Bar. The Test Suite dialog appears.



Select the type of unit to be tested using the 'Unit Under Test' combo box.

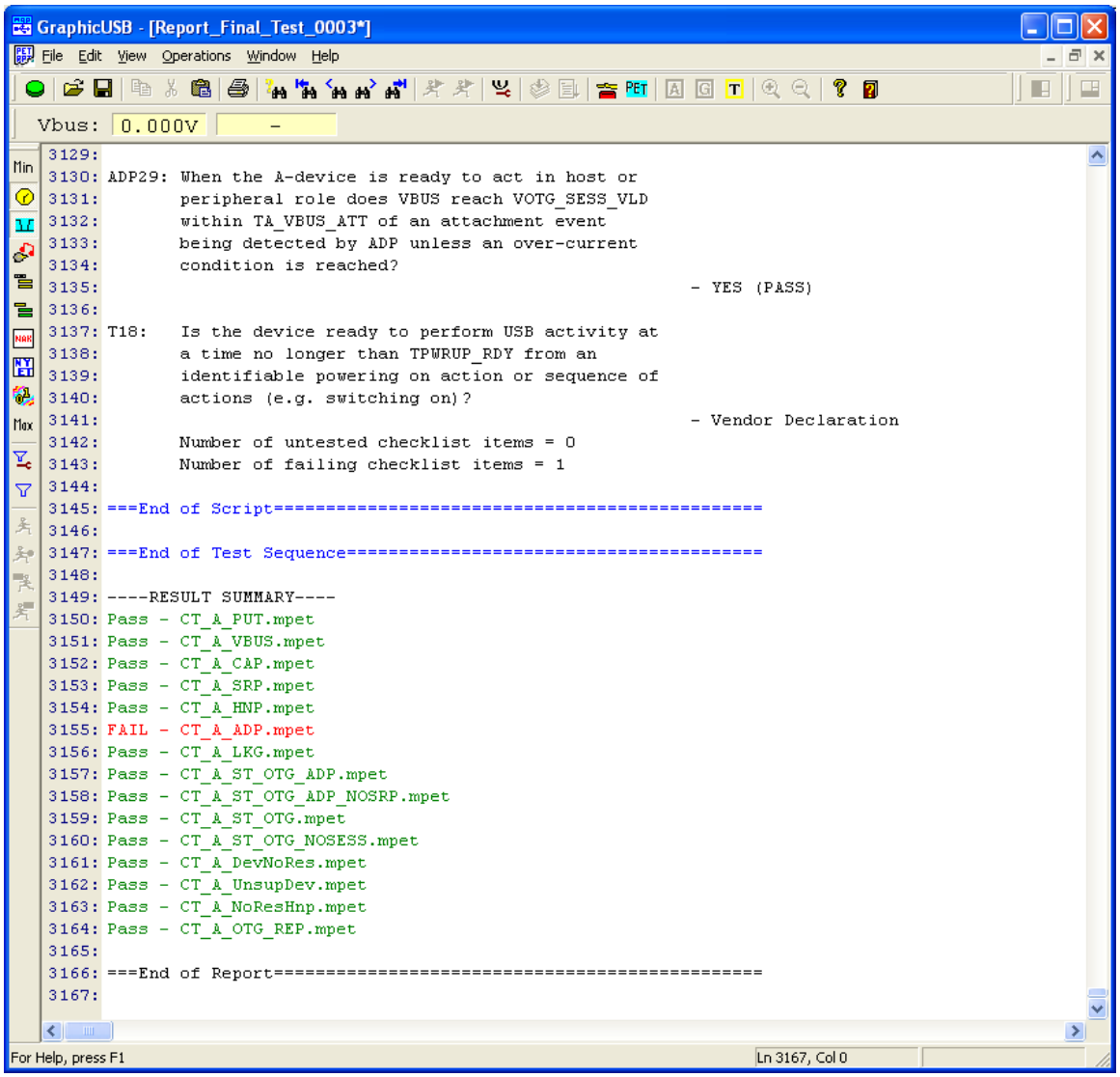
Then refer to the completed Compliance Checklist, and ensure that the other 'Unit Under Test' checkboxes and parameters are correctly entered (see next section for details).

The appropriate tests will be loaded into the 'Selected Tests' list box. These tests are now ready to automatically be run in sequence.

Decide whether to check 'Do Analyser Capture', and specify a Product name so that the reports can be saved into an appropriate folder.

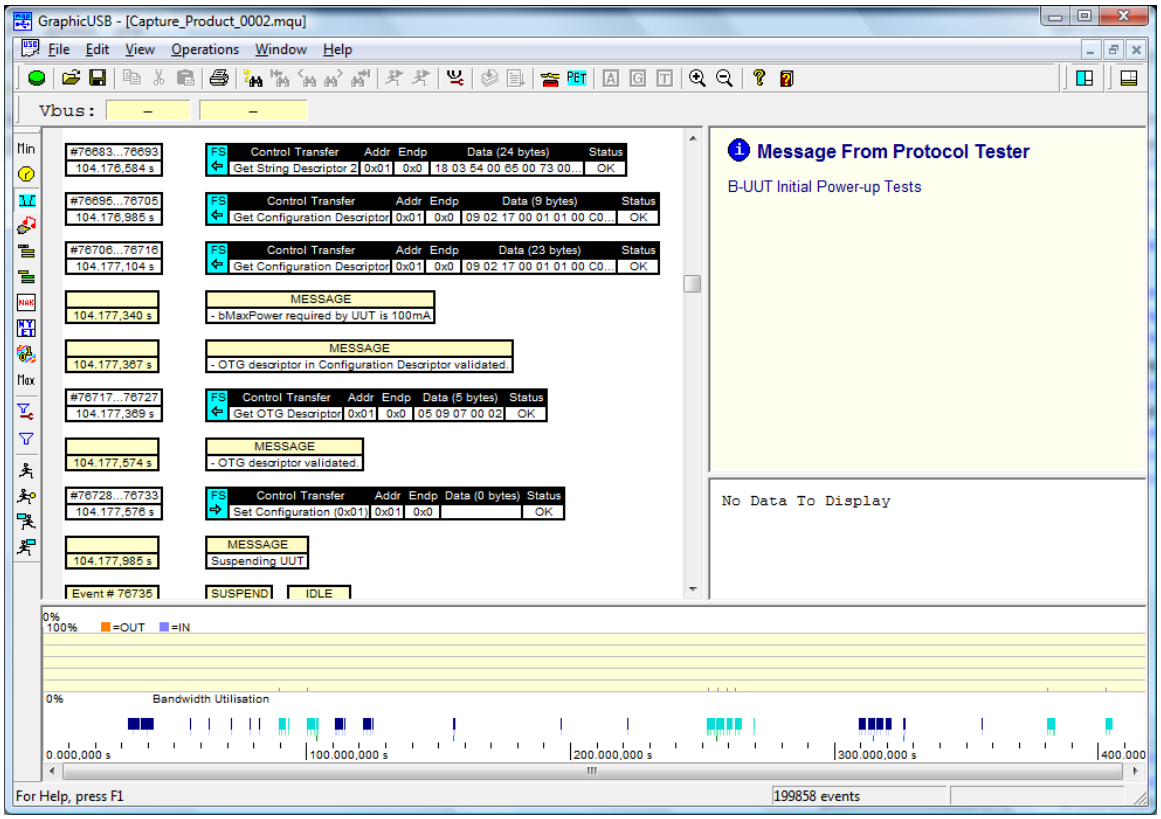
Click on 'Run' to start the test suite.

A text report file will be created, into which the test results are written.



The report ends with a list of checklist items and whether each passed, failed or was not able to be tested; followed by a summary indicating which test sequences passed.

If 'Do Analyser Capture' was checked, an analyser capture file will also be created.



This file will be useful to help interpret the reasons for any test failures.

The report and capture files should be saved as required. Meaningful file names and folders are suggested, based on the folder name entered in the USB-PET Test Suites dialog.

The report file has the extension ‘.PetRpt’, and the capture file has the extension ‘.mqu’.

PET report files with the ‘.PetRpt’ extension are actually ASCII files with 8 bit characters, and may be renamed to ‘.txt’ files if required. If opened later in GraphicUSB, the context colouring feature will be lost if the file has been renamed.

2.7 Compliance Checklist Entries in the Test Suite Dialog

Using the Compliance Checklist(s) provided by the vendor, modify the check boxes and edit boxes as required.

2.7.1 Items from OTG Checklist

Input	Type	Purpose
OTG Device	Mutually exclusive check boxes	Automatically selected by UUT items OTG-A or OTG-B.
Embedded Host		Automatically selected by UUT item Embedded Host.
Peripheral Only		Automatically selected by UUT item Embedded Host.
Supports Sessions	Check box	Check this box if the OTG A-UUT does not keep VBUS enabled all the time that the ID pin is held low, or if the EH does not keep VBUS high all the time it is powered up. In either case it is assumed that SRP or ADP is available to detect the presence of a device.
SRP as A-device	Check box	Check this box if the UUT, as an A-device, supports detecting, and acting on, an SRP pulse generated by a connected device.
HNP as A-device	Check box	Check this box if the UUT, as an A-device, supports HNP to enable the connected B-device to become host if it so requires.
HNP Polling as A-device	Check box	Check this box if the UUT, as an A-device, supports HNP polling. If it does it is allowed to remain as host, for as long as the other device does not set its Host Request Flag.
ADP as A-device	Check box	Check this box if the UUT, as an A-device, supports ADP probing to detect the presence or otherwise of a connected device.
SRP as B-device	Check box	Check this box if the UUT, as a B-device, supports generating an SRP pulse in order to start a session (cause the connected A-device to turn on VBUS).
HNP as B-device	Check box	Check this box if the UUT, as an B-device, supports HNP to allow it to become host if it so requires.
ADP as B-device	Check box	Check this box if the UUT, as an B-device, supports ADP sensing and probing to detect the presence or otherwise of a connected device.
FS Not Available	Check box	Check this box if UUT does not fully support full-speed operation. This is not permitted for an OTG device, but may be for an Embedded Host.
IA_VBUS_RATED	Edit box	The rated output current of an A-device in mA units.
bMaxPower	Edit box	bMaxPower (sic) is the highest current, in mA, which will be drawn by the UUT in its normal role, and is the value declared in any of its Configuration Descriptors. This value ignores current drawn under the Battery Charging provisions.
TPWRUP_RDY	Edit box	Maximum time, in seconds, specified by vendor from powering on the UUT until it is ready to perform USB functionality. By default this is set to 30 seconds, but a vendor is permitted to specify a longer time.
TA_WAIT_BCON max	Edit box	The maximum time, in seconds, that VBUS is left on for by an A-device, in the absence of a B-device connecting. The default value is thirty seconds. A vendor is permitted to specify a longer time, but should be aware that this will have an impact on the time taken for, and therefore possibly the cost of, compliance testing.
Unknown Dev (No HNP)	Edit boxes	The test will use the VID/PID combination specified during tests for error messages, when an unknown B-device, not capable of HNP, is connected. A

		default value (1A0A/0201) is used, but any other device not on the UUT's TPL may be defined here.
Unknown Dev (HNP)	Edit boxes	The test will use the VID/PID combination specified during tests for error messages, when an unknown B-device, capable of HNP, is connected. A default value (1A0A/0202) is used, but any other device not on the UUT's TPL may be defined here.

2.7.2 Additional Items from BC Checklist

Input	Type	Purpose
DCD Current Source	Check box	Check this if the device under test uses a current source to implement Data Contact Detect (DCD).
DM v VLGC (during Prim.)	Check box	Check this if the device under test, during Primary Detection, compares the voltage on D- with VLGC, and only determines that it is connected to a DCP or CDP if D- is greater than VDAT_REF but less than VLGC. This is for report information only and has no effect on any test.
Secondary Detection	Check box	Check this if the device under test supports secondary detection of charging ports.
ACA Detection	Check box	Check this if the device under test supports detection of ACAs.
Has Captive Cable	Check box	Check this box if a UUT has a captive cable. Test cable resistance will then be assumed to be zero.

2.7.3 Options Items

Input	Type	Purpose
Do Analyser Output	Check box	Check if analyser capture output required.
Debug	Check box	If checked, the variable _DEBUG_MODE is set to 1.
Save Info Only	Check box	If checked, clicking on 'Run' will save the settings without running the script suite.
Cable A	Edit box	Test Cable A loop resistance in mΩ.
Cable B	Edit box	Test Cable B loop resistance in mΩ.
Product/Folder Name	Edit box	Defines the folder in which the results will be saved by default.

2.8 High Speed Electrical Tests

The Test Suite Dialog has buttons near the bottom, each of which can select a single HS Electrical Test to be performed. These tests are intended to be carried out in conjunction with appropriate HS signal measuring equipment.

MQP offers a range of Test Fixtures, specially designed for use with the PET. These fixtures allow the ID signal to be conveyed between the PET, and the unit-under-test (UUT). They additionally allow automatic switching of the termination network, at the appropriate point in the test.

Various test equipment manufacturers produce suitable oscilloscopes, and test probes for use with these fixtures. Using such equipment, it is possible to observe and measure various signal quality characteristics.

Test Fixtures available include:

- High Speed Signal Quality (Host or Peripheral) plus Inrush Measurement Test Fixture
- Impedance Measurement (TDR) plus Inrush Measurement Test Fixture
- Receiver Sensitivity plus Inrush Measurement Test Fixture
- Economy Breakout Board plus Inrush Measurement Test Fixture

The available High Speed Electrical Tests are listed in the following section, and detailed in the USB 2.0 core specification, and in the OTG 2.0 Supplement. The major section ‘HIGH SPEED ELECTRICAL TESTS’ details use of the fixtures available from MQP Electronics.

2.9 Standard Compliance Tests Supplied with PET

The standard test scripts listed below appear in the folder:

```
…Pet Scripts - Official\OTG_2_0_BC_1_2
```

It is important that these files not be edited, otherwise the test suite can only be run with accompanying warnings, which will also be embedded in the text report. An authentication file confirms that the files are original.

It is possible to copy the files into a different folder and run the scripts individually in this new folder. In this case modifications are not reported, to allow changes to be made while investigating a UUT problem.

Files supplied are:

Filename	Function
OTG 2.0 A-UUT Tests	
CT_A_PUT.mpet	A-UUT Power-Up Tests
CT_A_VBUS.mpet	A-UUT Vbus Voltage and Current Measurement
CT_A_CAP.mpet	A-UUT Bypass Capacitance
CT_A_SRP.mpet	A-UUT SRP
CT_A_HNP.mpet	A-UUT HNP
CT_A_ADP.mpet	A-UUT ADP
CT_A_LKG.mpet	A-UUT Leakage
CT_A_ST_OTG_ADP.mpet	ADP-Capable A-OTG State Transition Test
CT_A_ST_OTG_ADP_NOSRP.mpet	A-OTG Capable of ADP but not SRP, State Transition Test
CT_A_ST_OTG.mpet	Non ADP-Capable A-OTG State Transition Test
CT_A_ST_OTG_NOSESS.mpet	A-OTG with no Session Support State Transition Test
CT_A_ST_EH_ADP.mpet	ADP-Capable EH State Transition Test
CT_A_ST_EH_ADP_NOSRP.mpet	EH Capable of ADP but not SRP, State Transition Test
CT_A_ST_EH_SRP.mpet	Non ADP-Capable EH State Transition Test
CT_A_ST_EH_NOSESS.mpet	EH with no Session Support State Transition Test
CT_A_DevNoRes.mpet	A-UUT Device No Response

CT_A_UnsupDev.mpet	A-UUT Unsupported Device
CT_A_NoResHnp.mpet	A-UUT Dev No Resp for HNP En
CT_A_OTG_REP.mpet	A-UUT Checklist Report

OTG 2.0 B-UUT Tests	
CT_B_PUT.mpet	B-UUT Power-Up Tests
CT_B_VBUS.mpet	B-UUT Vbus Voltage and Current Measurement
CT_B_CAP.mpet	B-UUT Bypass Capacitance
CT_B_SRP.mpet	B-UUT SRP
CT_B_HNP.mpet	B-UUT HNP
CT_B_ADP.mpet	B-UUT ADP
CT_B_LKG.mpet	B-UUT Leakage
CT_B_ST_OTG_ADP_HNP_SRP.mpet	B-OTG, Capable of ADP/HNP/SRP, State Transition Test
CT_B_ST_OTG_HNP_SRP.mpet	B-OTG, Capable of HNP/SRP, State Transition Test
CT_B_ST_OTG_ADP_SRP.mpet	B-OTG, Capable of ADP/SRP, State Transition Test
CT_B_ST_OTG_SRP.mpet	B-OTG, Capable of SRP only, State Transition Test
CT_B_ST_OTG_NONE.mpet	B-OTG, Capable of no Protocols, State Transition Test
CT_B_ST_PER_ADP_SRP.mpet	Peripheral Only, ADP Capable, B-Device State Transition Test
CT_B_ST_PER_SRP.mpet	Peripheral Only, SRP Only Capable, B-Device State Transition Test
CT_B_ST_PER_NONE.mpet	Peripheral Only, no Protocols, B-Device State Transition Test
CT_B_DevNoRes.mpet	B-UUT Device No Response
CT_B_UnsupDev.mpet	B-UUT Unsupported Device
CT_B_NoResHnp.mpet	B-UUT Dev No Resp for HNP En
CT_B_OTG_REP.mpet	B-UUT Checklist Report

High Speed Electrical Tests	
HsEtm_0101.mpet	Test_SE0_NAK (Host)
HsEtm_0102.mpet	Test_J (Host)
HsEtm_0103.mpet	Test_K (Host)
HsEtm_0104.mpet	Test_Packet (Host)
HsEtm_0106.mpet	HS_HOST_PORT_SUSPEND_RESUME (Host)
HsEtm_0107.mpet	SINGLE_STEP_GET_DEV_DESC (Host)
HsEtm_0108.mpet	SINGLE_STEP_GET_DEV_DESC_DATA (Host)
HsEtm_01.mpet	Test_SE0_NAK (Device)
HsEtm_02.mpet	Test_J (Device)
HsEtm_03.mpet	Test_K (Device)
HsEtm_04.mpet	Test_Packet (Device)

BC1.2 Portable Device Tests (with Almost Weak Battery)	
CT_PD_WEAK_PREP.mpet	PD – Test Preparation – Almost Weak Battery
CT_PD_DCD_CS.mpet	PD - Data Contact Detect Test with Current Source
CT_PD_DCD_TO.mpet	PD - Data Contact Detect Test, Timer Only
CT_PD_DCP_Det.mpet	PD - DCP Detection Test
CT_PD_CDP_Det.mpet	PD - CDP Detection Test
CT_PD_SDP_Det.mpet	PD - SDP Detection Test
CT_PD_Dock_Det.mpet	PD - ACA-Dock Detection Test
CT_PD_ACA_A_Det.mpet	PD - ACA-A Detection Test
CT_PD_ACA_B_Det.mpet	PD - ACA-B Detection Test
CT_PD_ACA_C_Det.mpet	PD - ACA-C Detection Test
CT_PD_ACA_GND_Det.mpet	PD - ACA-GND Detection Test
CT_PD_WEAK_REP.mpet	PD - Checklist Report

BC1.2 Portable Device Tests (with Good Battery)	
CT_PD_GOOD_PREP.mpet	PD – Test Preparation – Good Battery
CT_PD_CMO_FS.mpet	PD - Common Mode Test Full Speed
CT_PD_CMO_HS.mpet	PD - Common Mode Test High Speed
CT_PD_GOOD_REP.mpet	PD - Checklist Report

BC1.2 Portable Device Tests (Dead Battery Provision)	
CT_PD_DBP.mpet	PD - Dead Battery Provision Test
CT_PD_REP_DBP.mpet	PD - Checklist Report, Dead Battery Provision

BC1.2 Micro-ACA (Separate Charger) Tests	
CT_UACA_CAL.mpet	Micro-ACA, Cable Calibration Pre-Test
CT_UACA_SDP_NOT.mpet	Micro-ACA, SDP to Chgr Port, Nothing to Acc. Port
CT_UACA_SDP_A.mpet	Micro-ACA, SDP to Chgr Port, A-device to Acc. Port
CT_UACA_SDP_B.mpet	Micro-ACA, SDP to Chgr Port, B-device to Acc. Port
CT_UACA_CHG_NOT.mpet	Micro-ACA, DCP or CDP to Chgr Port, Nothing to Acc. Port
CT_UACA_CHG_A.mpet	Micro-ACA, DCP or CDP to Chgr Port, A-device to Acc. Port
CT_UACA_CHG_B.mpet	Micro-ACA, DCP or CDP to Chgr Port, B-device to Acc. Port
CT_UACA_NOT_NOT.mpet	Micro-ACA, Nothing to Chgr Port, Nothing to Acc. Port
CT_UACA_NOT_A.mpet	Micro-ACA, Nothing to Chgr Port, A-device to Acc. Port
CT_UACA_NOT_B.mpet	Micro-ACA, Nothing to Chgr Port, B-device to Acc. Port
CT_UACA_CAP.mpet	Micro-ACA, Bypass Capacitance Test
CT_UACA_REP.mpet	Micro-ACA, Checklist Report

BC1.2 Micro-ACA (Combined Charger) Tests	
CT_UACAC_CAL.mpet	Micro-ACA (Combined Charger), Cable Calibration Pre-Test
CT_UACAC_VIT.mpet	Micro-ACA (Combined Charger), Voltage, Current and Transient Test
CT_UACAC_OFF_NOT.mpet	Micro-ACA (Combined Charger), Off, Nothing to Acc. Port
CT_UACAC_OFF_A.mpet	Micro-ACA (Combined Charger), Off, A-device to Acc. Port
CT_UACAC_OFF_B.mpet	Micro-ACA (Combined Charger), Off, B-device to Acc. Port
CT_UACAC_ON_NOT.mpet	Micro-ACA (Combined Charger), On, Nothing to Acc. Port
CT_UACAC_ON_A.mpet	Micro-ACA (Combined Charger), On, A-device to Acc. Port
CT_UACAC_ON_B.mpet	Micro-ACA (Combined Charger), On, B-device to Acc. Port
CT_UACAC_CAP.mpet	Micro-ACA (Combined Charger), Bypass Capacitance Test
CT_UACA_REP.mpet	Micro-ACA, Checklist Report

BC1.2 Standard-ACA (Separate Charger) Tests	
CT_SACA_CAL.mpet	Standard-ACA, Cable Calibration Pre-Test
CT_SACA_SDP_NOT.mpet	Standard-ACA, SDP to Chgr Port, Nothing to Acc. Port
CT_SACA_SDP_B.mpet	Standard-ACA, SDP to Chgr Port, B-device to Acc. Port
CT_SACA_CHG_NOT.mpet	Standard-ACA, DCP or CDP to Chgr Port, Nothing to Acc. Port
CT_SACA_CHG_B.mpet	Standard-ACA, DCP or CDP to Chgr Port, B-device to Acc. Port
CT_SACA_NOT_NOT.mpet	Standard-ACA, Nothing to Chgr Port, Nothing to Acc. Port
CT_SACA_NOT_B.mpet	Standard-ACA, Nothing to Chgr Port, B-device to Acc. Port
CT_SACA_CAP.mpet	Standard-ACA, Bypass Capacitance Test
CT_SACA_REP.mpet	Standard-ACA, Checklist Report

BC1.2 Standard-ACA (Combined Charger) Tests	
CT_SACAC_CAL.mpet	Standard-ACA (Combined Charger), Cable Calibration Pre-Test
CT_SACAC_VIT.mpet	Standard -ACA (Combined Charger), Voltage, Current and Transient Test
CT_SACAC_OFF_NOT_B.mpet	Standard-ACA (Combined Charger), Off, Nothing/B to Acc. Port
CT_SACAC_ ON_NOT_B.mpet	Standard-ACA (Combined Charger), On, Nothing/B to Acc. Port
CT_SACAC_CAP.mpet	Standard-ACA (Combined Charger), Bypass Capacitance Test
CT_SACA_REP.mpet	Standard-ACA, Checklist Report

BC1.2 DCP Tests	
CT_DCP_OVRSHT.mpet	DCP Overshoot and Undershoot Voltage Test
CT_DCP_V_I.mpet	DCP Voltage and Current Test
CT_DCP_HNDSHK.mpet	DCP – Handshaking Test
CT_DCP_R_C.mpet	DCP Resistance and Capacitance Tests
CT_DCP_REP.mpet	DCP - Checklist Report

BC1.2 CDP Tests	
CT_CDP_OVRSHT.mpet	CDP Overshoot and Undershoot Voltage Test
CT_CDP_V_I.mpet	CDP Output Voltage and Current Test
CT_CDP_HNDSHK.mpet	CDP Handshaking Test
CT_CDP_CMO_FS.mpet	CDP Ground Offset Test, Full Speed
CT_CDP_CMO_HS.mpet	CDP Ground Offset Test, High Speed
CT_CDP_REP.mpet	CDP - Checklist Report

BC1.2 SDP Tests	
CT_SDP_HNDSHK.mpet	SDP Handshaking Test
CT_SDP_REP.mpet	SDP - Checklist Report

BC1.2 Multiple Role Port (MRP) Tests	
CT_MRP_FUNC.mpet	MRP Handshaking Test
CT_MRP_REP.mpet	MRP - Checklist Report

BC1.2 ACA-Dock Tests	
CT_ACADK_OVRSHT.mpet	ACA-Dock Over and Undershoot Voltage Test
CT_ACADK_V_I.mpet	ACA-Dock Voltage and Current Test
CT_ACADK_NOT_POW.mpet	ACA-Dock Not powered Tests
CT_ACADK_REP.mpet	ACA-Dock Checklist Report

Files containing supporting subroutines and definitions	
enum_dev_inc.mpet	Enumerate as device support
enum_dev_cmo_inc.mpet	Enumerate as device support - CMO version
enum_host_inc.mpet	Enumerate as host support
includes_inc.mpet	General definitions
assert_inc.mpet	Assert Defaults
GetVbusOn_inc.mpet	Vbus support
user_entries.mpet	Generated automatically each time the test suite is run. Contains values which depend on the checklist.
CT_OTG_REP.mpet	OTG Checklist Report Core
CT_PD_REP.mpet	PD Checklist Report Core

Miscellaneous	
Calibrate.mpet	User Calibration Check
PetAuth.mdat	Authentication file to ensure that the test scripts have not been accidentally altered.

3 TECHNICAL DATA

3.1 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 PET.
- Windows XP (Service Pack 1 or better), Vista or Windows 7. 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.

3.2 Specifications USB-PET

Weight:	515 g
Dimensions:	190 x 140 x 52 mm
Temperature:	0°C - 40°C
Humidity:	20% - 80% non condensing
Power:	Zero mA from USB when powered externally. Using the external power supply (included) is the required mode of operation.

3.3 Maximum Capture File Size

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

3.4 Safety

CE compliant.

3.4.1 External Power Supply

The Packet-Master USB-PET must be connected to the provided external power supply. GraphicUSB will not allow you to run a script otherwise.

The external supply has the following specification:

Output Voltage	9V Regulated
Output Current	2A
Polarity	Centre Pin Positive

3.4.2 Trigger Specification

The PET has a BNC trigger output which can be controlled by the appropriate script command:

```
WriteToETB SW_TRIGGER 0 (or 1)
```

This sets the state of the signal to the corresponding logic level, for use in triggering an item of test equipment, also connected to the Signal Out BNC.

Characteristic	Value	Condition
Output Impedance (Ω)	50	
VOL max (V)	0.4	(zero current)
VOH min (V)	2.4	(zero current)

3.5 Setting and Measurement Accuracy

Setting and measurement accuracy is currently specified at better than 0.5% at full range values. In practice 12-bit A/D and D/A converters together with individual digital calibration on all settings and measurements means that much better accuracy is achieved. The actual accuracy obtained will be presented in the next version of the user manual.

Measurement Amplifier Parameters		
Measurement Circuit	**Gain**	**Time Constant (ms)**
VBUS_AB	0.5	0.15
DM_AB	1	0.15
DP_AB	1	0.15
ID_AB	1	0.15
IVBUS_AB (low)	200	10
IVBUS_AB (mid)	20	0.2
IVBUS_AB (high)	2	0.2
VBUS_IIL	1	0.15
VBUS_IVL	2	0.2

4 GraphicUSB SOFTWARE - Running Scripts

4.1 Introduction

The USB-PET is a comprehensive compliance tester, capable of emulating (and measuring) all the electrical conditions and protocol requirements of a USB host, a USB device, an OTG device, an Embedded Host, an OTG Peripheral-Only device, a Charging Downstream Port, or a Dedicated Charger Port. It can also perform a series of compliance tests on a Micro-ACA or Standard-ACA.

The PET is controlled by a script, which is flexible enough to allow complete emulation as a host or peripheral. A set of standard scripts is provided for confirming the operation of devices designed to meet the OTG 2.0 and/or Battery Charging 1.2 specifications.

The normal output from the PET is a text-based report file, originating from specific SendMessage commands in the scripts.

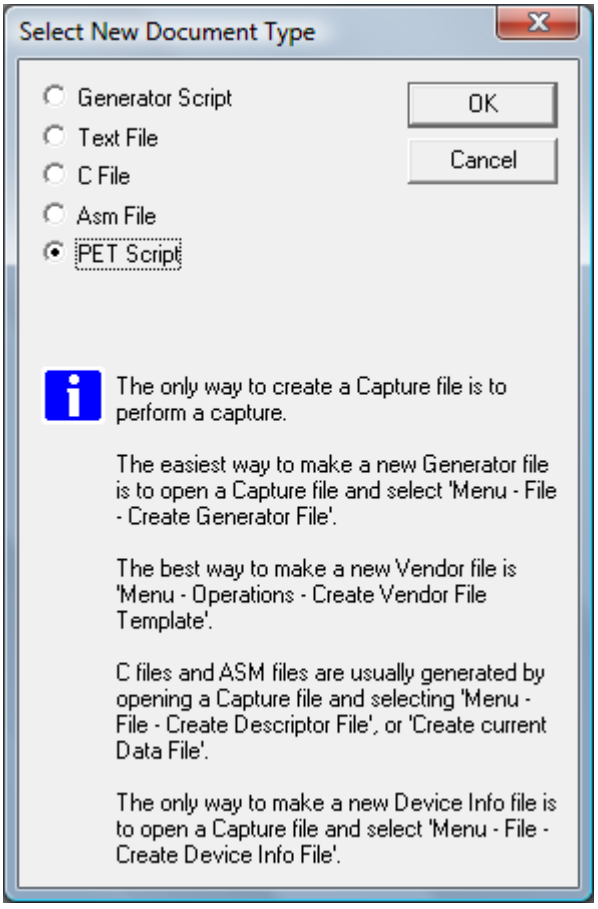
An additional output file may also be selected for output. This is a 'Capture' file, identical to one output by one of our analysers, such as the USB500 AG. The capture file reveals the exact sequence of events.

An alternative to using the standard scripts is to write your own. The following sections describe the procedure.

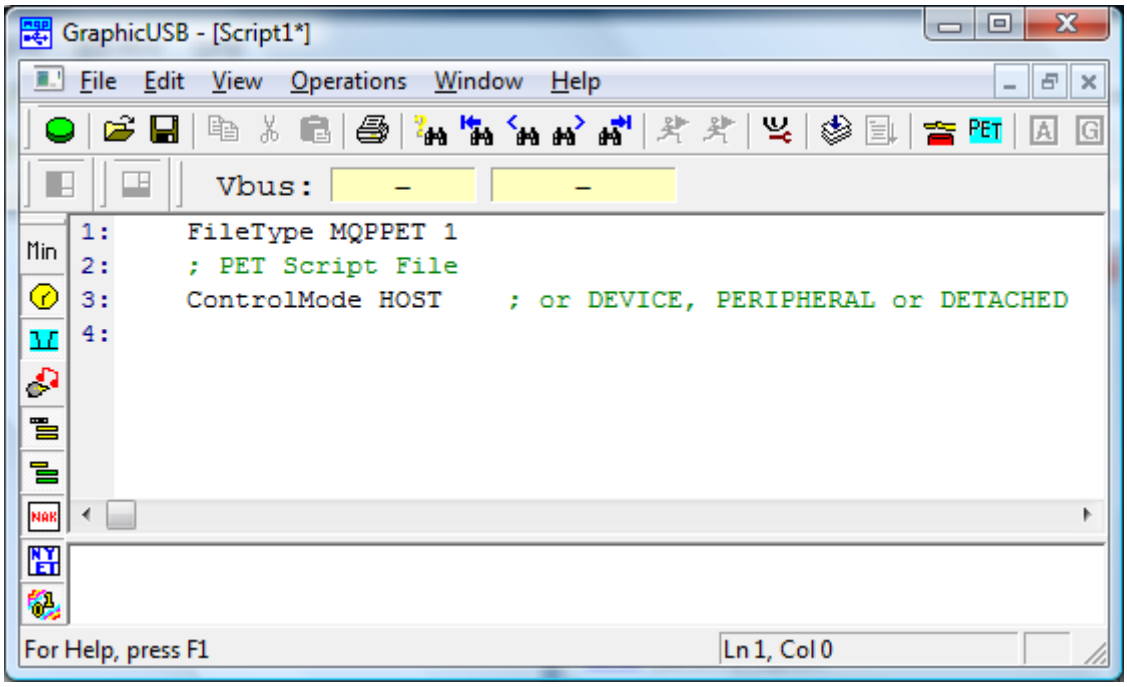
4.2 Creating a PET Script

Start by selecting menu item:

`File…New…`



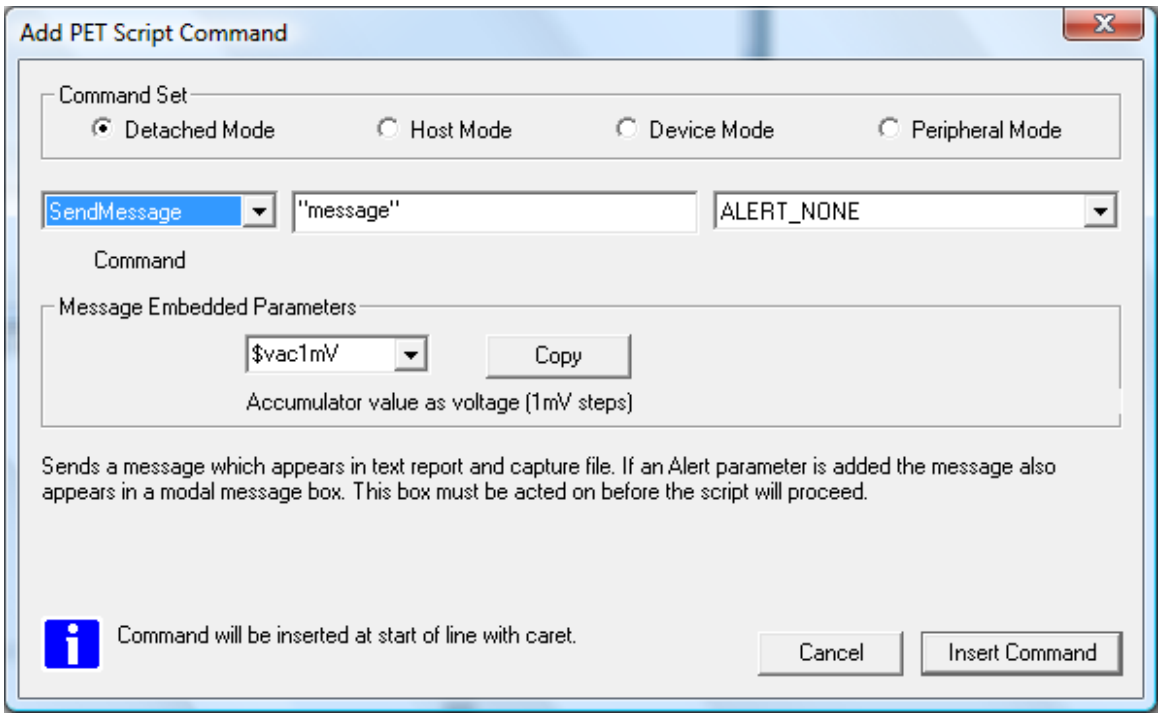
Click on the PET Script button, then OK. A barebones script is created for you to add to. This section describes only the mechanism of creating and compiling scripts. Later sections provide the information required to understand what to put in a script.



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

The easiest way to add a command is menu item:

`Edit…Insert Command… (or Ctrl/I)`



The procedure is:

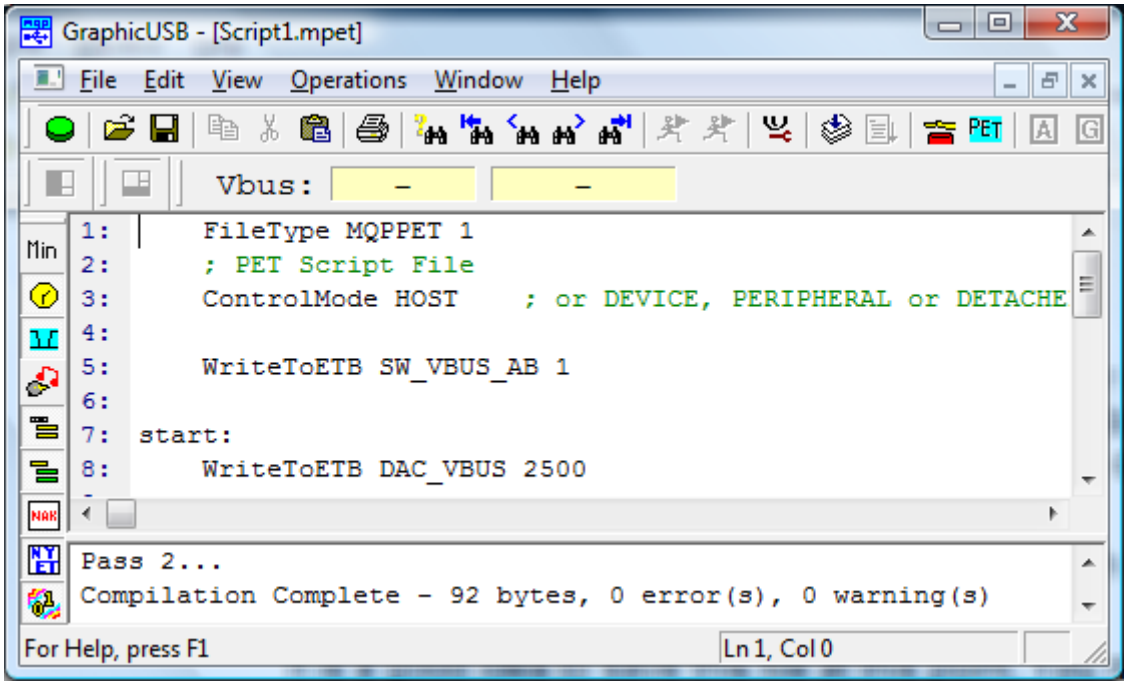
- Before entering the dialog, ensure that your caret is located at the start of the line you want to insert the command in front of.
- Then select the command required from the command selection box. The purpose of the command is explained underneath.

Now make appropriate selections in the other boxes to the right of the command. When happy with your selection, click on 'Insert Command'.

Using this method helps to get the correct number of parameters and the correct spelling for commands. All the available keywords are offered in this dialog, together with brief descriptions of the functionality of each command.

Continue to add commands until you have enough to perform the required task.

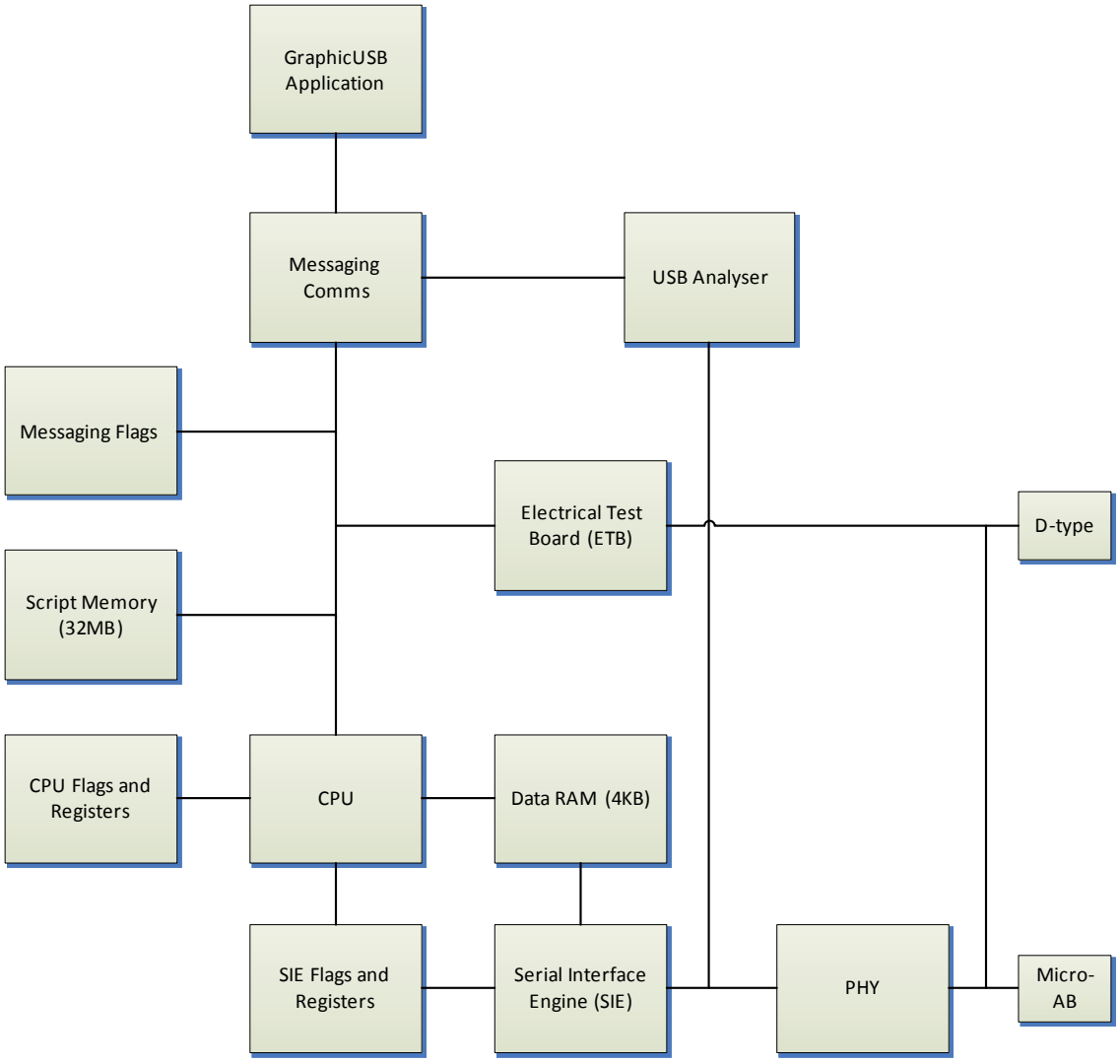
It is a good idea to save this file at this point. You may now try compiling the file using menu item File…Compile, or click on the 'Compile' icon in the toolbar .



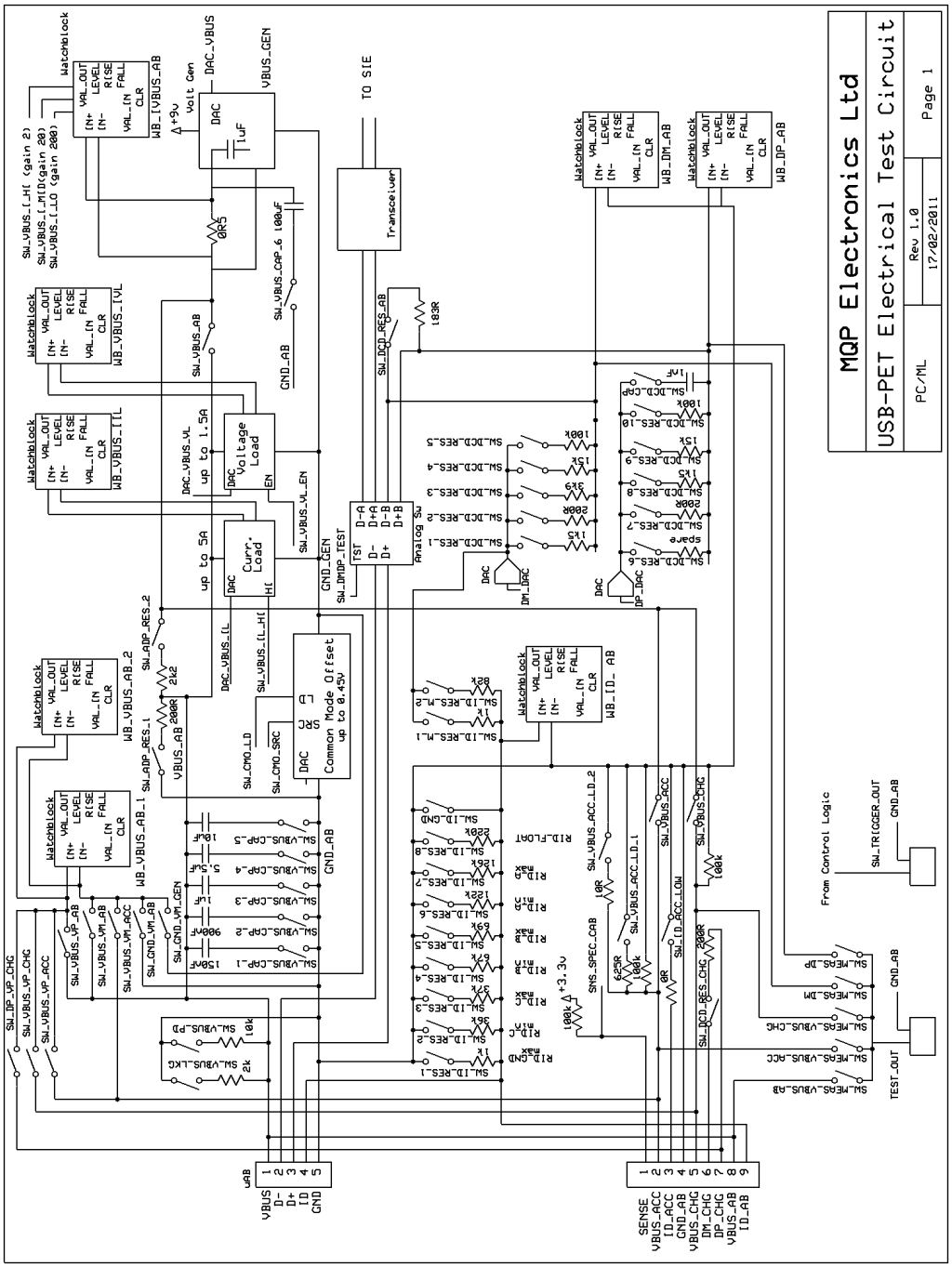
If there is an error, it will be announced in the lower pane. Double click on the error message to point at the line in the script containing the error. Alternatively press the F4 key to highlight the errors one at a time.

Edit your file, as required, checking its validity with frequent compilations. Then you are ready to run your PET script. Do this by using menu item File…Run, or click on the 'Run' icon  in the toolbar.

4.3 PET Processor Architecture



4.3.1 Electrical Test Board



4.3.2 Alphabetical Command List

Command	Parameters	Description
ADAR	<REG>	Add Register to A leaving the result in A.
ADIA	<16_bit_value>	Add the specified immediate value to A, leaving the result in A.
ANDIA	<16_bit_value>	Logical AND A with the specified immediate value, leaving the result in A.
CALL	<label>	Save the current PC on the hardware CALL stack, and start execution at the specified label.
ClearFlag	<FLAG>	Clear the specified flag to 0.
ControlMode	HOST | DEVICE | PERIPHERAL | DETACHED	This determines what the PET is emulating: Detached and Device modes connect no 15K pull-down resistors. Host mode connects pull-down resistors on both data lines. Peripheral mode connects a pull-down resistor only on D-. Except in the case of Detached mode, the appropriate Serial Interface Engine is automatically enabled.
CPAR	<REG>	Compare A with Register and set comparison flags accordingly. e.g. If the word contents of A is less than the Register the LessThanFlag is set.
CPIA	<16_bit_value>	Compare A with the specified immediate value, and set the condition flags accordingly. e.g. If the word contents of A is less than the immediate value, the LessThanFlag is set.
DATA0_1	-	Tells the Register Group 1 SIE buffer to expect or send a DATA0 packet next, as appropriate.
DATA1_1	-	Tells the Register Group 1 SIE buffer to expect or send a DATA1 packet next, as appropriate.
DATA2_1	-	Tells the Register Group 1 SIE buffer to expect or send a DATA2 packet next, as appropriate.
DATATOGGLE_1	-	Tells the Register Group 1 SIE buffer to toggle which data packet to expect or send next, as appropriate, between DATA0 and DATA1.
DATA0_2	-	Tells the Register Group 2 SIE buffer to expect or send a DATA0 packet next, as appropriate.
DATA1_2	-	Tells the Register Group 2 SIE buffer to expect or send a DATA1 packet next, as appropriate.
DATA2_2	-	Tells the Register Group 2 SIE buffer to expect or send a DATA2 packet next, as appropriate.
DATATOGGLE_2	-	Tells the Register Group 2 SIE buffer to toggle which data packet to expect or send next, as appropriate, between DATA0 and DATA1.
DATA0_3	-	Tells the Register Group 3 SIE buffer to expect or send a DATA0 packet next, as appropriate.
DATA1_3	-	Tells the Register Group 3 SIE buffer to expect or send a DATA1 packet next, as appropriate.
DATA2_3	-	Tells the Register Group 3 SIE buffer to expect or send a DATA2 packet next, as appropriate.

DATATOGGLE_3	-	Tells the Register Group 3 SIE buffer to toggle which data packet to expect or send next, as appropriate, between DATA0 and DATA1.
DATA0_4	-	Tells the Register Group 4 SIE buffer to expect or send a DATA0 packet next, as appropriate.
DATA1_4	-	Tells the Register Group 4 SIE buffer to expect or send a DATA1 packet next, as appropriate.
DATA2_4	-	Tells the Register Group 4 SIE buffer to expect or send a DATA2 packet next, as appropriate.
DATATOGGLE_4	-	Tells the Register Group 4 SIE buffer to toggle which data packet to expect or send next, as appropriate, between DATA0 and DATA1.
DJNZ	<label>	Decrement Count and if the result is not zero, jump to the specified label.
DnldToRam	<RamAddr> (<byte>….)	Store the specified sequence of bytes into successive data RAM locations, starting with the one specified. This command is allowed to span more than one line.
EndPeriod0	-	See the StartPeriodn_xxx commands
EndPeriod1	-	See the StartPeriodn_xxx commands
EndPeriod2	-	See the StartPeriodn_xxx commands
EndPeriod3	-	See the StartPeriodn_xxx commands
EndPeriod4	-	See the StartPeriodn_xxx commands
Equate	<symbol_name> <16-bit value>	Set the 16 bit value of a specified symbol.
FileType	MQPPET 1	Defines file type and version number. Must be set to MQPPET 1
HALT	-	Stop execution of script.
Idle		Do nothing for number of 60MHz clock cycles specified. Note: SOFs continue.
IdleSec		Do nothing for number of seconds specified. Note: SOFs continue.
InbufData0_0	-	Tells the Register Group 0 SIE IN buffer to expect a DATA0 packet next.
InbufData1_0	-	Tells the Register Group 0 SIE IN buffer to expect a DATA1 packet next.
InbufDataToggle_0	-	Tells the Register Group 0 SIE IN buffer to toggle which data packet to expect next, between DATA0 and DATA1.
Include	"filename.mpet"	Includes the text of a specified file at this point in the script. The file is assumed to be in the same directory as the script file, and should have the extension .mpet
Jump	<label>	Unconditionally continue execution at the address of the specified label.
JumpIfFalse	<FLAG> <label>	Jump to the specified label, if the specified flag is 0, else continue with the next instruction.
JumpIfTrue	<FLAG> <label>	Jump to the specified label, if the specified flag is 1, else continue with the next instruction
LDA	<RamAddr>	Load low byte of A from data RAM byte specified.
LDAR	<REG>	Load A from Register.
LDAW	<RamAddr>	Load A from word in data RAM specified.

LDAX	-	Load low byte of A from data RAM byte indexed by REGIndex, high byte of A becomes 0.
LDAXW	-	Load A from word in data RAM indexed by REGIndex.
LDIA	<16_bit_value>	Load the specified immediate value into A.
MDATA_1	-	Tells the Register Group 1 SIE buffer to send an MDATA packet next.
MDATA_2	-	Tells the Register Group 2 SIE buffer to send an MDATA packet next.
MDATA_3	-	Tells the Register Group 3 SIE buffer to send an MDATA packet next.
MDATA_4	-	Tells the Register Group 4 SIE buffer to send an MDATA packet next.
ORIA	<16_bit_value>	OR A with the specified immediate value, leaving the result in A.
OutbufData0_0	-	Tells the Register Group 0 SIE OUT buffer to transmit a DATA0 packet next.
OutbufData1_0	-	Tells the Register Group 0 SIE OUT buffer to transmit a DATA1 packet next.
OutbufDataToggle_0	-	Tells the Register Group 0 SIE OUT buffer to toggle which data packet to transmit next, between DATA0 and DATA1.
POP	-	POP the last 16 bit value PUSHed onto the data stack, back into the accumulator A.
PullupOff	-	**DEVICE or PERIPHERAL mode only** Disconnect pull-up resistor from data line. In High Speed, also cancel High Speed termination.
PullupOn	FULLSPEED | LOWSPEED	**DEVICE or PERIPHERAL mode.** Apply specified pull-up resistor to data line. **TEST Mode – HOST, DEVICE or PERIPHERAL Mode**. In high speed test mode, sets data lines to high speed J or K condition.
PUSH	-	The content of the accumulator A is pushed onto the hardware data stack. This stack has space for 128 words. A is not affected, and the data can later be restored using a POP command.
ReadFromETB	<ETB_addr>	Read 16 bit value from specified location on Electrical Test Board into A. This command is not available to the Packet-Master USB500 AG+.
ResetStart	-	**HOST mode only.** Tells SIE to start a USB reset. This will complete automatically, performing a HS handshake if the HsRequired flag is set. The flags HighSpeedSet and LowSpeedSet will be adjusted appropriately as a result of this operation, and should be tested to determine the speed negotiated.

Resume	-	**HOST mode** Tells SIE to start a USB resume. Apply K for 23-24 ms followed with LSEOP for correct completion (applies SE0 for 2 low speed bit times, followed by 1 bit time of J). **Device or Peripheral** Apply K for 2-3 ms.
Retries	SAMEFRAME | NEXTFRAME	**HOST mode only.** Specifies when a NAKed packet should be retried.
RETURN	-	Recover a saved PC from the hardware CALL stack, and continue execution at that address.
SBAR	<REG>	Subtract Register from A.
SBIA	<16_bit_value>	Subtract the specified immediate value from A.
SendInfo	"<message>"	Sends a message which appears in text report and capture file. It has the advantage that it will not dismiss a modal dialog. No Alert parameter is permitted.
SendMessage	"<message>" [<alert_param>]	Sends a message which appears in text report and capture file. If an Alert parameter is added the message also appears in a modal message box. This box must be acted on before the script will proceed.
SendPrompt	"<message>" <alert_param>	Sends a message which appears in text report and capture file. The message also appears in a modeless message box. This type of message box does not prevent the script from proceeding, and will automatically be dismissed if a further message or prompt is received.
SendReport	"<message>" [<alert_param>]	Sends a message which appears only in text report. If an Alert parameter is added the message also appears in a modal message box. Such a box must be acted on before the script will proceed.
SendReportInfo	"<message>"	Sends a message which appears only in text report. It has the advantage that it will not dismiss a modal dialog. No Alert parameter is permitted.
SendValue	<value_param>	Sends a value, or an operation request relating to a value or values. This type of 'message' does not appear in the text report or the capture file, although parameters set by this operation can subsequently in normal messages.
SetCount	<16_bit_value>	Sets the counter used by DNJZ command to a count from 1 to 65535.
SetFlag	<FLAG>	Set the specified flag to 1.
SHLA	-	Shifts A left one bit. Zero bit enters from right.
SHRA	-	Shifts A right one bit. Zero bit enters from left.
SMIAR	<REG>	If Register is less than A, swap A and Register.
SOFs	<16_bit_value>	**HOST mode only.** Peform no transactions until the specified number of SOFs has been sent.
STA	<RamAddr>	Store low byte of A into data RAM byte specified.
STAR	<REG>	Store A to Register.
StartPeriod0_1us	-	Starts the PERIOD0 timer using a 1us clock. The timer can be stopped using EndPeriod0. The time measured can be acquired from register PERIOD0. The maximum time count is 65535.

StartPeriod1_1us	-	Starts the PERIOD1 timer using a 1us clock. The timer can be stopped using EndPeriod1. The time measured can be acquired from register PERIOD1. The maximum time count is 65535.
StartPeriod2_1us	-	Starts the PERIOD2 timer using a 1us clock. The timer can be stopped using EndPeriod2. The time measured can be acquired from register PERIOD2. The maximum time count is 65535.
StartPeriod3_1us	-	Starts the PERIOD3 timer using a 1us clock. The timer can be stopped using EndPeriod3. The time measured can be acquired from register PERIOD3. The maximum time count is 65535.
StartPeriod4_1us	-	Starts the PERIOD4 timer using a 1us clock. The timer can be stopped using EndPeriod4. The time measured can be acquired from register PERIOD4. The maximum time count is 65535.
StartPeriod0_10us	-	Starts the PERIOD0 timer using a 10us clock. The timer can be stopped using EndPeriod0. The time measured can be acquired from register PERIOD0. The maximum time count is 65535.
StartPeriod1_10us	-	Starts the PERIOD1 timer using a 10us clock. The timer can be stopped using EndPeriod1. The time measured can be acquired from register PERIOD1. The maximum time count is 65535.
StartPeriod2_10us	-	Starts the PERIOD2 timer using a 10us clock. The timer can be stopped using EndPeriod2. The time measured can be acquired from register PERIOD2. The maximum time count is 65535.
StartPeriod3_10us	-	Starts the PERIOD3 timer using a 10us clock. The timer can be stopped using EndPeriod3. The time measured can be acquired from register PERIOD3. The maximum time count is 65535.
StartPeriod4_10us	-	Starts the PERIOD4 timer using a 10us clock. The timer can be stopped using EndPeriod4. The time measured can be acquired from register PERIOD4. The maximum time count is 65535.
StartPeriod0_100us	-	Starts the PERIOD0 timer using a 100us clock. The timer can be stopped using EndPeriod0. The time measured can be acquired from register PERIOD0. The maximum time count is 65535.
StartPeriod1_100us	-	Starts the PERIOD1 timer using a 100us clock. The timer can be stopped using EndPeriod1. The time measured can be acquired from register PERIOD1. The maximum time count is 65535.
StartPeriod2_100us	-	Starts the PERIOD2 timer using a 100us clock. The timer can be stopped using EndPeriod2. The time measured can be acquired from register PERIOD2. The maximum time count is 65535.
StartPeriod3_100us	-	Starts the PERIOD3 timer using a 100us clock. The timer can be stopped using EndPeriod3. The time measured can be acquired from register PERIOD3. The maximum time count is 65535.
StartPeriod4_100us	-	Starts the PERIOD4 timer using a 100us clock. The timer can be stopped using EndPeriod4. The time measured can be acquired from register PERIOD4. The maximum time count is 65535.

StartPeriod0_1ms	-	Starts the PERIOD0 timer using a 1ms clock. The timer can be stopped using EndPeriod0. The time measured can be acquired from register PERIOD0. The maximum time count is 65535.
StartPeriod1_1ms	-	Starts the PERIOD1 timer using a 1ms clock. The timer can be stopped using EndPeriod1. The time measured can be acquired from register PERIOD1. The maximum time count is 65535.
StartPeriod2_1ms	-	Starts the PERIOD2 timer using a 1ms clock. The timer can be stopped using EndPeriod2. The time measured can be acquired from register PERIOD2. The maximum time count is 65535.
StartPeriod3_1ms	-	Starts the PERIOD3 timer using a 1ms clock. The timer can be stopped using EndPeriod3. The time measured can be acquired from register PERIOD3. The maximum time count is 65535.
StartPeriod4_1ms	-	Starts the PERIOD4 timer using a 1ms clock. The timer can be stopped using EndPeriod4. The time measured can be acquired from register PERIOD4. The maximum time count is 65535.
StartTimer0	<60MHzClks>	Sets Timer0 to the number of 60MHz clock cycles specified and clears the TIMEOUT0 flag. When the timer expires it sets the TIMEOUT0 flag. Maximum permitted value is 2147483647 corresponding to a time of ca. 35 seconds
StartTimer0Sec	<Seconds>	Sets Timer0 to the number of 60MHz clock cycles specified and clears the TIMEOUT0 flag. When the timer expires it sets the TIMEOUT0 flag. Maximum permitted value is 6000 corresponding to a time of 100 minutes.
StartTimer1	<60MHzClks>	Sets Timer1 to the number of 60MHz clock cycles specified and clears the TIMEOUT1 flag. When the timer expires it sets the TIMEOUT1 flag.
StartTimer1Sec	<Seconds>	Sets Timer1 to the number of seconds specified and clears the TIMEOUT1 flag. When the timer expires it sets the TIMEOUT1 flag.
StartTimer2	<60MHzClks>	Sets Timer2 to the number of 60MHz clock cycles specified and clears the TIMEOUT2 flag. When the timer expires it sets the TIMEOUT2 flag.
StartTimer2Sec	<Seconds>	Sets Timer2 to the number of seconds specified and clears the TIMEOUT2 flag. When the timer expires it sets the TIMEOUT2 flag.
STAW	<RamAddr>	Store both bytes of A into data RAM word specified. Low byte goes into address specified, high byte into next higher address.
STAX	-	Store low byte of A into data RAM byte indexed by REGIndex.
STAXW	-	Store both bytes of A into data RAM word indexed by REGIndex.
SuspendStart	-	HOST mode only. Tells SIE to start a USB suspend.
SWAPAR		

TransmitTestPacket	-	Starts the transmission of the USB specified high speed test packet. High speed must already be engaged, and the SUPPRESSSOFS flag must be set if emulating a host.
VBUSOFF	-	This is a command, which is only available to the Packet-Master USB500 AG+ when it is operating in 'Protocol Tester' mode. It switches off the supply to VBUS on the 'Device Under Test' sockets.
VBUSON	-	This is a command, which is only available to the Packet-Master USB500 AG+ when it is operating in 'Protocol Tester' mode. It causes a nominal 5V to be connected to VBUS on the 'Device Under Test' sockets.
WriteToETB	<ETB_addr> <16_bit_value>	Write specified 16 bit value specifed, to specified Electrical Test Board location. This command is not available to the Packet-Master USB500 AG+.
WriteToETBAcc	<ETB_addr>	Write specified 16 bit value from A to specified Electrical Test Board location. This command is not available to the Packet-Master USB500 AG+.

4.3.3 Parameter Types Used in Command List

Parameter Type	Description
"<message>"	Text of a single line of message for display in the report file, and in some cases in the capture file. It is currently recommended that the length of this text be limited to 80 characters for best results.
<alert_param>	The available *Alert* parameters are listed in a section below. The parameter describes the buttons which will be displayed along with the message box, to allow the user to select the appropriate action. Clicking on a button results in the setting of the appropriate response flag as defined in a section below. All response flags are automatically cleared when the SendMessage command is executed.
<16_bit_value>	A decimal number from 0 to 65535, or a hexadecimal number from 0x0000 to 0xFFFF.
<60MHzClks>	A decimal number from 0 to 2147483647 (corresponding to a time of ca. 35 seconds).
<byte>	A decimal number from 0 to 255, or a hexadecimal number from 0x00 to 0xFF.
<ETB_addr>	The address of an Electrical Test Board element. A complete list of addresses is listed in a section below.
<FLAG>	The name of a flag. A complete list of flags is listed in a section below.
<label>	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 label is used as the target of a `Jump` command or a `DJNZCount` command.
<RamAddr>	A number from 0x000 to 0xFFF.
<REG>	The name of a register. A complete list of registers is listed in a section below.
<Seconds>	A decimal number from 0 to 6000

<symbol_name>	Symbols are defined in the Syntax section. They are names for values defined by an 'Equate'.
<value_param>	See section 'SendValue Parameters' for a complete list.

4.3.4 CPU Registers

Register Name	Description
A	16 bit accumulator.
REGB	16 bit register which can be used in conjunction with A to assist with data manipulation.
REGIndex	Points at a RAM buffer address. Used in conjunction with commands ADAR, CPAR, LDAR, SBAR, SMIAR and STAR.
PC	Program Counter. Not directly accessible.
Call Stack	8 level hardware CALL stack. Not directly accessible.
Data Stack	128 level 16-bit wide data stack. Accessed via PUSH and POP commands.
Timer0	Timer accessed by commands StartTimer0 and StartTimer0Sec and flag Timeout0.
Timer1	Timer accessed by commands StartTimer1 and StartTimer1Sec and flag Timeout1.
Timer2	Timer accessed by commands StartTimer2 and StartTimer2Sec and flag Timeout2.
REGPeriod0	16-bit timer accessed by commands StartPeriod0_xxx and EndPeriod0. Stops counting at 65,535 if not stopped before.
REGPeriod1	16-bit timer accessed by commands StartPeriod1_xxx and EndPeriod1. Stops counting at 65,535 if not stopped before.
REGPeriod2	16-bit timer accessed by commands StartPeriod2_xxx and EndPeriod2. Stops counting at 65,535 if not stopped before.
REGPeriod3	16-bit timer accessed by commands StartPeriod3_xxx and EndPeriod3. Stops counting at 65,535 if not stopped before.
REGPeriod4	16-bit timer accessed by commands StartPeriod40_xxx and EndPeriod4. Stops counting at 65,535 if not stopped before.
Count	Counter accessed by commands SetCount and DJNZ.

4.3.5 SIE Registers

Register Name	Description
General Purpose Registers	
REGState	Simple storage location used to keep track of task number being performed while managing configured state behaviour during device mode enumeration.
REGSofsRcvd	**Device or Peripheral Mode** A read only register which is set by the hardware after performing a SOF reception test. The test is started by setting flag StartSOFTest, and finished when enough time has elapsed to have received 250 SOFs at either FS or HS as appropriate. At this point the flag SOFTESTCOMPLETE is set, and REGSofsRcvd contains the number of SOFs correctly received.
Endpoint Group 0-5 Registers	
REGAddr (Write only)	**Host, Device or Peripheral Mode** Used as device address for all transactions. Lowest 7 bits only are used.

REGHubAddr	**Host Mode** Used in conjunction with all 5 endpoint groups to define the hub address used in split transactions. Lowest 7 bits only are used. [not yet implemented]
REGPortAddr	**Host Mode** Used in conjunction with all 5 endpoint groups to define the hub port address used in split transactions. Lowest 7 bits only are used. [not yet implemented]
REGStatus	**Device or Peripheral Mode** Contains information about the transaction currently being processed. In general, one flag will be set and the endpoint involved will be specified. REGStatus is implemented as a 16-deep queue of status words in the order in which they occur. The queue is advanced by clearing the SETUPRCVD, INSENT, OUTRCVD, STALLED, ILLEGALNAK, TRANSACTIONTIMEDOUT or LPMRCVD flag as appropriate. Note that some of these flags are used in HOST mode as well, but that REGStatus has no meaning during HOST mode, and should not be read. Writing any value to REGStatus will clear the queue completely. This should always be done after a USB reset, to ensure that previous transaction information has been cleared. If NoRetryOnTimeout flag is not set, then clearing TRANSACTIONTIMEDOUT has no effect on the queue. Bit 10 – 1 if LPMRCVD, else 0 If NoRetryOnTimeout flag set: Bit 9 – 1 if TRANSACTIONTIMEDOUT, else 0 If NoRetryOnTimeout flag not set: Bit 9 – 0 Bit 8 – 1 if ILLEGALNAK, else 0 Bit 7 – 1 if STALLED, else 0 Bit 6 – 1 if SETUPRCVD, else 0 Bit 5 – 1 if OUTRCVD, else 0 Bit 4 – 1 if INSENT, else 0 Bits 3:0 – Endpoint number
Endpoint Group 0 Registers	
REGBytePtrOut0	**Host, Device or Peripheral Mode** Points at start of RAM buffer for OUT transactions on control endpoint 0. Used by host or device for OUT direction transaction.
REGPacketSize0	**Host, Device or Peripheral Mode** Simple storage location for endpoint 0 packet size.
REGLength	**Host, Device or Peripheral Mode** Simple storage location for length word read from SETUP
REGBytePtrSetup	**Host, Device or Peripheral Mode** Points at start of RAM buffer for SETUP transactions on control endpoint 0. Default value is zero. This results in unexpectedly early data (device or peripheral mode) being placed in a known location. Remember that SETUPs cannot be NAKed or STALLed.
REGBytePtrIn0	**Host, Device or Peripheral Mode** Points at start of RAM buffer for IN transactions on control endpoint 0. Used by host or device for IN direction transaction.
REGBytesLeftIn0	**Host, Device or Peripheral Mode** Simple storage location for number of untransmitted or unreceived bytes.

REGCountIn0	**Host Mode** Used by 'IN Transaction circuit'. Must be programmed with the maximum number of bytes expected in the transaction. After the transaction, contains the number of bytes received. This cannot exceed the max number of bytes expected. **Device or Peripheral Mode** Used by 'IN Transaction circuit'. Word containing number of bytes to send.
REGInCtrl0	**Host, Device or Peripheral Mode** Used by 'IN Buffer'. Bit 0: Enable Bit1: STALL (Device or Peripheral) Bit 2: reserved (0) Bit 3: Split [not yet implemented] Bit 4: SC [not yet implemented] Bit 5: Speed [not yet implemented] Bit 6: End [not yet implemented] Bit 7: Don't Respond*
REGOutCtrl0	**Host, Device or Peripheral Mode** Used by 'OUT Buffer'. Bit 0: Enable Bit1: STALL (Device or Peripheral) Bit 2: PING (Host and HS only) Bit 3: Split [not yet implemented] Bit 4: SC [not yet implemented] Bit 5: Speed [not yet implemented] Bit 6: End [not yet implemented] Bit 7: Don't Respond*
REGSetupCtrl (WRITE ONLY)	**Host, Device or Peripheral Mode** Used by 'Setup Transaction Engine'. Bit 0: Enable (Host only) Bits 1-2: reserved (0) Bit 3: Split [not yet implemented] Bit 4: SC [not yet implemented] Bit 5: Speed [not yet implemented] Bit 6: End [not yet implemented] Bit 7: Don't Respond*
REGCountOut0	**Device or Peripheral Mode** Used by 'OUT Transaction circuit'. Must be programmed with the maximum number of bytes expected in the transaction. After the transaction, contains the number of bytes received. This cannot exceed the max number of bytes expected. **Host** Used by 'OUT Transaction circuit'. Word containing number of bytes to send. (Note: number of device SETUP transaction bytes stored in RAM is fixed at 8 by hardware.)
REGBytesLeftOut0	Simple storage location for number of untransmitted or unreceived OUT bytes.
Endpoint Group 1 Registers	

REGEndp1	Endpoint to be used by this register group as host or as device. Bit 7 – 1=IN, 0=OUT Bits 3:0 - Endp
REGType1	Type of transfer for use by this endpoint. Bits 1:0 00 = Control 01 = Isochronous 10 = Bulk 11 = Interrupt
REGBytePtr1	**Host, Device or Peripheral Mode** Points at start of RAM buffer for IN or OUT transactions on endpoint Group 1.
REGBytesLeft1	Simple storage location to keep track of transfer data size.
REGCount1	Device or Peripheral Mode Used by 'OUT Transaction circuit'. Must be programmed with the maximum number of bytes expected in the transaction. After the transaction, contains the number of bytes received. This cannot exceed the max number of bytes expected. **Used by 'IN Transaction circuit'. Word containing number of bytes to send.** Host Used by 'OUT Transaction circuit'. Word containing number of bytes to send. Used by 'IN Transaction circuit'. Must be programmed with the maximum number of bytes expected in the transaction. After the transaction, contains the number of bytes received. This cannot exceed the max number of bytes expected.
REGCtrl1	**Host, Device or Peripheral Mode** Used by 'OUT Buffer'. Bit 0: Enable Bit1: STALL (**Device or Peripheral**) Bit 2: PING (**Host and HS OUT only**) Bits 3-6: reserved (0) Bit 7: Don't Respond*
REGPacketSize1	Simple storage location for this Register Group packet size.
REGResult1	**Host, Device or Peripheral Mode** Bits 15:2 [reserved] Bits 1:0 00 = DATA0 received 01 = DATA1 received 10 = DATA2 received 11 = MDATA received
Endpoint Group 2 Registers	
REGEndp2	Endpoint to be used by this register group as host or as device. Bit 7 – 1=IN, 0=OUT Bits 3:0 - Endp

REGType2	Type of transfer for use by this endpoint. Bits 1:0 00 = Control 01 = Isochronous 10 = Bulk 11 = Interrupt
REGBytePtr2	**Host, Device or Peripheral Mode** Points at start of RAM buffer for IN or OUT transactions on endpoint Group 2.
REGBytesLeft2	Simple storage location to keep track of transfer data size.
REGCount2	Device or Peripheral Mode Used by 'OUT Transaction circuit'. Must be programmed with the maximum number of bytes expected in the transaction. After the transaction, contains the number of bytes received. This cannot exceed the max number of bytes expected. **Used by 'IN Transaction circuit'. Word containing number of bytes to send.** Host Used by 'OUT Transaction circuit'. Word containing number of bytes to send. Used by 'IN Transaction circuit'. Must be programmed with the maximum number of bytes expected in the transaction. After the transaction, contains the number of bytes received. This cannot exceed the max number of bytes expected.
REGCtrl2	**Host, Device or Peripheral Mode** Used by 'OUT Buffer'. Bit 0: Enable Bit1: STALL (**Device or Peripheral**) Bit 2: PING (**Host and HS OUT only**) Bits 3-6: reserved (0) Bit 7: Don't Respond*
REGPacketSize2	Simple storage location for this Register Group packet size.
REGResult2	**Host, Device or Peripheral Mode** Bits 15:2 [reserved] Bits 1:0 00 = DATA0 received 01 = DATA1 received 10 = DATA2 received 11 = MDATA received
Endpoint Group 3 Registers	
REGEndp3	Endpoint to be used by this register group as host or as device. Bit 7 – 1=IN, 0=OUT Bits 3:0 - Endp
REGType3	Type of transfer for use by this endpoint. Bits 1:0 00 = Control 01 = Isochronous 10 = Bulk 11 = Interrupt

REGBytePtr3	**Host, Device or Peripheral Mode** Points at start of RAM buffer for IN or OUT transactions on endpoint Group 3.
REGBytesLeft3	Simple storage location to keep track of transfer data size.
REGCount3	Device or Peripheral Mode Used by 'OUT Transaction circuit'. Must be programmed with the maximum number of bytes expected in the transaction. After the transaction, contains the number of bytes received. This cannot exceed the max number of bytes expected. **Used by 'IN Transaction circuit'. Word containing number of bytes to send.** Host Used by 'OUT Transaction circuit'. Word containing number of bytes to send. Used by 'IN Transaction circuit'. Must be programmed with the maximum number of bytes expected in the transaction. After the transaction, contains the number of bytes received. This cannot exceed the max number of bytes expected.
REGCtrl3	**Host, Device or Peripheral Mode** Used by 'OUT Buffer'. Bit 0: Enable Bit1: STALL (**Device or Peripheral**) Bit 2: PING (**Host and HS OUT only**) Bits 3-6: reserved (0) Bit 7: Don't Respond*
REGPacketSize3	Simple storage location for this Register Group packet size.
REGResult3	**Host, Device or Peripheral Mode** Bits 15:2 [reserved] Bits 1:0 00 = DATA0 received 01 = DATA1 received 10 = DATA2 received 11 = MDATA received
Endpoint Group 4 Registers	
REGEndp4	Endpoint to be used by this register group as host or as device. Bit 7 – 1=IN, 0=OUT Bits 3:0 - Endp
REGType4	Type of transfer for use by this endpoint. Bits 1:0 00 = Control 01 = Isochronous 10 = Bulk 11 = Interrupt
REGBytePtr4	**Host, Device or Peripheral Mode** Points at start of RAM buffer for IN or OUT transactions on endpoint Group 4.
REGBytesLeft4	Simple storage location to keep track of transfer data size.

REGCount4	Device or Peripheral Mode Used by 'OUT Transaction circuit'. Must be programmed with the maximum number of bytes expected in the transaction. After the transaction, contains the number of bytes received. This cannot exceed the max number of bytes expected. **Used by 'IN Transaction circuit'. Word containing number of bytes to send.** Host Used by 'OUT Transaction circuit'. Word containing number of bytes to send. Used by 'IN Transaction circuit'. Must be programmed with the maximum number of bytes expected in the transaction. After the transaction, contains the number of bytes received. This cannot exceed the max number of bytes expected.
REGCtrl4	**Host, Device or Peripheral Mode** Used by 'OUT Buffer'. Bit 0: Enable Bit1: STALL (**Device or Peripheral**) Bit 2: PING (**Host and HS OUT only**) Bits 3-6: reserved (0) Bit 7: Don't Respond*
REGPacketSize4	Simple storage location for this Register Group packet size.
REGResult4	**Host, Device or Peripheral Mode** Bits 15:2 [reserved] Bits 1:0 00 = DATA0 received 01 = DATA1 received 10 = DATA2 received 11 = MDATA received
Extension Group Registers	
REGExtCtrl	**Host, Device or Peripheral Mode** Used by 'Extension Transaction Buffer'. Bit 0: Enable Bit1: STALL (Device or Peripheral) Bit 2: NYET (Device or Peripheral) Bits 6:3: reserved (0) Bit 7: Don't Respond (fot testing behaviour on non-response) Bits 10:8: 000 = LPM Others Reserved
REGExtCount	**Host** Used by 'Extension Transaction Buffer'. Word containing number of bytes to send. **Device or Peripheral Mode** Used by 'Extension Transaction Buffer'. Must be programmed with the maximum number of bytes expected in the transaction. After the transaction, contains the number of bytes received. This cannot exceed the number of bytes expected.

REGExtBytePtr	**Host, Device or Peripheral Mode** Points at start of RAM buffer for Extension Transaction.

4.3.6 Flags

All flags are automatically cleared to zero at the start of running a script.

4.3.6.1 CPU Flags

Flag Name	Function
Timeout0	Set by hardware when TIMER0 expires. Can be cleared using ClearFlag instruction.
Timeout1	Set by hardware when TIMER1 expires. Can be cleared using ClearFlag instruction.
Timeout2	Set by hardware when TIMER2 expires. Can be cleared using ClearFlag instruction.
LessThan	This condition flag is set by a CPIA/CPAR instruction if the word contents of A is less than the referenced value.
GreaterThan	This condition flag is set by a CPIA/CPAR instruction if the word contents of A is greater than the referenced value.
Equals	This condition flag is set/cleared by a CPIA/CPAR instruction depending on whether the referenced value matches the word contents of A

4.3.6.2 SIE Flags - Control

Flag Name	Function
HsRequired	Set this flag to indicate to the SIE that a High Speed connection should be attempted. This will determine whether chirps are sent.
InhibitCapture	This flag, when set to a 1, prevents USB events being included in the capture file. SendMessage events are still added to the capture file. The flag is useful to prevent overload when generating or receiving high speed test packets.
BDevice	**Host, Peripheral and Device Mode** This must be set appropriately at start of script to indicate to the SIE whether UUT is an A-device or a B-device. Cleared = A-device Set = B-device The flag is used to ensure the correct duration for the connect debounce timing during HNP.
NoRetryOnTimeout	**Host Mode** Prevents SIE from retrying any transaction, if the transaction times out. Prevents SIE from leaving IN engine enabled if the transaction times out.

Test Mode Control	
EnableTestMode	**Device and Peripheral Mode** When set to a 1, if chirp handshake fails, enters permanent test mode state, with HS termination. **Host, Peripheral and Device Mode** When set to a 1 (during HS connected state) allows the commands PullupOn FULLSPEED and PullupOn LOWSPEED to set the high speed terminated J and K conditions respectively.
SuppressSOFs	**Host Mode** When set to a 1, this prevents generated SOFs from being sent to the bus, whilst retaining a terminated high speed state.
StartSOFTest	**Device or Peripheral Mode** See REGSofsRcvd register description for details.
EnableLineStateDetect	**Host, Device and Peripheral Mode** When set to a 1 (during HS connected state) allows the flags DplusHigh, DminusHigh and SE0 to indicate the presence of J, K and SE0 states on the bus. It sets the transceiver to a state (also used for chirps) which disables bit stuffing and NRZI to allow the detection of line state. This flag has no effect unless EnableTestMode is also set.

4.3.6.3 SIE Flags – Status

Flag Name	Function
Host Mode Transaction StatusFlags	
SetupSent	**Host Mode** Flag set by the SIE to indicate that a Setup transaction has been sent.
InRcvd	**Host Mode** Flag set by the SIE to indicate that an In transaction has been received. Not set if Stalled.
OutSent	**Host Mode** Flag set by the SIE to indicate that an Out transaction has been sent.
LpmSent	**Host Mode** Flag set by the SIE to indicate that an LPM transaction has been sent.
DataToggleError	**Host Mode** Set by hardware if an IN transaction received the wrong data toggle. The InRcvd flag will not be set in this case.
NAKed	**Host Mode** Set by hardware if legally NAKed, and the behaviour for the current transaction is not to retry. In this case then the completion flag such as OutSent or InRcvd or SetupSent will not be set.

ERRed	**Host Mode** Set by hardware if ERR received as the result of a complete split transaction. In this case then the completion flag such as OutSent or InRcvd or SetupSent will not be set.
REGStatus Flags. In Device or Peripheral modes, the next seven flags are presented to the controller in the order received to prevent transaction synchronization problems. This queuing mechanism is invisible to the user. It is good practice to clear the queue immediately after a USB reset, when operating as a device or peripheral, by writing (any value) to REGStatus. The flags are also available in REGStatus, along with the relevant endpoint number. The queue is advanced by clearing the flag at the top of the queue. See also the description of word-wide register REGStatus. In Host mode, any flags valid in that mode are **not** queued, as only one transaction can be initiated at a time, and its details are known.	
Device and Peripheral ModeTransaction Status Flags	
SetupRcvd	**Device or Peripheral Mode** Flag set by the SIE to indicate that a Setup transaction has been received. The flag must be cleared in software after processing its occurrence. See REGStatus Flags description above. When this flag is placed in the queue, the registers REGInCtrl0 and REGOutCtrl0 are automatically cleared, to cancel any pending control transfer transactions.
InSent	**Device or Peripheral Mode** Flag set by the SIE to indicate that an IN transaction has been successfully sent. The flag must be cleared in software after processing its occurrence. See REGStatus Flags description above.
OutRcvd	**Device or Peripheral Mode** Flag set by the SIE to indicate that an OUT transaction has been successfully received. The flag must be cleared in software after processing its occurrence. See REGStatus Flags description above.
LpmRcvd	**Device or Peripheral Mode** This flag is set by the SIE to indicate that an LPM transaction has been successfully completed. The flag must be cleared in software after processing its occurrence. See REGStatus Flags description above.
Mixed Mode Transaction Status Flags	
Stalled	**Host Mode** Set by the SIE to indicate that a SETUP, IN or OUT transaction has been stalled. This is set **instead of** SetupSent, InRcvd or OutSent. The flag must be cleared by software before enabling the transaction which may cause the flag to be set. **Device or Peripheral Mode** Flag set by the SIE to indicate that an IN transaction has been illegally stalled. The flag must be cleared in software after processing its occurrence. See REGStatus Flags description above.

IllegalNak	**Host Mode** Set by hardware instead of SetupSent, if a SETUP transaction is NAKed. The flag must be cleared by software before enabling the transaction which may cause the flag to be set. **Device or Peripheral Mode** Set by hardware instead of InSent, if an IN transaction is (illegally) NAKed. See REGStatus Flags description above.
TransactionTimedOut	**Host Mode** Set by hardware if a transaction is timed out. If NoRetryOnTimeout is set, only one of the flags SetupSent, InRcvd, OutSent, Stalled, IllegalNak, TransactionTimedOut or DataToggleError will be set by the SIE. If NoRetryOnTimeout is not set, then SetupSent, InRcvd, OutSent Stalled or IllegalNak as appropriate must be true before testing TransactionTimedOut. **Device or Peripheral Mode** Applies to IN transactions only. Set by hardware if an IN transaction is timed out. If NoRetryOnTimeout is set, only one of the flags InSent, TransactionTimedOut Stalled or IllegalNak will be set by the SIE. See REGStatus Flags description above.
Bus State Flags	
DeviceConnected	This flag is set by the SIE when a device connects using its data line pull-up resistor, and is cleared when the SIE detects that the device disconnects, or VBUS falls below session valid. This flag should not be used to determine when the device releases D+, when the state of VBUS is in question. Use DPlusHigh for that purpose.
ResetStarted	This flag is set by the SIE (when ControlMode = DEVICE) when a bus reset start is detected. It must be cleared using the ClearFlag instruction.
ResetEnded	This flag is set by the SIE (when ControlMode = DEVICE) when a bus reset end is detected. It must be cleared using the ClearFlag instruction.
HighSpeedSet	This flag is set or cleared by the SIE (ControlMode = HOST) during a reset, to indicate whether a high speed connection was established.
LowSpeedSet	This flag is set or cleared by the SIE (ControlMode = HOST) during a reset, to indicate whether a low speed connection was established.
Suspended	**Host Mode** Set by SIE if it enters suspend state (by SuspendStart command, or by a successful LPM transaction). **Device or Peripheral Mode** This flag is set by the SIE if it gets suspended by the UUT. The flag must be cleared by software.
Resumed	**Host Mode** Set by the SIE if the UUT resumes the tester. The flag must be cleared by software. **Device or Peripheral Mode** This flag is set by the SIE if the tester gets resumed by the UUT. The flag must be cleared by software.

DPlusHigh	**Host Mode** Shows the state of D+ in host mode. **Host, Device or Peripheral Mode** Shows the state of D+ if high speed terminated state if EnableLineStateDetect has been set. (Used for USB Test Mode Support.)
DMinusHigh	**Host Mode** Shows the state of D- in host mode. **Host, Device or Peripheral Mode** Shows the state of D- if high speed terminated state if EnableLineStateDetect has been set. (Used for USB Test Mode Support.)
SE0	**Host Mode** True when D+ and D- are low.
SessionValid	VBUS is above a voltage between 0.8V and 4V.
ChirpSequenceError	**Host Mode** Set by hardware if something about a chirp sequence is out of spec. Cleared by script. (*not currently implemented*)
Test Mode Status Flags	
Inactivity	**Peripheral or Device Mode** This flag is set by the SIE after 3ms of inactivity.
Data0Rcvd	This flag is set whenever a packet with a PID of DATA0 is received. Used to detect high speed test packets (these have a PID of DATA0).
InTestMode	**Peripheral or Device Mode** Indicates that the SIE has detected that the UUT host is sending high speed conditions J, K or SE0. The logic is that if EnableTestMode is set, and we are in a High Speed state, and there is no activity, then the InTestMode flag gets set by the SIE and the PHY mode is changed to a state (also used for chirps) which disables bit stuffing and NRZI to allow the detection of line state.
SOFTestComplete	See REGSofsRcvd register description for details.

4.3.6.4 Messaging Flags

Flag Name	Function
ResponseYes ResponseOk ResponseRetry	Set by communication from application in response to dialog creating SendMessage, SendPrompt or SendReport. Cleared by software. These three flag names are alternative names for the same flag.
ResponseNo ResponseIgnore	Set by communication from application in response to dialog creating SendMessage, SendPrompt or SendReport. Cleared by software. These two flag names are alternative names for the same flag.
ResponseCancel ResponseAbort	Set by communication from application in response to dialog creating SendMessage, SendPrompt or SendReport. Cleared by software. These two flag names are alternative names for the same flag.

4.3.7 Alert Parameters

Parameter	Function
ALERT_NONE	The message defined will be sent to the text report in the GraphicUSB application, and also to the Capture report if any. The parameter keyword ALERT_NONE may be omitted.
ALERT_OK	In addition to the report message, a modal dialog will be displayed containing the message, plus an OK button. Until the OK button is pressed, the script will be paused. Pressing OK results in the ResponseOk flag being set, for testing by the script.
ALERT_OKCANCEL	In addition to the report message, a modal dialog will be displayed containing the message, plus an OK button and a CANCEL button. Until one of the buttons is pressed, the script will be paused. Pressing Ok or Cancel results in the ResponseOk or ResponseCancel flag being set, for testing by the script.
ALERT_YESNO	In addition to the report message, a modal dialog will be displayed containing the message, plus a YES button and a NO button. Until one of the buttons is pressed, the script will be paused. Pressing Yes or No results in the ResponseYes or ResponseNo flag being set, for testing by the script.
ALERT_ABORTRETRYIGNORE	In addition to the report message, a modal dialog will be displayed containing the message, plus an ABORT button, a RETRY button and an IGNORE button. Until one of the buttons is pressed, the script will be paused. Pressing Abort, Retry or Ignore results in the ResponseAbort, ResponseRetry or ResponseIgnore flag being set, for testing by the script.
ALERT_RETRYCANCEL	In addition to the report message, a modal dialog will be displayed containing the message, plus a RETRY button and a CANCEL button. Until one of the buttons is pressed, the script will be paused. Pressing Retry or Cancel results in the ResponseRetry or ResponseCancel flag being set, for testing by the script.
ALERT_YESNOCANCEL	In addition to the report message, a modal dialog will be displayed containing the message, plus a YES button, a NO button and an CANCEL button. Until one of the buttons is pressed, the script will be paused. Pressing Yes, No or Cancel results in the ResponseYes, ResponseNo or ResponseCancel flag being set, for testing by the script.

4.4 Communication with Electrical Test Board

The Electrical Test Board (ETB) is accessed using two script commands:

```
WriteToETB <addr> <value>
ReadFromETB <addr>
```

Addresses for the ETB elements are usually represented by keywords. All the available keywords are defined in the tables on the following pages. For example, to connect the VBUS generator to the front panel micro-AB test socket, use:

```
WriteToETB SW_VBUS_AB 1
```

Or because the keyword 'ON' has the value 1:

```
WriteToETB SW_VBUS_AB ON
```

To set the VBUS generator to 5V, use:

```
WriteToETB DAC_VBUS 2500
```

The value 2500 represents 5V because this DAC has a unit size of 2mV. For any given DAC (or ADC) the unit size is defined in one of the following tables.

An example of reading the ETB follows. To read the ADC in one of the VBUS voltage watch-blocks, to determine what the VBUS voltage actually is, use:

```
ReadFromETB ADC_VBUS_AB
```

After executing this command, the voltage in question is placed in the (16 bit wide) accumulator. In this case the units are defined (in a table below) as 2mV, so 5V is represented by 2500.

All the keywords for the ETB elements begin with a prefix illustrating the read/write direction, and the range of values available.

Prefix	Purpose	Value Range
SW_	A write-only switch	0=off, 1=on
SNS_	A read-only sense point	0=off, 1=on
DAC_	A write-only DAC setting	Typically 0-4095, though the valid range may be further limited
ADC_	A read-only ADC result	Typically 0-4095, though the valid range may be further limited

4.4.1 Complete List of Control Reads and Writes on ETB

4.4.1.1 ETB Switches (Write Only)

Switch Name	Sub-circuit	Value	Purpose
SW_RESET_ALL	Control	1	Sets all switches to 0
SW_RESET_MOST	Control	1	Sets all switches to 0, except SW_ID_GND SW_VBUS_CAP_1 SW_VBUS_CAP_2 SW_VBUS_CAP_3 SW_VBUS_CAP_4 SW_VBUS_CAP_5 SW_VBUS_PD
SW_VBUS_PD	VBUS pull-down circuit 10K	0 (off), 1 (on)	Helps to detect externally connected VBUS
SW_VBUS_LKG	VBUS pull-down circuit 2K	0 (off), 1 (on)	Used to measure VBUS leakage
SW_VBUS_CAP_1	VBUS capacitive loading	0 (off), 1 (on)	150nF on VBUS
SW_VBUS_CAP_2	VBUS capacitive loading	0 (off), 1 (on)	900nF on VBUS
SW_VBUS_CAP_3	VBUS capacitive loading	0 (off), 1 (on)	1uF on VBUS
SW_VBUS_CAP_4	VBUS capacitive loading	0 (off), 1 (on)	5.5uF on VBUS
SW_VBUS_CAP_5	VBUS capacitive loading	0 (off), 1 (on)	9 uF on VBUS
SW_VBUS_CAP_6	VBUS capacitive loading	0 (off), 1 (on)	100 uF on VBUS
SW_ADP_RES_1	ADP source	0 (off), 1 (on)	10K pull-down for ADP
SW_ADP_RES_2	ADP source	0 (off), 1 (on)	2K2 pull-up for ADP
SW_VBUS_AB	VBUS sourcing	0 (off), 1 (on)	Connects VBUS to UUT socket
The following eight resistors are available to connect between the µAB UUT connector ID pin, and ground. Only one of these resistors should be switched on at a time. It should be noted that in the absence of any other resistor, the 220k resistor SW_ID_RES_8 will be automatically switched on to avoid misoperation of the watch-block measurement circuitry ; when another value is switched on, the SW_ID_RES_8 will be switched off.			
SW_ID_RES_1	ID pin resistors	0 (off), 1 (on)	RID_GND max (1k)
SW_ID_RES_2	ID pin resistors	0 (off), 1 (on)	RID_C min (36k)
SW_ID_RES_3	ID pin resistors	0 (off), 1 (on)	RID_C max (37k)
SW_ID_RES_4	ID pin resistors	0 (off), 1 (on)	RID_B min (67k)
SW_ID_RES_5	ID pin resistors	0 (off), 1 (on)	RID_B max (69k)
SW_ID_RES_6	ID pin resistors	0 (off), 1 (on)	RID_A min (122k)
SW_ID_RES_7	ID pin resistors	0 (off), 1 (on)	RID_A max (126k)
SW_ID_RES_8	ID pin resistors	0 (off), 1 (on)	RID_FLOAT min (220k). This switch is automatically on when SW_ID_RES_1 to 7 are all off.
SW_ID_GND	ID pin resistors	0 (off), 1 (on)	Connect ID_AB pin to ground
The following two resistor switches are provided to allow the measurement of ID pin resistance in the UUT.			

SW_ID_RES_M_1	ID measurement	0 (off), 1 (on)	1K pullup
SW_ID_RES_M_2	ID measurement	0 (off), 1 (on)	82K pullup
DCD and charger detect circuit elements			
SW_DMDP_TEST	Data line test circuit	0 (off), 1 (on)	Switches data lines to measurement circuit instead of transceiver.
SW_DCD_RES_AB	DCD and charger detect	0 (off), 1 (on)	200R DP to DM, to simulate DCP.
SW_DCD_RES_1	DCD and charger detect	0 (off), 1 (on)	1k5 to DM
SW_DCD_RES_2	DCD and charger detect	0 (off), 1 (on)	200R to DM
SW_DCD_RES_3	DCD and charger detect	0 (off), 1 (on)	3k9 to DM
SW_DCD_RES_4	DCD and charger detect	0 (off), 1 (on)	15K to DM
SW_DCD_RES_5	DCD and charger detect	0 (off), 1 (on)	100k to DM
SW_DCD_RES_6	DCD and charger detect	0 (off), 1 (on)	Spare (do not use)
SW_DCD_RES_7	DCD and charger detect	0 (off), 1 (on)	200R to DP
SW_DCD_RES_8	DCD and charger detect	0 (off), 1 (on)	1k5 to DP
SW_DCD_RES_9	DCD and charger detect	0 (off), 1 (on)	15k to DP
SW_DCD_RES_10	DCD and charger detect	0 (off), 1 (on)	100k to DP
SW_DCD_CAP	DCD and charger detect	0 (off), 1 (on)	1nF to DP, used for DCP data line capacitance measurement.
Miscellaneous controls for ACA testing			
SW_VBUS_ACC_LD1	VBUS_ACC load	0 (off), 1 (on)	Connect 625 Ω to ground
SW_VBUS_ACC_LD2	VBUS_ACC load	0 (off), 1 (on)	Connect 10 Ω to ground
SW_VBUS_ACC	VBUS sourcing	0 (off), 1 (on)	Connects VBUS to VBUS_ACC
SW_ID_ACC_LOW	Accessory port ID	0 (off), 1 (on)	Connects 0Ω from ID_ACC to ground
SW_VBUS_CHG	VBUS sourcing	0 (off), 1 (on)	Connects VBUS to VBUS_CHG
SW_DCD_RES_CHG	ACA charger detect	0 (off), 1 (on)	200 Ω DP to DM on charger port
Inserts a common mode voltage offset into the USB ground, to simulate the effect of large charging port current. Used in conjunction with DAC_CMO.			
SW_CMO_SRC	Common Mode Offset	0 (off), 1 (on)	Switch common mode circuit to work when tester sources VBUS
SW_CMO_LD	Common Mode Offset	0 (off), 1 (on)	Switch common mode circuit to work when UUT sources VBUS
Eight signals to clear the watch-block RISE and FALL condition latches			
SW_WB_VBUS_AB_1_CLR	Watch-block for AB connector VBUS	1	Clear Watch-block RISE and FALL latches
SW_WB_VBUS_AB_2_CLR	Watch-block for AB connector VBUS	1	Clear Watch-block RISE and FALL latches
SW_WB_VBUS_IIL_CLR	Watch-block for AB connector VBUS Current Load current	1	Clear Watch-block RISE and FALL latches
SW_WB_VBUS_IVL_CLR	Watch-block for AB connector VBUS Voltage Load current	1	Clear Watch-block RISE and FALL latches
SW_WB_IVBUS_AB_CLR	Watch-block for AB connector VBUS current	1	Clear Watch-block RISE and FALL latches

SW_WB_DP_AB_CLR	Watch-block for AB connector DP	1	Clear Watch-block RISE and FALL latches
SW_WB_DM_ AB_CLR	Watch-block for AB connector DM	1	Clear Watch-block RISE and FALL latches
SW_WB_ID_ AB_CLR	Watch-block for AB connector ID	1	Clear Watch-block RISE and FALL latches
SW_VBUS_IL_ HI	VBUS Current Load	0 (off), 1 (on)	High current range
SW_VBUS_VL_EN	VBUS Voltage Load	0 (off), 1 (on)	Enable Voltage Load
SW_VBUS_I_HI	Vbus current measurement.	0 (off), 1 (on)	**High Range** 0-2000 1mA units Switching SW_VBUS_I_HI on, switches off SW_VBUS_I_MID and SW_VBUS_I_LO automatically.
SW_VBUS_I_MID	VBUS current measurement.	0 (off), 1 (on)	**Mid Range** 0-2000 0.1mA units Switching SW_VBUS_I_MID on, switches off SW_VBUS_I_HI and SW_VBUS_I_LO automatically.
SW_VBUS_I_LO	VBUS current measurement.	0 (off), 1 (on)	**Low Range** 0-2000 0.01mA units Switching SW_VBUS_I_LO on, switches off SW_VBUS_I_MID and SW_VBUS_I_HI automatically.
Signals to select the sources of the positive and negative inputs to the VBUS voltage watch-block			
Switching on one of the next four switches, has the effect of switching the other three off, though it is good practice to switch any other switch in use off first.			
SW_VBUS_VP_AB	VBUS voltage measurement	0 (off), 1 (on)	Connect Watch-block positive input to Vbus on AB connector
SW_VBUS_VP_ACC	VBUS voltage measurement	0 (off), 1 (on)	Connect Watch-block positive input to VBUS on accessory connector
SW_VBUS_VP_CHG	VBUS voltage measurement	0 (off), 1 (on)	Connect Watch-block positive input to VBUS on charger connector
SW_DP_VP_CHG	Voltage measurement	0 (off), 1 (on)	Connect VBUS Watch-block +ve to DP_CHG
Switching on one of the next four switches, has the effect of switching the other three off, though it is good practice to switch any other switch in use off first.			
SW_VBUS_VM_AB	VBUS voltage measurement	0 (off), 1 (on)	Connect Watch-block negative input to VBUS on AB connector
SW_VBUS_VM_ACC	VBUS voltage measurement	0 (off), 1 (on)	Connect Watch-block negative input to VBUS on accessory connector
SW_GND_VM_AB	VBUS voltage measurement	0 (off), 1 (on)	Connect Watch-block negative input to GND on AB connector
SW_GND_VM_GEN	VBUS voltage measurement	0 (off), 1 (on)	Connect Watch-block to GND_GEN
Controls external measurement connections on back panel			
SW_TRIGGER_OUT	Ext measurement	0 (off), 1 (on)	Sets level of back panel trigger output
SW_MEAS_VBUS_AB	Ext measurement	0 (off), 1 (on)	Select one item for back panel

			measurement terminal, deselects all others
SW_MEAS_VBUS_ACC	Ext measurement	0 (off), 1 (on)	Select one item for back panel measurement terminal, deselects all others
SW_MEAS_VBUS_CHG	Ext measurement	0 (off), 1 (on)	Select one item for back panel measurement terminal, deselects all others
SW_MEAS_DP	Ext measurement	0 (off), 1 (on)	Select one item for back panel measurement terminal, deselects all others
SW_MEAS_DM	Ext measurement	0 (off), 1 (on)	Select one item for back panel measurement terminal, deselects all others

4.4.1.2 Supervisory (Write Only)

Switch Name	Value	Purpose
SW_INHIBIT_UPDATE	0 (off), 1 (on)	The Disable Switch Update flag. When set it prevents any writes to switches from being carried out, on clearing, all the switches get updated in one operation.
SW_INHIBIT_READ	0 (off), 1 (on)	The Disable ADC Read Function flag. When set, no reads take place. Default is cleared.

4.4.1.3 ETB Sense Points

Sense Point Name	Sub-circuit	Value	Purpose
Each watch-block contains a latch which remembers whether the voltage or current being watched rose higher than the selected watch value, since the latch was cleared. These sense points are read to discover the result.			
SNS_WB_VBUS_AB_1_RISE	Watch-block 1 for AB connector VBUS	0=FALSE 1=TRUE	Watch-block rise latch
SNS_WB_VBUS_AB_2_RISE	Watch-block 2 for AB connector VBUS	0=FALSE 1=TRUE	Watch-block rise latch
SNS_WB_VBUS_IIL_RISE	Watch-block for AB connector VBUS Current Load current	0=FALSE 1=TRUE	Watch-block rise latch
SNS_WB_VBUS_IVL_RISE	Watch-block for AB connector VBUS Voltage Load current	0=FALSE 1=TRUE	Watch-block rise latch
SNS_WB_IVBUS_AB_RISE	Watch-block for AB connector VBUS current	0=FALSE 1=TRUE	Watch-block rise latch
SNS_WB_DP_AB_RISE	Watch-block for AB connector DP	0=FALSE 1=TRUE	Watch-block rise latch
SNS_WB_DM_ AB_RISE	Watch-block for AB connector DM	0=FALSE 1=TRUE	Watch-block rise latch

SNS_WB_ID_ AB_RISE	Watch-block for AB connector ID	0=FALSE 1=TRUE	Watch-block rise latch
Each watch-block contains a latch which remembers whether the voltage or current being watched fell lower than the selected watch value, since the latch was cleared. These sense points are read to discover the result.			
SNS_WB_VBUS_AB_1_FALL	Watch-block 1 for AB connector VBUS	0=FALSE 1=TRUE	Watch-block fall latch
SNS_WB_VBUS_AB_2_FALL	Watch-block 2 for AB connector VBUS	0=FALSE 1=TRUE	Watch-block fall latch
SNS_WB_VBUS_IIL_FALL	Watch-block for AB connector VBUS Current Load current	0=FALSE 1=TRUE	Watch-block fall latch
SNS_WB_VBUS_IVL_FALL	Watch-block for AB connector VBUS Voltage Load current	0=FALSE 1=TRUE	Watch-block fall latch
SNS_WB_IVBUS_AB_FALL	Watch-block for AB connector VBUS current	0=FALSE 1=TRUE	Watch-block fall latch
SNS_WB_DP_AB_FALL	Watch-block for AB connector DP	0=FALSE 1=TRUE	Watch-block fall latch
SNS_WB_DM_ AB_FALL	Watch-block for AB connector DM	0=FALSE 1=TRUE	Watch-block fall latch
SNS_WB_ID_ AB_FALL	Watch-block for AB connector ID	0=FALSE 1=TRUE	Watch-block fall latch
Each watch-block can be directly read to discover whether the voltage or current being watched is currently higher than the selected watch value. These sense points are read to discover the result.			
SNS_WB_VBUS_AB_1_LEVEL	Watch-block 1 for AB connector VBUS	0=FALSE 1=TRUE	Watch-block level
SNS_WB_VBUS_AB_2_LEVEL	Watch-block 2 for AB connector VBUS	0=FALSE 1=TRUE	Watch-block level
SNS_WB_VBUS_IIL_LEVEL	Watch-block for AB connector VBUS Current Load current	0=FALSE 1=TRUE	Watch-block level
SNS_WB_VBUS_IVL_LEVEL	Watch-block for AB connector VBUS Voltage Load current	0=FALSE 1=TRUE	Watch-block level
SNS_WB_IVBUS_AB_LEVEL	Watch-block for AB connector VBUS current	0=FALSE 1=TRUE	Watch-block level
SNS_WB_DP_AB_LEVEL	Watch-block for AB connector DP	0=FALSE 1=TRUE	Watch-block level
SNS_WB_DM_ AB_LEVEL	Watch-block for AB connector DM	0=FALSE 1=TRUE	Watch-block level
SNS_WB_ID_ AB_LEVEL	Watch-block for AB connector ID	0=FALSE 1=TRUE	Watch-block level
Used for ACA testing			
SNS_VBUS_I_OUT	VBUS current direction detector	0=FALSE 1=TRUE	High if current is sourced by tester

4.4.1.4 ETB DAC Writes

DAC Name	Sub-circuit	Value	Purpose
DAC_RESET_ALL	Control	0	Sets all DACs to 0
Each watch-block can be programmed with a voltage or current, as appropriate, which determines the value reported by the RISE and FALL watch-block outputs.			
DAC_WB_VBUS_AB_1	Watch-block 1 for AB connector VBUS	0-4095 2mV units	Watch-block DAC input
DAC_WB_VBUS_AB_2	Watch-block 2 for AB connector VBUS	0-4095 2mV units	Watch-block DAC input
DAC_WB_VBUS_IIL	Watch-block for AB connector VBUS Current Load current	0-3000 2mA units	Watch-block DAC input
DAC_WB_VBUS_IVL	Watch-block for AB connector VBUS Voltage Load current	0-2000 1mA units	Watch-block DAC input
DAC_WB_IVBUS_AB	Watch-block for AB connector VBUS current	**High Range** 0-2000 1mA units **Mid Range** 0-2000 0.1mA units **Low Range** 0-2000 0.01mA units	Watch-block DAC input Use SW_VBUS_I_HI, SW_VBUS_I_MID or SW_VBUS_I_LO to select range.
DAC_WB_DP_AB	Watch-block for AB connector DP	0-4095 1mV units	Watch-block DAC input
DAC_WB_DM_ AB	Watch-block for AB connector DM	0-4095 1mV units	Watch-block DAC input
DAC_WB_ID_ AB	Watch-block for AB connector ID	0-4095 1mV units	Watch-block DAC input
DAC_VBUS	VBUS generator	0-3250 2mV units	VBUS generator DAC input
DAC_VBUS_IL	VBUS current load	**High Range** 0-2500 2mA **Low Range** 0-2500 0.2mA	VBUS current load, SW_VBUS_IL_HI selects range, 0=Low, 1=High.
DAC_VBUS_VL	VBUS voltage load	0-2500 2mV	VBUS voltage load
DAC_DM	DM voltage source	0-3600 1mV	DM voltage source
DAC_DP	DP voltage source	0-3600 1mV	DP voltage source

DAC_CMO	Common mode offset	0-500 1mV	Common Mode Offset

4.4.1.5 ETB ADC Reads

ADC Name	Sub-circuit	Value	Purpose
ADC_WB_VBUS_AB_1 ADC_WB_VBUS_AB_2	Watch-block for AB connector VBUS	0-4095 2mV units	Watch-block ADC output
ADC_WB_VBUS_IIL	Watch-block for AB connector VBUS Current Load current	0-4095 2mA units	Watch-block ADC output. Limit current to 6A when performing loads, and restrict loads above 500mA to 5 seconds. Thermal shut-down protects PET.
ADC_WB_VBUS_IVL	Watch-block for AB connector VBUS Voltage Load current	0-4095 1mA units	Watch-block ADC output. Limit current to 2A when performing loads, and restrict loads above 500mA to 5 seconds. Thermal shut-down protects PET.
ADC_WB_IVBUS_AB	Watch-block for AB connector VBUS current	**High Range** 0-2000 1mA units **Mid Range** 0-2000 0.1mA units **Low Range** 0-2000 0.01mA units	Watch-block ADC output. Use SW_VBUS_I_HI, SW_VBUS_I_MID or SW_VBUS_I_LO to select range.
ADC_WB_DP_AB	Watch-block for AB connector DP	0-4095 1mV units	Watch-block ADC output
ADC_WB_DM_ AB	Watch-block for AB connector DM	0-4095 1mV units	Watch-block ADC output
ADC_WB_ID_ AB	Watch-block for AB connector ID	0-4095 1mV units	Watch-block ADC output

4.4.2 Sending Information from the Script to the Application

The following script commands are available for this purpose:

Command	Function
SendMessage "message" <message parameter>	Sends message to application which will appear both in text report, and in capture. Unless the parameter ALERT_NONE or no parameter is used, the message will also appear in a pop-up dialog. While this modal dialog is visible, no further messages will be processed by the application. Therefore the script must wait until a response flag is set by clicking one of the dialog buttons.
SendReport "message" < message parameter>	Sends message to application which will appear in the text report, but not in the capture.
SendPrompt "message" < message parameter>	This command, must be used with a <message parameter> which invokes a dialog. This dialog is non-modal, in the sense that further script processing may take place, and any further SendMessage will dismiss the dialog. If the dialog needs not to be dismissed by a further message then that furter message should be sent using SendInfo.
SendInfo "message"	This command is the only way to send a message to appear in the text report, and the capture, without dismissing the non-modal dialog created by SendPrompt. It is illegal to use a parameter with SendInfo, as a dialog is already assumed to be present.
SendValue <value parameter>	This is a method used to send a value to the application for processing. No visible message is associated with the command. The <value parameter> specifies what the application is required to do with the value in the accumulator at the time.

SendMessage is the means of conveying information from the running script back to the GraphicUSB application. It can be used:

- to display an informative message in the report at the actual time an event is taking place; this would also appear embedded at the appropriate position in the capture file.
- to display such a message but embed a formatted value in the message, which can originate from the A register value, or from a specially calculated voltage or time value from options below.
- to display a message which requires a response from the user; this would also appear embedded at the appropriate position in the capture file.

- to send a procedural message involving the use of the host application for data processing, which should not appear in the display or in the capture. An example would be defining a sequence of voltage measurements, which need to be averaged.

  In the script the message is formatted as:

  SendMessage “text” [PARAM(S)]

  Or

  SendValue [PARAM(S)]

4.4.2.1 Send Message Formatting Values

Template	Function	Units
Templates to display Accumulator value at the time of sending message, with assumed units		
$decnmu	accumulator value as decimal unsigned number	-
$hexnm2	accumulator value as 2 character wide hexadecimal with leading zeros	-
$hexnm4	accumulator value as 4 character wide hexadecimal with leading zeros	-
$vac1mV	accumulator value as voltage	1mV
$vac2mV	accumulator value as voltage	2mV
$iac_01mA	accumulator value as current	0.01mA
$iac_05mA	accumulator value as current	0.05mA
$iac_1mA	accumulator value as current	0.1mA
$iac1mA	accumulator value as current	1mA
$iac2mA	accumulator value as current	2mA
$tm10us	accumulator value as time	10us
Templates to display the average value accumulated using PREPARETOAVERAGE, VALUETOAVERAGE and PERFORMAVERAGE, using assumed units.		
$ave2mV	calculated average as voltage	2mV
$ave_01mA	calculated average as current	0.01mA
$ave_05mA	calculated average as current	0.05mA
$ave_1mA	calculated average as current	0.1mA
$ave1mA	calculated average as current	1mA
Templates to display the duration values measured using TIMESTART0, TIMEEND0 etc		
$durat0	duration0	Automatically formatted as required
$durat1	duration1	Automatically formatted as required
$durat2	duration2	Automatically formatted as required
$durat3	duration3	Automatically formatted as required
$durat4	duration4	Automatically formatted as required
Templates to display the ADP duration values		
$duradp0	Duration of most recent ADP probe (n)	Automatically formatted as required
$duradp1	Duration of ADP probe (n-1)	Automatically formatted as required
$duradp2	Duration of ADP probe (n-2)	Automatically formatted as

		required
Templates to display 'value1'		
$v1decu	value1 as unsigned decimal	-
$v1hex2	value1 as 2 character wide hexadecimal with leading zeros	-
$v1hex4	value1 as 4 character wide hexadecimal with leading zeros	-
$v1_1mV	value1 as voltage	1mV
$v1_2mV	value1 as voltage	2mV
$v2decu	value2 as unsigned decimal	-
$v2hex2	value2 as 2 character wide hexadecimal with leading zeros	-
$v2hex4	value2 as 4 character wide hexadecimal with leading zeros	-
$v3decu	value3 as unsigned decimal	-
$v3hex2	value3 as 2 character wide hexadecimal with leading zeros	-
$v3hex4	value3 as 4 character wide hexadecimal with leading zeros	-
$v4decu	value4 as unsigned decimal	
$v4hex2	value4 as 2 character wide hexadecimal with leading zeros	
$v4hex4	value4 as 4 character wide hexadecimal with leading zeros	
$v5decu	value5 as unsigned decimal	
$v5hex2	value5 as 2 character wide hexadecimal with leading zeros	
$v5hex4	value5 as 4 character wide hexadecimal with leading zeros	
$v6decu	value6 as unsigned decimal	
$v6hex2	value6 as 2 character wide hexadecimal with leading zeros	
$v6hex4	value6 as 4 character wide hexadecimal with leading zeros	
$v7decu	value7 as unsigned decimal	
$v7hex2	value7 as 2 character wide hexadecimal with leading zeros	
$v7hex4	value7 as 4 character wide hexadecimal with leading zeros	
$v8decu	value8 as unsigned decimal	
$v8hex2	value8 as 2 character wide hexadecimal with leading zeros	
$v8hex4	value8 as 4 character wide hexadecimal with leading zeros	
Templates to display Special (ADP) Timer Values		
$tmspcr	calculated special timer current value	Automatically formatted as

		required
$tmspmn	calculated special timer min value	Automatically formatted as required
$tmspmx	calculated special timer max value	Automatically formatted as required
$tmspjt	calculated special timer jitter percentage	-
$tm1us	accumulator value as time	1us
$tm10us	accumulator value as time	10us
$tm100us	accumulator value as time	100us
$tm1ms	accumulator value as time	1ms
Templates to display pass/fail counts from ASSERT mechanism		
$assert	Display (as appropriate): • YES (PASS) • NO (FAIL) • NOT TESTED • NOT APPLICABLE • YES • NO In relation to the assertion indexed by the accumulator value. The last two results YES or NO appear in place of YES (PASS) or NO (FAIL), if the SendValue parameter STYLEPASS has been previously sent	
$pass_cnt	Number of passing results displayed in SendMessage strings during the current script suite.	
$fail_cnt	Number of failing results displayed in SendMessage strings during the current script suite.	
$not_test_cnt	Number of 'not tested' results displayed in SendMessage strings during the current script suite.	
$not_appl	Number of 'not applicable' results displayed in SendMessage strings during the current script suite.	
Templates to display values useful when decoding descriptors used during enumeration		
$viddec	Vendor ID from accumulator value expressed in decimal	
$vidhex	Vendor ID from accumulator value expressed in hexadecimal	
$vidstr	Vendor ID from accumulator value expressed as a string, based on USB-IF file **usb.if**. Note that this file currently contains only current members instead of vendors who have legally purchased a Vendor ID from the USB-IF.	
$strlang	Language description string from accumulator value, based on list of languages defined by the USB-IF in relation to string descriptors.	

$class	Class description string from accumulator value, based on list of valid classes defined by the USB-IF.	
$subclass	Subclass description string from accumulator value, based on list of valid classes defined by the USB-IF.	
$protocol	Protocol description string from accumulator value, based on list of valid classes defined by the USB-IF.	
$str1	String built up using SendValue STR1CLEAR and SendValue STR1CHAR.	
$str2	String built up using SendValue STR2CLEAR and SendValue STR2CHAR.	

4.4.2.2 SendValue Parameters

Parameter Name	Function
TESTFAILEDFATAL	Tells the application to report a fatal failure at the end of the script. In standard test scripts this will prevent further scripts being run.
TESTFAILEDNONFATAL	Tells the application to report a non-fatal failure at the end of the script. In standard test scripts this will still allow further scripts to be run.
PREPARETOAVERAGE	Primes the application to average a series of values (clears any previous averaging result in `calculatedAverage`).
VALUETOAVERAGE	The accompanying accumulator value (a 16-bit value) is added to those being averaged.
PERFORMAVERAGE	The values to be averaged are averaged at this point, and the result is placed in a store called `calculatedAverage`.
MINAVERAGE	The accumulator value accompanying this parameter is compared with `calculatedAverage`. If `calculatedAverage` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXAVERAGE	The accumulator value accompanying this parameter is compared with `calculatedAverage`. If `calculatedAverage` is less than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
AVERAGESUBTRACT	The accumulator value accompanying this parameter is subtracted from the `calculatedAverage`. The result is constrained to not be less than zero.
TIMESTART0	The accompanying timestamp (automatically generated) defines the start time for the store called `duration0`.
TIMEEND0	The accompanying timestamp (automatically generated) defines the end time for the store called `duration0`, and causes the duration to be calculated.
MINDURATION0_10US	The accumulator value accompanying this parameter, assuming units of 10µs, is compared with `duration0`. If `duration0` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXDURATION0_10US	The accumulator value accompanying this parameter, assuming units of 10µs, is compared with `duration0`. If `duration0` is less than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MINDURATION0_1MS	The accumulator value accompanying this parameter, assuming units of 1ms, is compared with `duration0`. If `duration0` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXDURATION0_1MS	The accumulator value accompanying this parameter, assuming units of 1ms, is compared with `duration0`. If `duration0` is less than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
TIMESTART1	The accompanying timestamp (automatically generated) defines the start time for the store called `duration1`.
TIMEEND1	The accompanying timestamp (automatically generated) defines the end time for the store called `duration1`, and causes the duration to be calculated.
MINDURATION1_10US	The accumulator value accompanying this parameter, assuming units of 10µs, is

	compared with `duration1`. If `duration1` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXDURATION1_10US	The accumulator value accompanying this parameter, assuming units of 10µs, is compared with `duration1`. If `duration1` is less than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MINDURATION1_1MS	The accumulator value accompanying this parameter, assuming units of 1ms, is compared with `duration1`. If `duration1` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXDURATION1_1MS	The accumulator value accompanying this parameter, assuming units of 1ms, is compared with `duration1`. If `duration1` is less than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
TIMESTART2	The accompanying timestamp (automatically generated) defines the start time for the store called `duration2`.
TIMEEND2	The accompanying timestamp (automatically generated) defines the end time for the store called `duration2`, and causes the duration to be calculated.
MINDURATION2_1US	The accumulator value accompanying this parameter, assuming units of 1µs, is compared with `duration2`. If `duration2` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXDURATION2_1US	The accumulator value accompanying this parameter, assuming units of 1µs, is compared with `duration2`. If `duration2` is less than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MINDURATION2_10US	The accumulator value accompanying this parameter, assuming units of 10µs, is compared with `duration2`. If `duration2` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXDURATION2_10US	The accumulator value accompanying this parameter, assuming units of 10µs, is compared with `duration2`. If `duration2` is less than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MINDURATION2_1MS	The accumulator value accompanying this parameter, assuming units of 1ms, is compared with `duration2`. If `duration2` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXDURATION2_1MS	The accumulator value accompanying this parameter, assuming units of 1ms, is compared with `duration2`. If `duration2` is less than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
TIMESTART3	The accompanying timestamp (automatically generated) defines the start time for the store called `duration3`.
TIMEEND3	The accompanying timestamp (automatically generated) defines the end time for the store called `duration3`, and causes the duration to be calculated.
MINDURATION3_10US	The accumulator value accompanying this parameter, assuming units of 10µs, is compared with `duration3`. If `duration3` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXDURATION3_10US	The accumulator value accompanying this parameter, assuming units of 10µs, is compared with `duration3`. If `duration3` is less than or equal to the

	accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MINDURATION3_1MS	The accumulator value accompanying this parameter, assuming units of 1ms, is compared with `duration3`. If `duration3` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXDURATION3_1MS	The accumulator value accompanying this parameter, assuming units of 1ms, is compared with `duration3`. If `duration3` is less than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
TIMESTART4	The accompanying timestamp (automatically generated) defines the start time for the store called `duration4`.
TIMEEND4	The accompanying timestamp (automatically generated) defines the end time for the store called `duration4`, and causes the duration to be calculated.
MINDURATION4_10US	The accumulator value accompanying this parameter, assuming units of 10µs, is compared with `duration4`. If `duration4` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXDURATION4_10US	The accumulator value accompanying this parameter, assuming units of 10µs, is compared with `duration4`. If `duration4` is less than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MINDURATION4_1MS	The accumulator value accompanying this parameter, assuming units of 1ms, is compared with `duration4`. If `duration4` is greater than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
MAXDURATION4_1MS	The accumulator value accompanying this parameter, assuming units of 1ms, is compared with `duration4`. If `duration4` is less than or equal to the accumulator value, then the PET RESPONSE_YES flag is set, otherwise the PET RESPONSE_NO flag is set.
VALUE1	The accumulator value accompanying this parameter will be placed in the store called `value1`. This value can then be displayed in SendMessage type comands, by using the `$v1decu, $v1hex2, $v1hex4 $v1_1mV or $v1_1mV` templates.
VALUE2	The accumulator value accompanying this parameter will be placed in the store called `value2`. This value can then be displayed in SendMessage type comands, by using the `$v2decu, $v2hex2` or `$v2hex4` templates.
VALUE3	The accumulator value accompanying this parameter will be placed in the store called `value3`. This value can then be displayed in SendMessage type comands, by using the `$v3decu, $v3hex2` or `$v3hex4` templates.
VALUE4	The accumulator value accompanying this parameter will be placed in the store called `value4`. This value can then be displayed in SendMessage type comands, by using the `$v4decu, $v4hex2` or `$v4hex4` templates.
VALUE5	The accumulator value accompanying this parameter will be placed in the store called `value5`. This value can then be displayed in SendMessage type comands, by using the `$v5decu, $v5hex2` or `$v5hex4` templates.
VALUE6	The accumulator value accompanying this parameter will be placed in the store called `value6`. This value can then be displayed in SendMessage type comands, by using the `$v6decu, $v6hex2` or `$v6hex4` templates.
VALUE7	The accumulator value accompanying this parameter will be placed in the store called `value7`. This value can then be displayed in SendMessage type comands,

	by using the `$v7decu, $v7hex2` or `$v7hex4` templates.
VALUE8	The accumulator value accompanying this parameter will be placed in the store called `value8`. This value can then be displayed in SendMessage type comands, by using the `$v8decu, $v8hex2` or `$v8hex4` templates.
TIME_SPEC_CLEAR	Prepare the application to perform a special time period and jitter function. This function exaines a series of consectutive periods, and allows the minimum and maximum values of this period to be validated, along with the jitter of the periods. The first time period starts when TIME_SPEC_START is sent. Each time that TIME_SPEC_END is sent a period ends and a further period begins. The parameters TIME_SPEC_JITTER_MAX, TIME_SPEC_MIN and TIME_SPEC_MAX are used to test the periods. The set of functions is useful to testing the parameters of ADP probes. The SendMessage templates `$tmspcr` (current period value), `$tmspmn` (minimum period value), `$tmspmx` (maximum period value), and `$tmspjt` (jitter percentage), can be used to display the appropriate values.
TIME_SPEC_START	See TIME_SPEC_CLEAR function.
TIME_SPEC_END	See TIME_SPEC_CLEAR function.
TIME_SPEC_JITTER_MAX	See TIME_SPEC_CLEAR function.
TIME_SPEC_MIN	See TIME_SPEC_CLEAR function.
TIME_SPEC_MAX	See TIME_SPEC_CLEAR function.
DUR1_MINUS_DUR0_TO_DUR2	Sets `duration2 = duration1 - duration0`.
ADPCHANGEVAL	The accumulator value accompanying this parameter is used to set the assumed change in ADP probe period caused by a change in capacitance of 500nF. The default period is 6120 clocks. Modify this to match the ADP generation technique used.
ADPINIT	Initialises the three ADP period samples
TIMESTARTADP	Indicates that an ADP probe starts.
TIMEENDADP	Indicates that an ADP probe ends. Transfers the sample time into 'n', shifting previous 'n' to 'n-1' and previous 'n-1' to 'n-2'. If 'n' differs by more than ADPCHANGEVAL from 'n-2', the ResponseYes flag is set, otherwise the ResponseNo flag is set. Note that the first TIMEENDADP after ADPINIT will set all three samples to the same value. Also if a capacitance change is detected, then the most recent sample will be copied to the other two samples.
ADPEQUALISE	Sets samples 'n-2' and 'n-1' equal to sample 'n'.
ASSERTPASS	Indicates that the assertion indexed by the accumulator has passed a test. This result may still be mofified if on a later occasion the same assertion fails a test.
ASSERTFAIL	Indicates that the assertion indexed by the accumulator has failed a test. Once failed, no further modification is possible.
ASSERTYES	Indicates that the assertion indexed by the accumulator was a question to which the answer is ‘yes’.
ASSERTNO	Indicates that the assertion indexed by the accumulator was a question to which the answer is ‘no’.
ASSERTNOTAPPLICABLE	Indicates that the assertion indexed by the accumulator is not applicable in the current circumstances.
STYLEPASS	If the accumulator value is 1, then the SendMessage template $assert will be replaced with‘YES’ instead of ‘YES (PASS)’, or ‘NO’ instead of NO (FAIL)’.
TIMESTAMPOFF	Prevents the timestamp from being displayed on each message displayed in the report.

TIMESTAMPON	Enables the timestamp to be displayed on each message displayed in the report.
TABSIZE	Sets the tab size used in the report to be the number of columns indicated by the accumulator value.
ROWSIZE	Sets the maximum characters per row used in the report to be the number indicated by the accumulator value.
CHECKVID	If the Vendor ID value in the accumulator matches one found in the USB-IF official file, this sets the ResponseYes flag, otherwise it sets the ResponseNo flag. Until the application has had time to process this, neither flag will be set.
STR1CLEAR	string1 is a string which is stored in the application, but can be defined by the running script, one character at a time.The string can be displayed by using the SendMessage template $str1. This parameter clears the string.
STR1CHAR	string1 is a string which is stored in the application, but can be defined by the running script, one character at a time.The string can be displayed by using the SendMessage template $str1. This parameter concatenates one character to the string.
STR2CLEAR	String2 is a string which is stored in the application, but can be defined by the running script, one character at a time.The string can be displayed by using the SendMessage template $str2. This parameter clears the string.
STR2CHAR	String2 is a string which is stored in the application, but can be defined by the running script, one character at a time.The string can be displayed by using the SendMessage template $str2. This parameter concatenates one character to the string.
STRLANG	This sets the value nStrLang in the application from the accumulator value. The SendMessage template $strlang can then be used to display the name of the language.
CLASS	This sets the value nClass in the application from the accumulator value. The SendMessage template $class can then be used to display the name of the class.
SUBCLASS	This sets the value nSubclass in the application from the accumulator value. The SendMessage template $subclass can then be used to display the name of the protocol.
PROTOCOL	This sets the value nProtocol in the application from the accumulator value. The SendMessage template $protocol can then be used to display the name of the protocol.
DESCTYPE	This sets the value btDescType in the application from the accumulator value. It is used to specify the descriptor type containing class, subclass and prototype, so that its desrcription can be adjusted according when using the SendMessage templates $class, $subclass or $protocol.
BEEP	Causes a sound be be emitted.

4.5 PET Script Language Syntax Rules

4.5.1 Command Sequence

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

FileType MQPPET 1

ControlMode HOST | PERIPHERAL | DEVICE | DETACHED

…

4.5.2 Case Sensitivity

All commands and parameters are case insensitive. Thus

`SendMessage` is the same as `SENDMESSAGE`.

Scripts tend to use mixed case for commands and all upper case for pre-defined values, for clarity.

4.5.3 Command Lines

All commands must start on a new line. Command:

```
DnldToRam
```

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

Blank lines are allowed.

4.5.4 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 label is used as the target of a `Jump` command or a `DJNZCount`*`n`* command (without the ':'.

An example of a label is:

```
Label12:
```


4.5.5 Symbols

A symbol is defined as a sequence of alphanumeric characters (including '_') starting with an alphabetic character or '_'. It must be defined somewhere in the script, using an 'Equate', and should not match any keywords used by the compiler. A symbol is used in place of an immediate value.

An example of a symbol is:

```
_msAvailable
```

4.5.6 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.

```
DnLdToRam DevStatusAddr (
     0x01       ; self powered
     0x00)      ; remote wakeup not enabled
```

Comments are completely ignored by the compiler.

4.5.7 Tabs

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

4.5.8 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. After

```
Equate bMaxPower 50
```

the following:

```
50, 0x32 and bMaxPower
```

all represent the same value.

4.5.9 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.

4.6 Running a Script from the Command Line

It is possible to control GraphicUSB from another application using a command-line syntax. This allows another application to run a PET script and produce a report, and optionally a capture file. For the sake of simplicity, the example below shows 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().

Normal operation is that GraphicUSB will start up, run the script chosen, visibly generate the report, and will then save the required files and exit with an exit code reflecting the degree of success of the script.

4.6.1 Command Line Syntax

The required command line command is:

```
GraphicUSB -ps scriptfilename -pr reportfilename [-pc capturefilename]
```

`scriptfilename` is the name of the script file to be run, including its extension. An example is ‘script1.mpet’.

`reportfilename` is the name of the report file to be generated, including its extension. An example is ‘report1.petrpt’.

`capturefilename` is the name of the capture file to be generated, including its extension. An example is ‘capture1.mqu’. Please note that the capture file is optional.

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.

GraphicUSB returns an exit code. This has the value 0 for no error, 1 if the script used the command **`SendValue TESTFAILEDNONFATAL`**, or 2 if the script used the command **`SendValue TESTFAILEDFATAL`**.

Before and after the script is run, all ETB switches are cleared.

A typical implementation in a C program is:

```
::WinExec("c:\\Program Files\\MQP
Electronics\\GraphicUSB\\GraphicUSB -ps script1.mpet -pr
report1.petrpt -pc capture1.mqu", SW_SHOW);
```

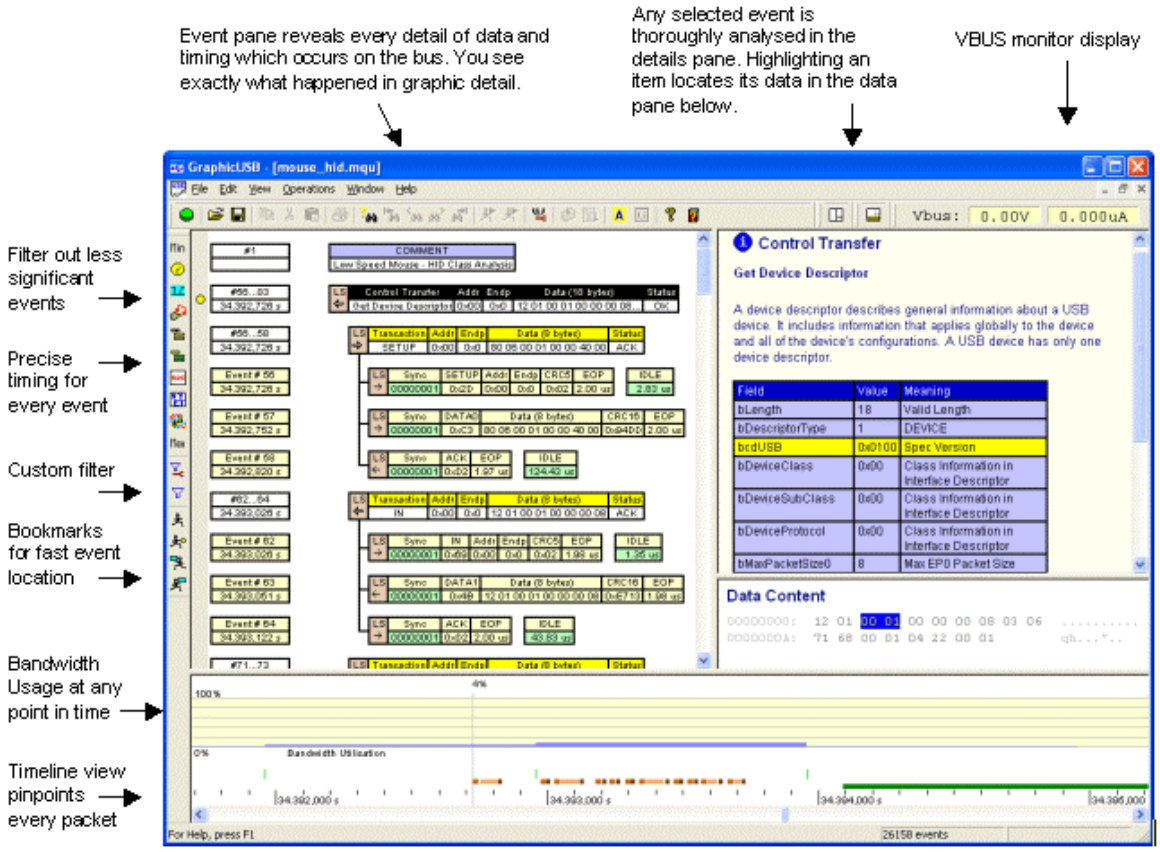
(All on one line in the actual code.)

GraphicUSB SOFTWARE - ANALYSER

4.7 Overview

The PET shares the Capture File format with the other Packet-Master Analysers. A simple, yet information-rich display allows you to view every detail of a capture.

This shows the organization of the screen after a typical capture:

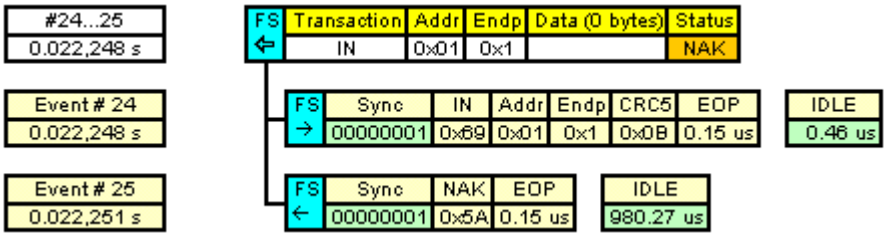


4.8 Panes

4.8.1 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 16.66ns for the USB-PET. A vertical line associates the packets within the transaction. Looking in detail at the display for event #24 below, the following information is displayed:

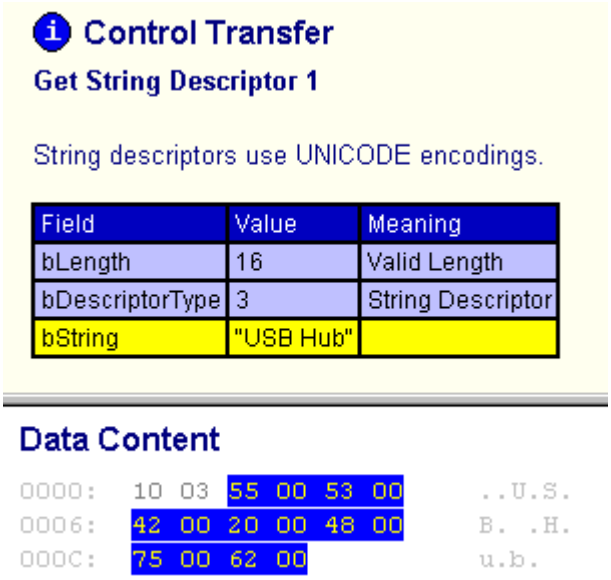
- The speed of the transmission. HS is High Speed (480MHz), FS is full speed (12MHz) and LS is low speed (1.5MHz).
- The direction of the packet. A right pointing arrow is for host to device and a left pointing arrow for device to host.
- The Synchronisation field. The bit pattern received is shown.
- 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.



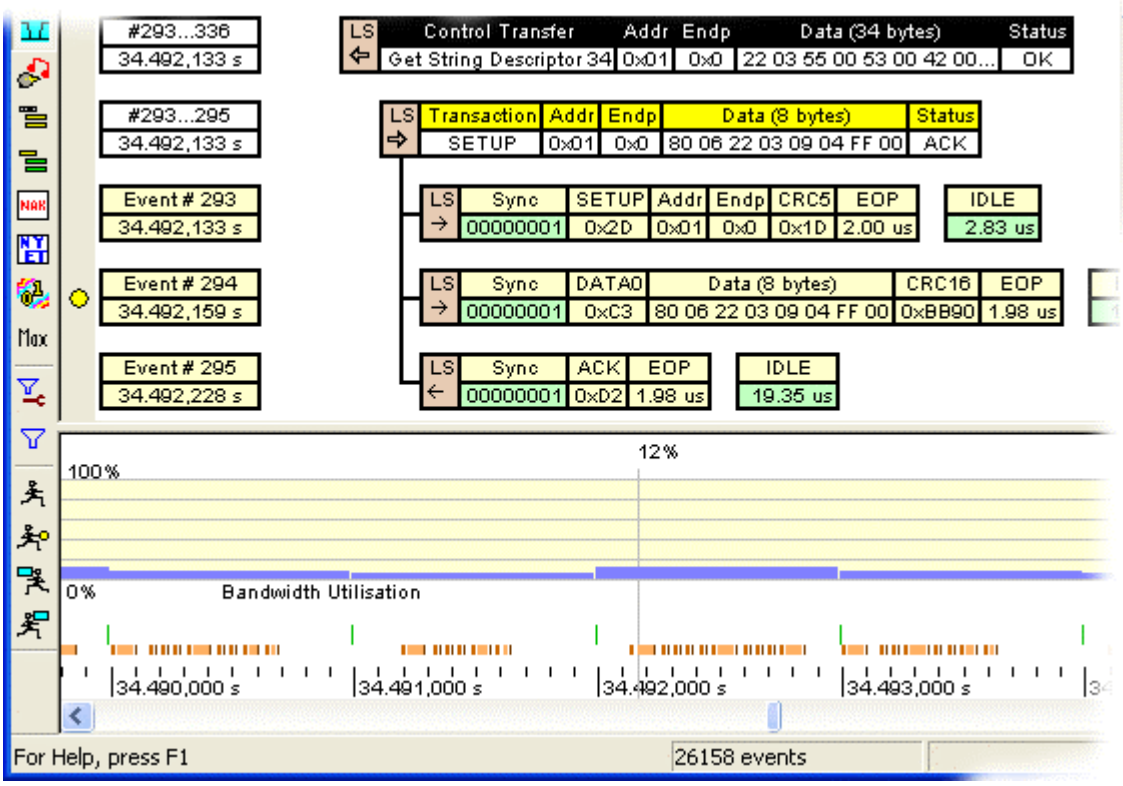
An error e.g. an incorrect CRC is indicated by the appropriate field being highlighted in red. A field highlighted in orange indicates a potential problem or warning.

4.8.2 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.



4.8.3 Timeline and Bandwidth Panes



Across the bottom of the window is the timeline and bandwidth utilization 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.

4.8.3.1 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.

4.8.3.2 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.

4.8.3.3 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.

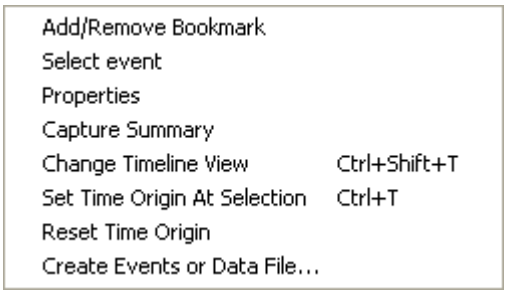
To be certain of identifying a particular packet, first click the Max button to the left of the event pane.

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

4.9 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.

4.9.1 Event Pane Properties



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

4.9.2 Detail Pane Properties

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

4.9.3 Data Pane Properties

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

4.9.4 Timeline Pane Properties

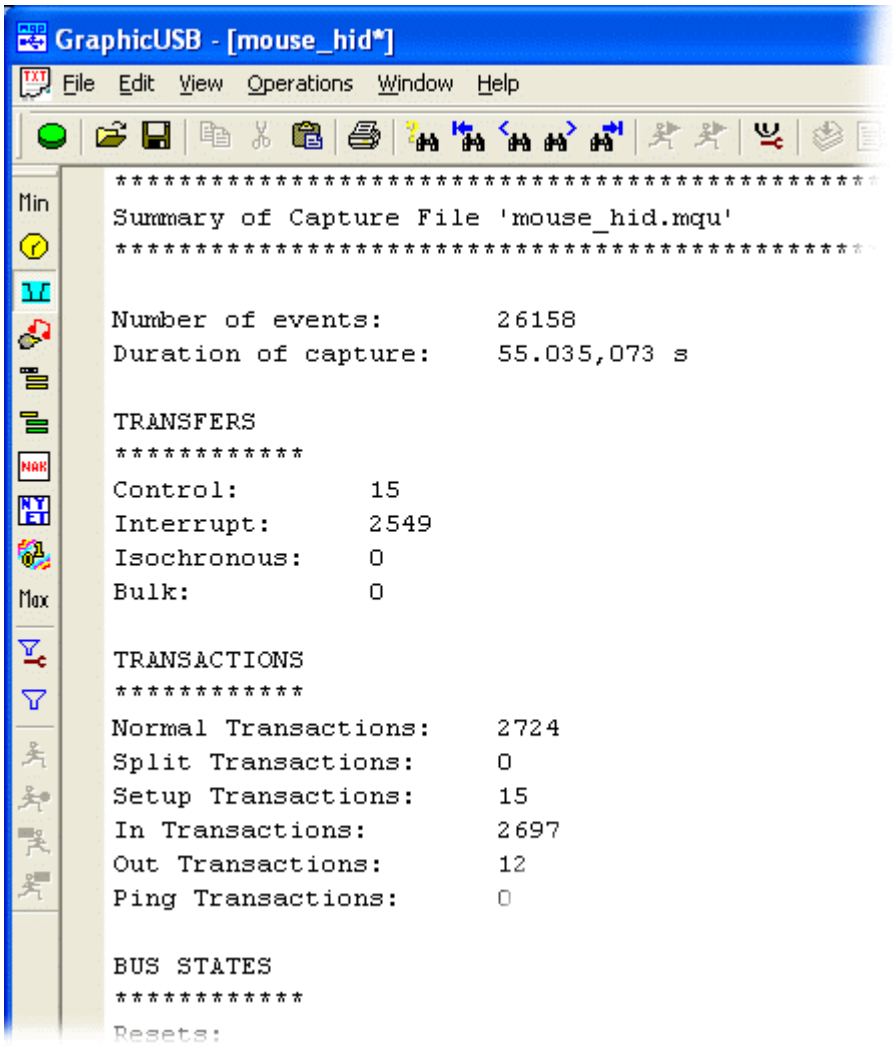
'Change Timeline View' (or Ctrl+Shift+T) allows you to show or hide parts of the timeline view.

4.10 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.



4.11 Toolbars

4.11.1 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 (PET Script)

 Run (PET Script)

4.11.2 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

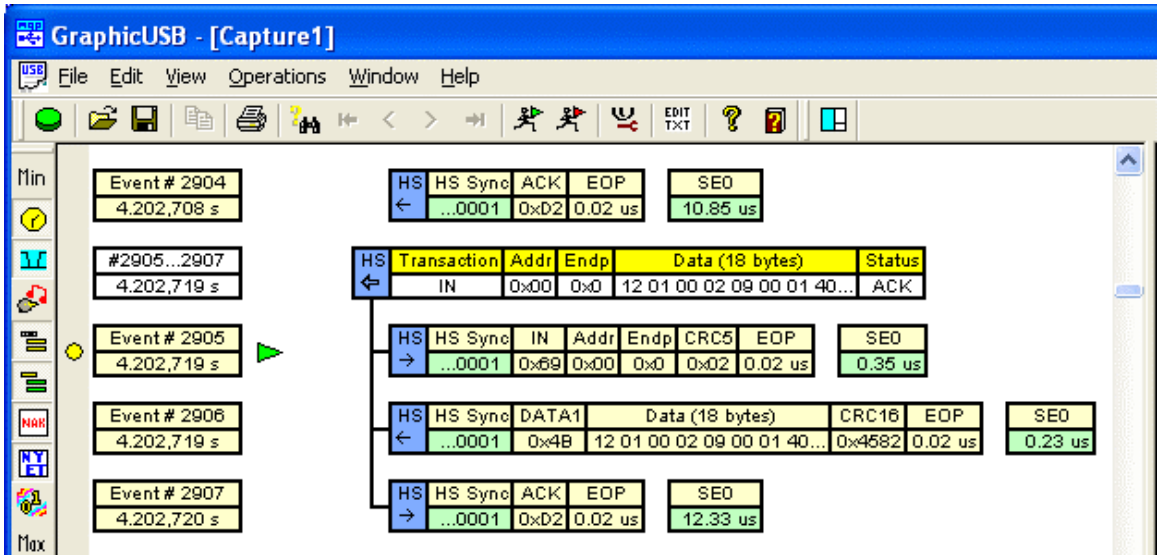
4.12 Capturing

Capturing is enabled by checking the 'Do Analyser Capture' check box in the dialog used to run the script(s). The capture file wil be generated underneath the report text file.

4.12.1 Finding the Trigger Points

If the capture file contains a Trigger Start or Trigger Stop event (does not apply to PETs) 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.



4.13 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.

	FS	Control Transfer	Addr	Endp	Data (0 bytes)	Status
#81...87 5.126,195 s	⇨	Set Address (0x01)	0x00	0x0		OK

	FS	Transaction	Addr	Endp	Data (8 bytes)	Status
#81...83 5.126,195 s	⇨	SETUP	0x00	0x0	00 05 01 00 00 00 00 00	ACK

	FS	Transaction	Addr	Endp	Data (0 bytes)	Status
#85...87 5.127,195 s	⇦	IN	0x00	0x0		ACK

	FS	Control Transfer	Addr	Endp	Data (18 bytes)	Status
#127...145 5.166,196 s	⇦	Get Device Descriptor	0x01	0x0	12 01 00 02 00 00 00 08...	OK

	FS	Transaction	Addr	Endp	Data (8 bytes)	Status
#127...129 5.166,196 s	⇨	SETUP	0x01	0x0	80 06 00 01 00 00 12 00	ACK

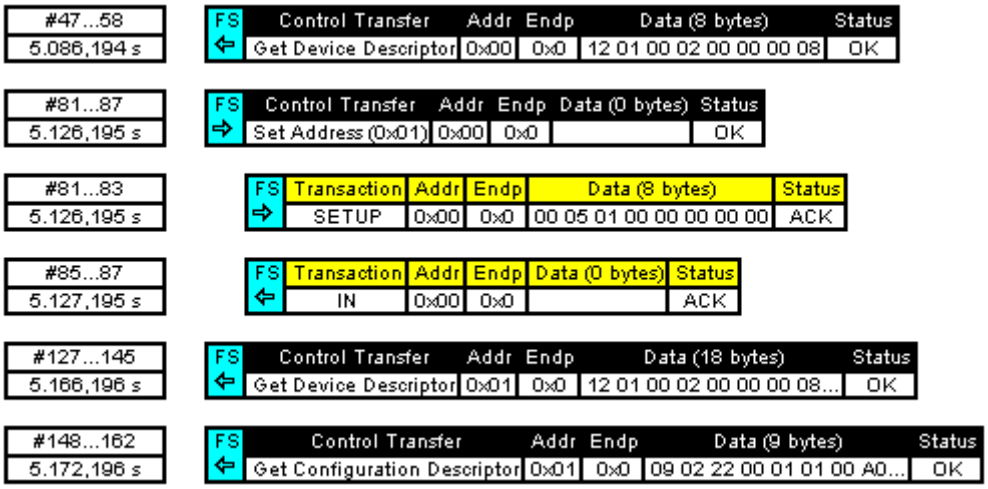
Show Transactions

	FS	Control Transfer	Addr	Endp	Data (0 bytes)	Status
#81...87 5.126,195 s	⇨	Set Address (0x01)	0x00	0x0		OK

	FS	Control Transfer	Addr	Endp	Data (18 bytes)	Status
#127...145 5.166,196 s	⇦	Get Device Descriptor	0x01	0x0	12 01 00 02 00 00 00 08...	OK

Hide Transactions

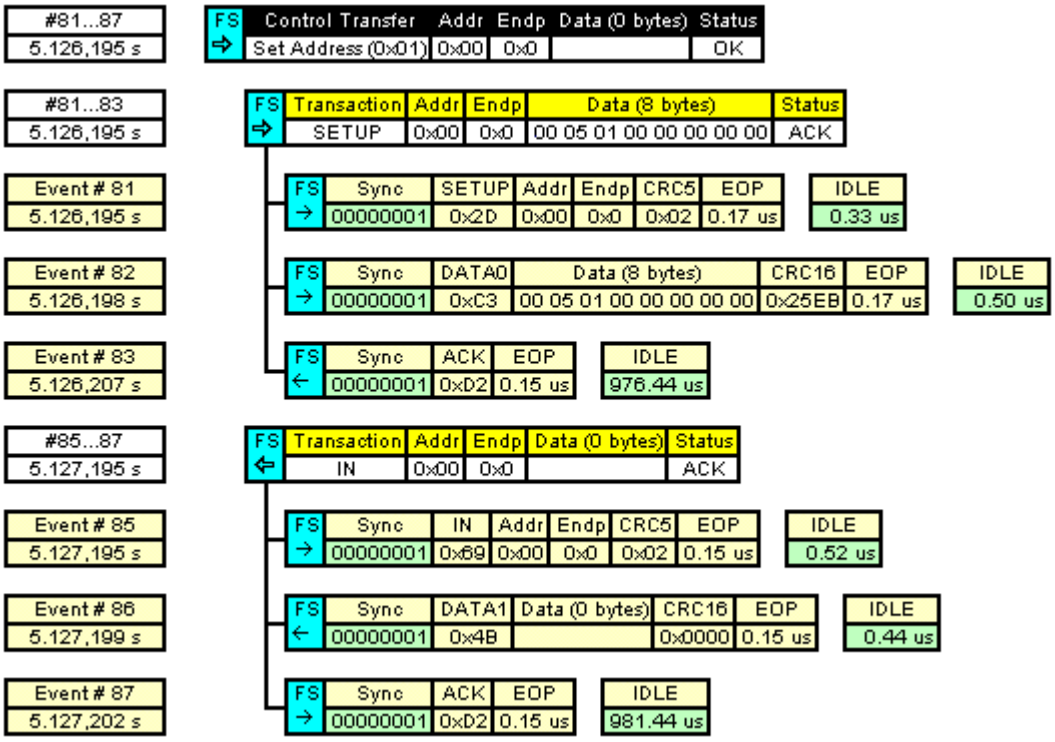
If transactions within control transfers have been filtered out, then double clicking on a particular control transfer will reveal the transactions within it, as shown below.



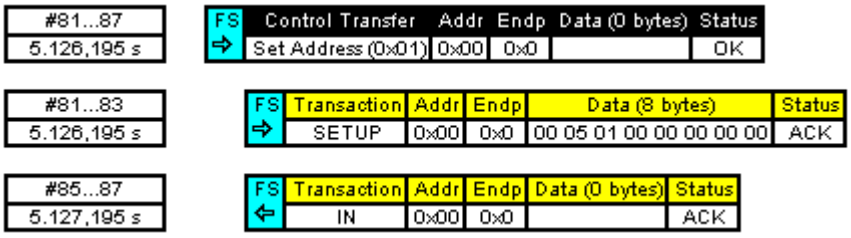
Show Selected Transactions

Show Packets

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

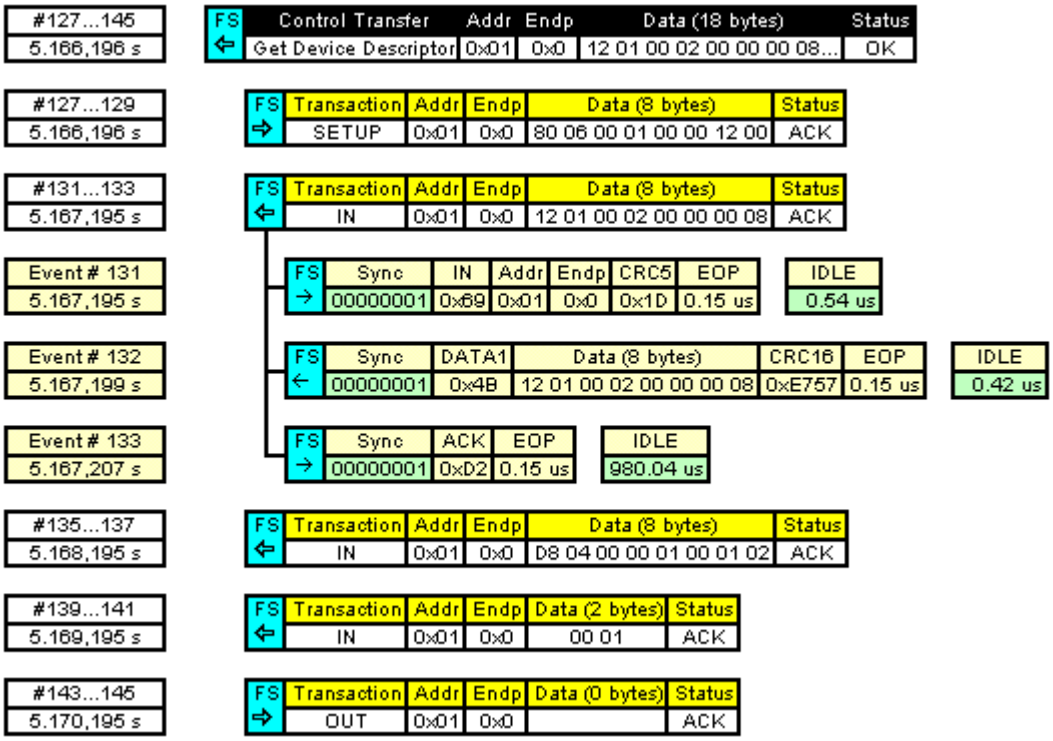


Show Packets



Hide Packets

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



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.

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.

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.

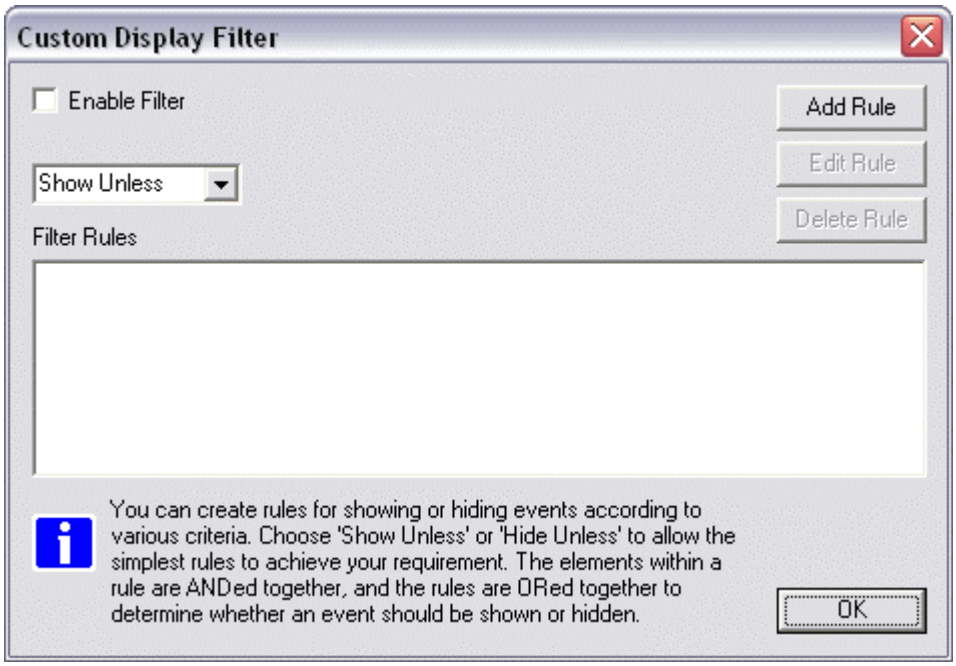
4.14 Custom Filter

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.

4.14.1 Custom Filter Settings

Custom Filter Settings

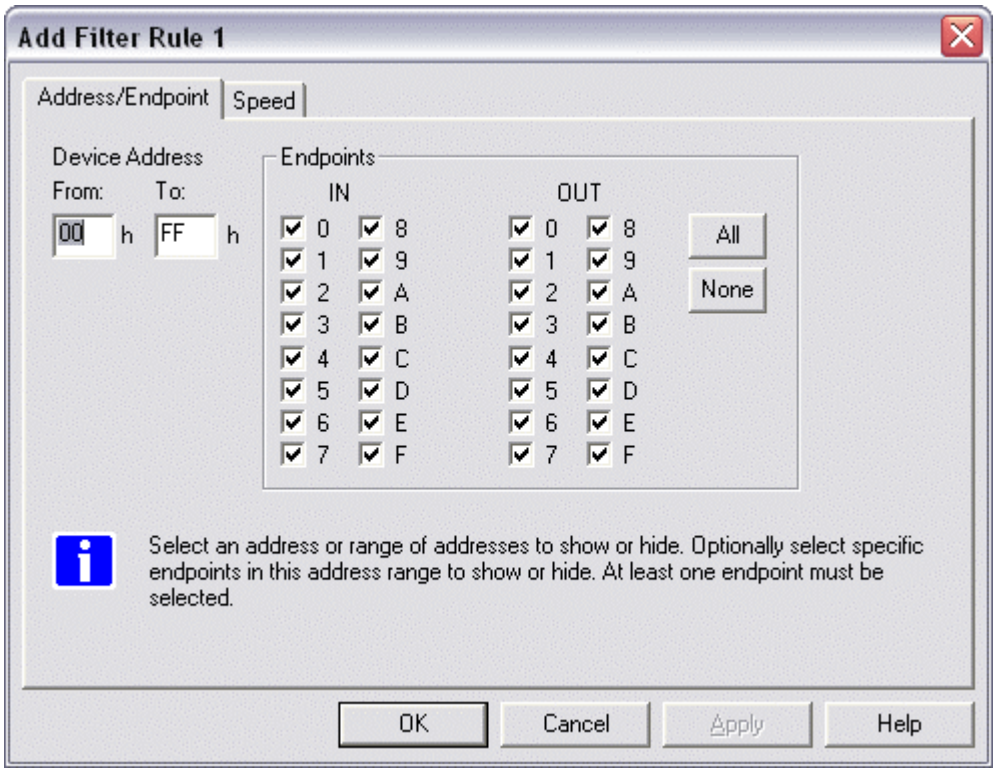
When you click on the ‘Custom Filter Settings’ button (or select it from the View menu), the following dialog appears:



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

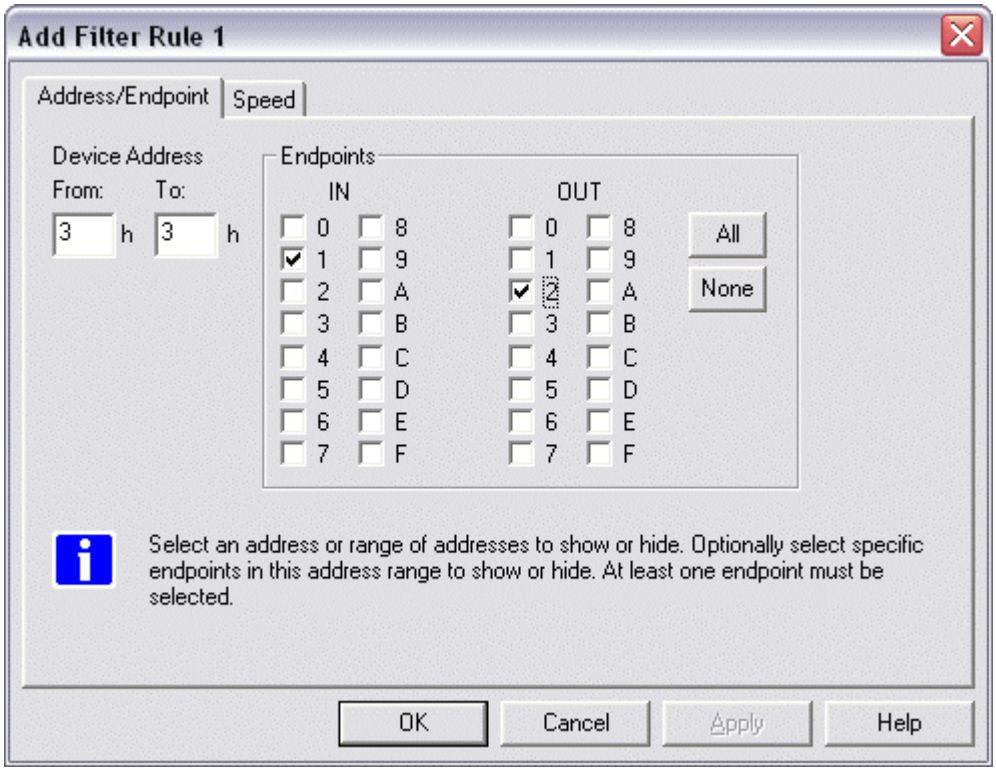
add your first rule the filter will automatically be enabled, though you can choose to disable it again.

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

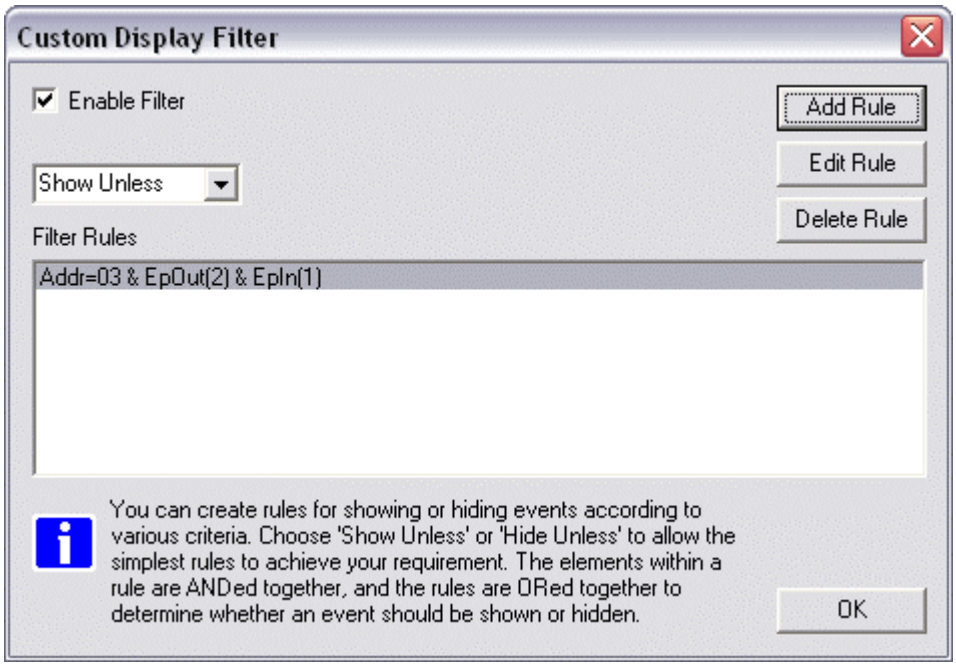


There are two tabs to select the conditions for your rule. Both tabs can contribute to the rule if required. The first tab defines device address and endpoint. The default settings include all possible addresses and endpoints, so if you do not make a change the rule will have no effect, and you will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

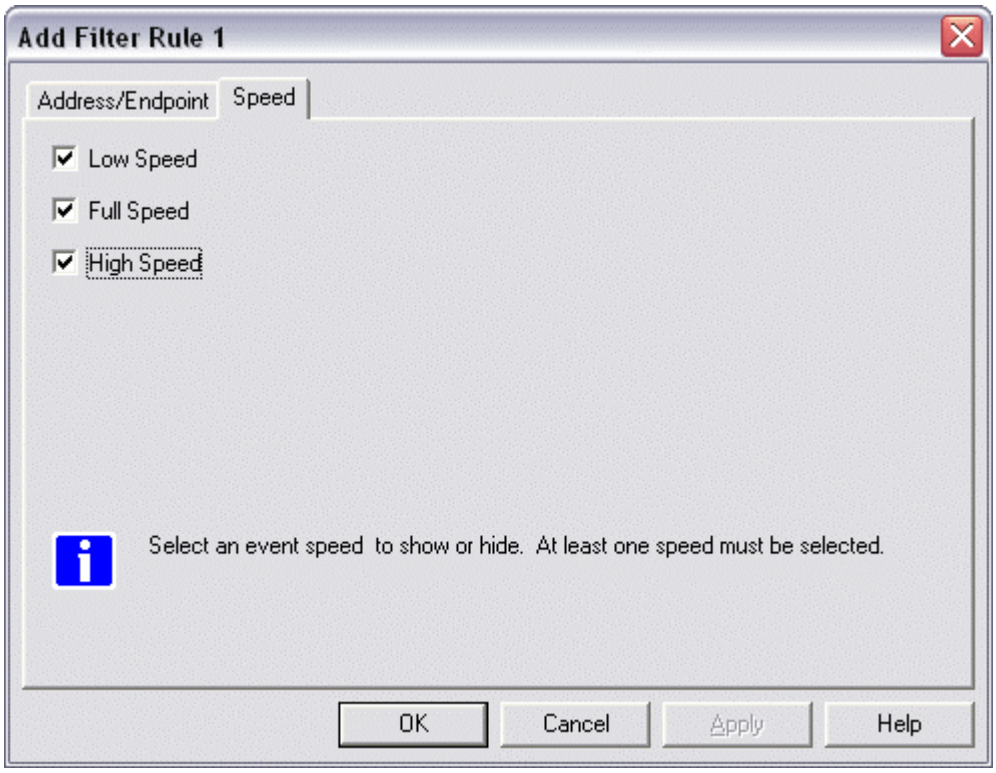
As an example we specify here that we do not wish to see transactions using (address 3) AND ((endpoint 1 in) OR (endpoint 2 out)):



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:



4.14.2 Custom Filter Enable

Custom Filter Enable

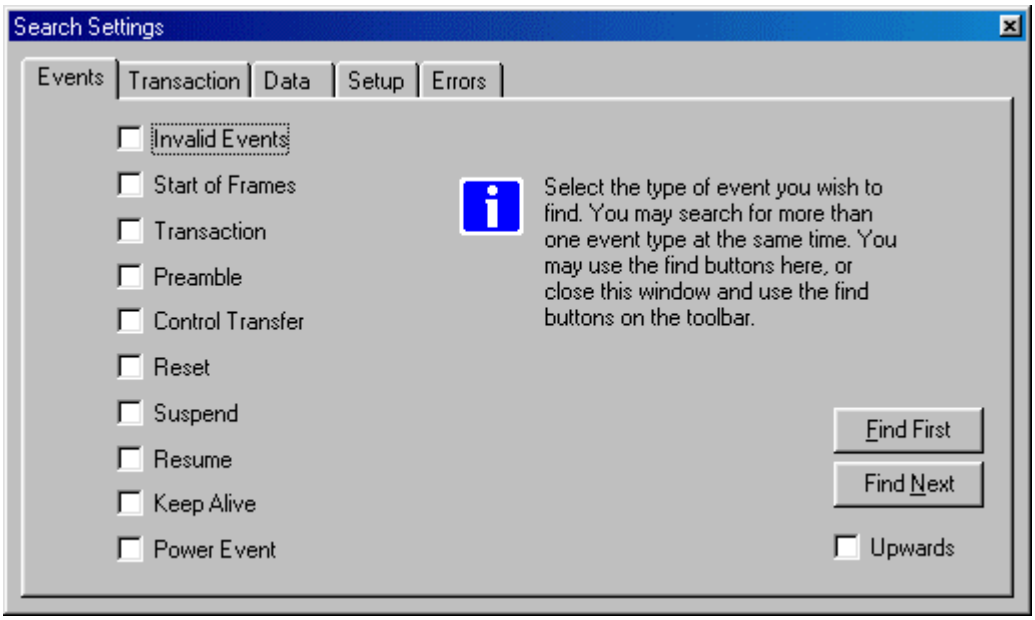
The filter rules are global to the application, and can quickly be turned on and off using the custom filter enable button. The Menu item in the View Menu can also be used.

4.15 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.

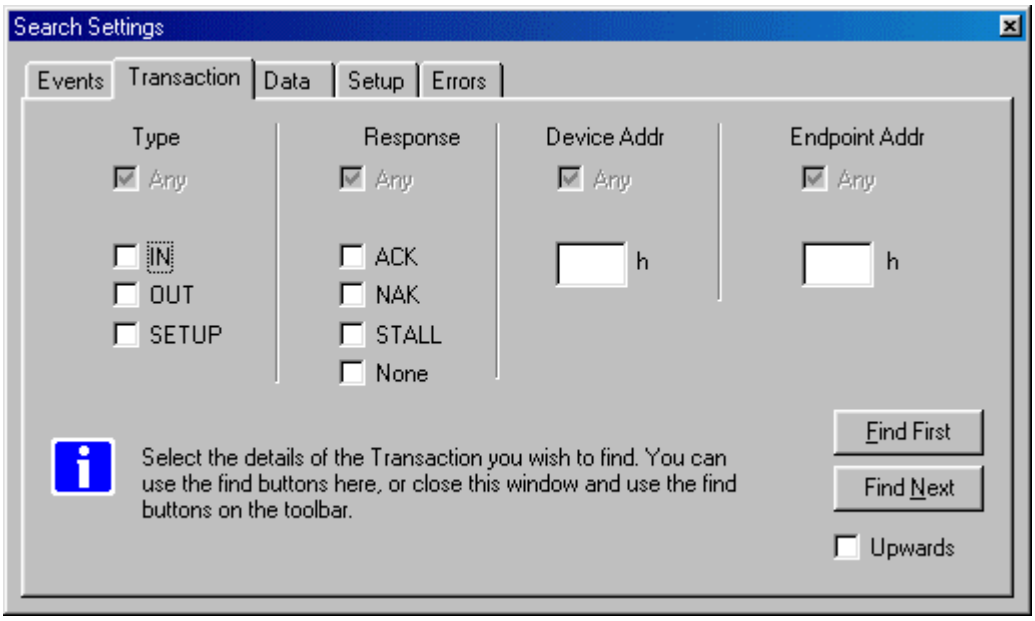
4.15.1 Event Search

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



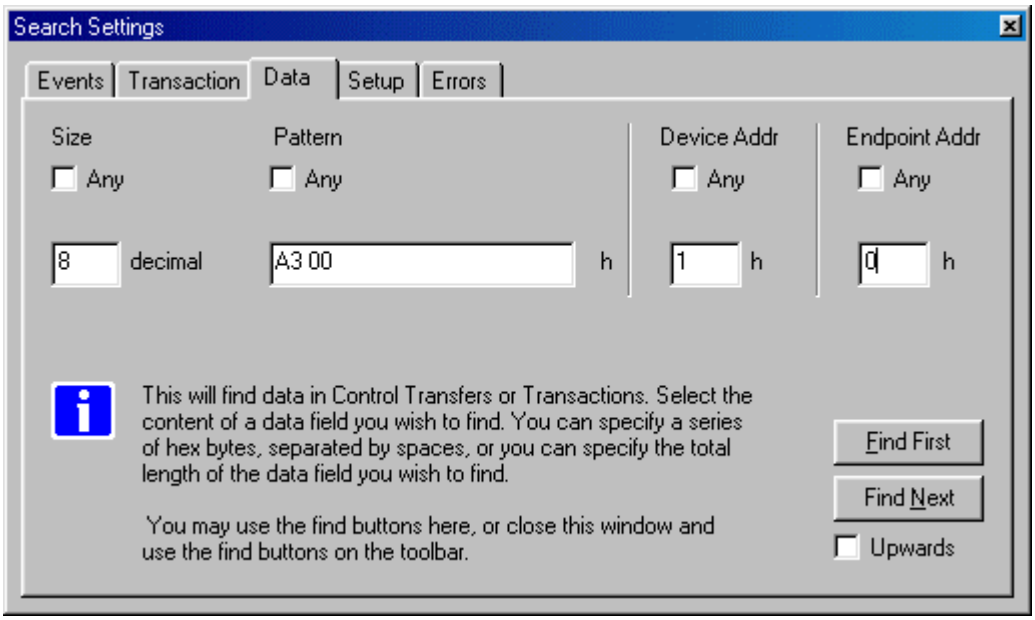
4.15.2 Transaction Search

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.



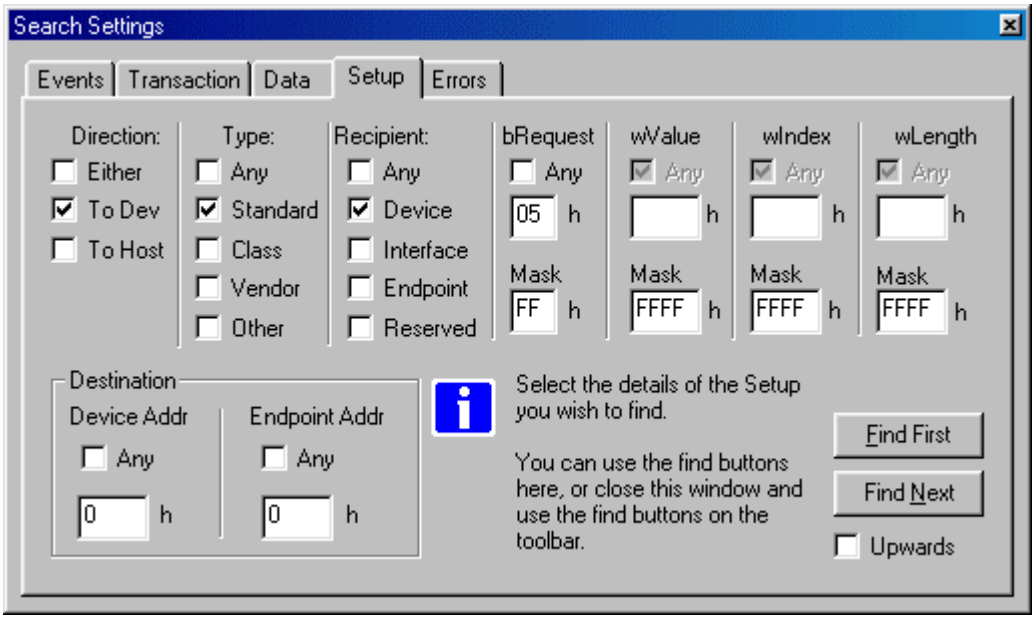
4.15.3 Data Search

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.



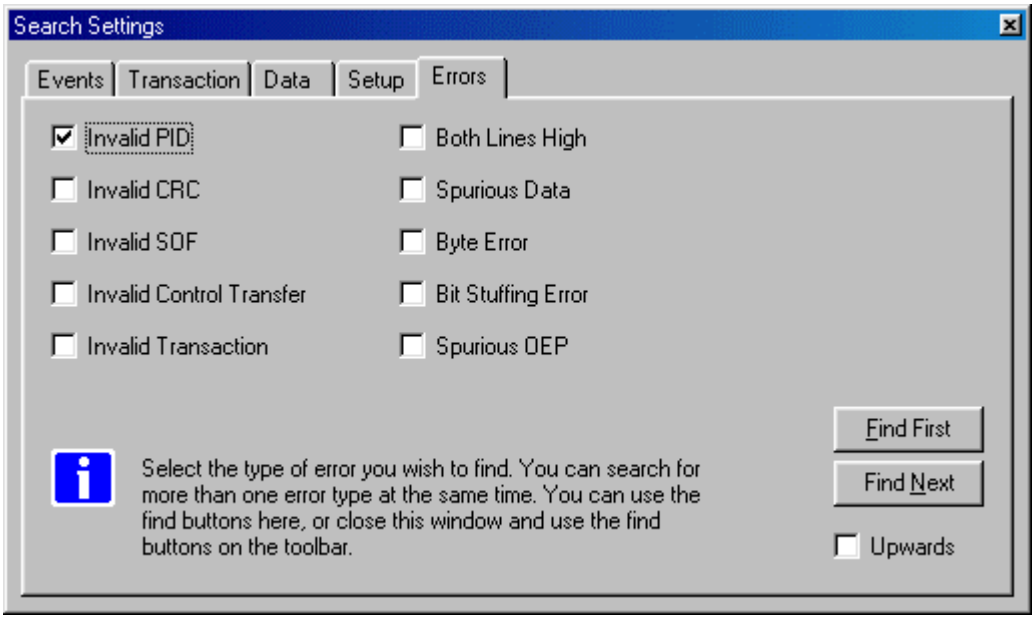
4.15.4 Setup Search

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.



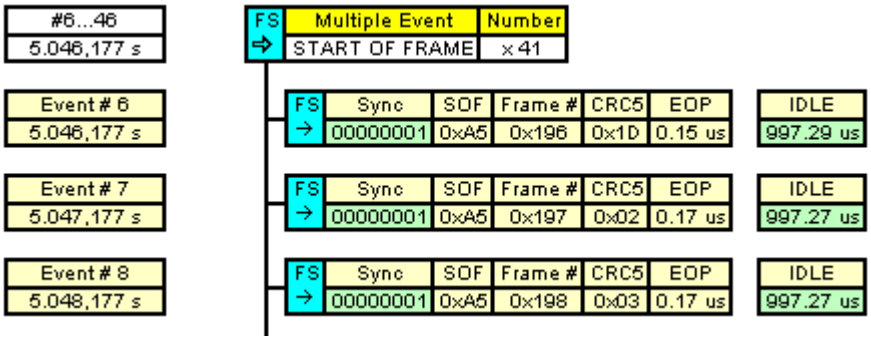
4.15.5 Error Search

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.



4.16 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

4.17 Bookmarks

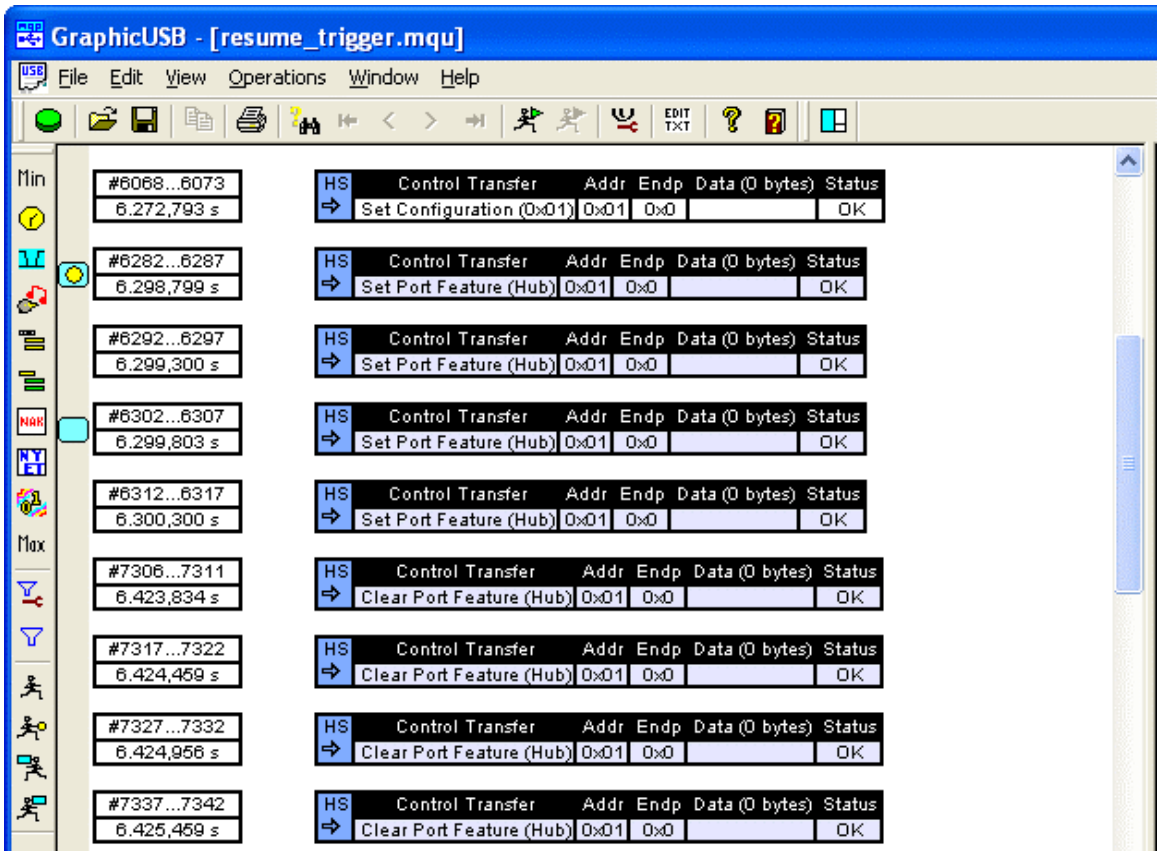
A bookmark allows you to mark an event of interest, allowing you to locate it quickly when it is not showing in the window.

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

4.18 Printing

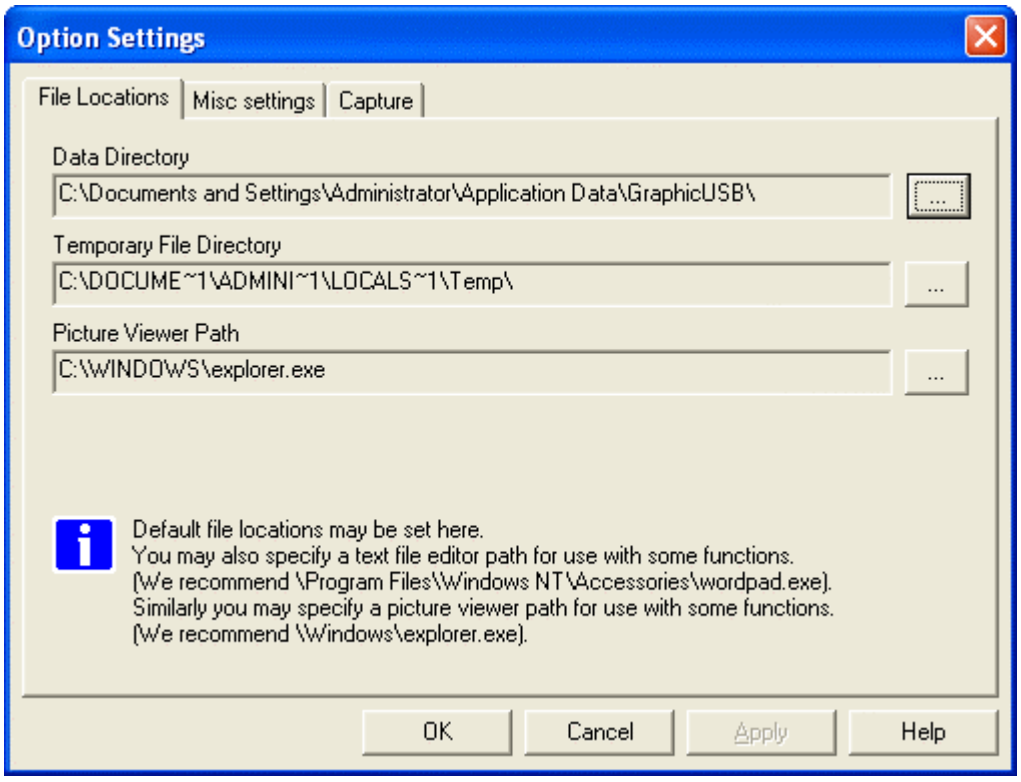
Any of the panes may be printed. To select a pane to print, click on that pane, or click on the Select Print Pane icon on the tool bar until the required pane is indicated. Then print in the usual way, either from the file menu, or using the print icon on the tool bar.

4.19 Option Settings

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

4.19.1 File Locations

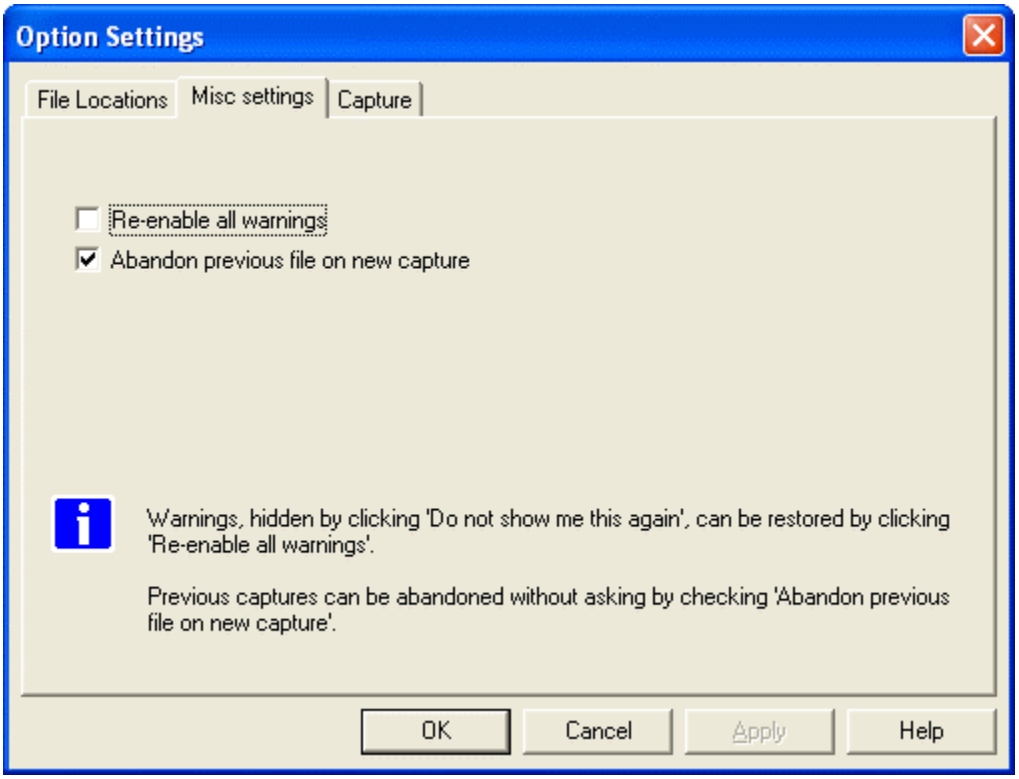
Use this to specify the locations of the Capture and temporary files.



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

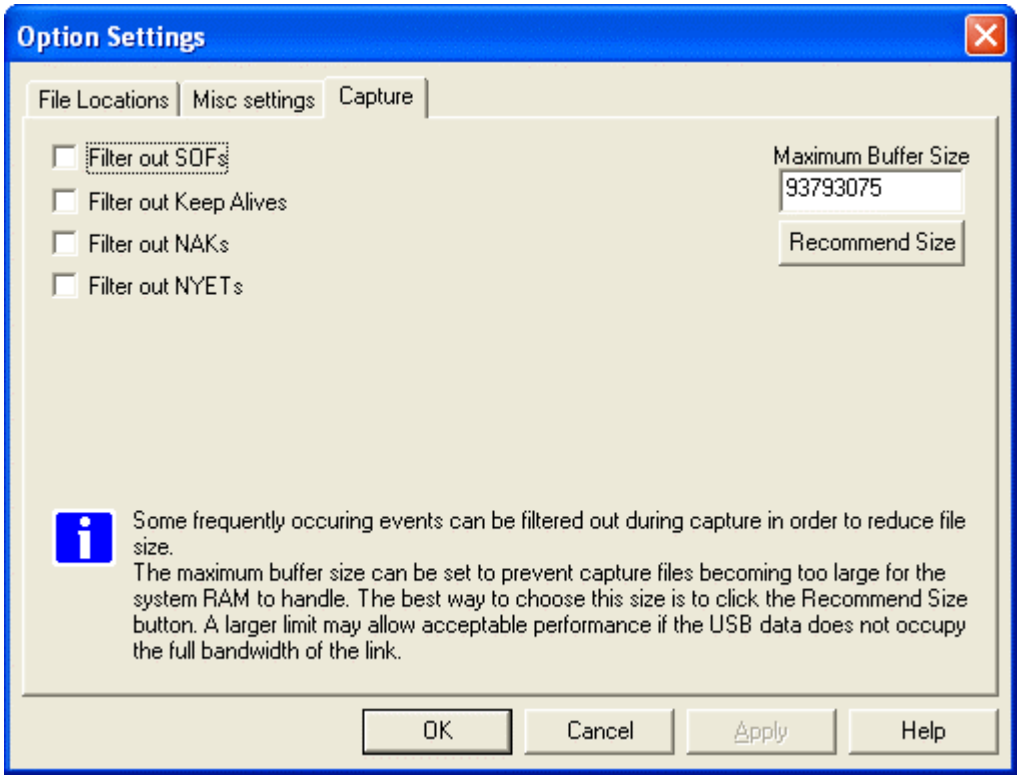
4.19.2 Miscellaneous Settings

- 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.



4.19.3 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.

4.20 Export Functions

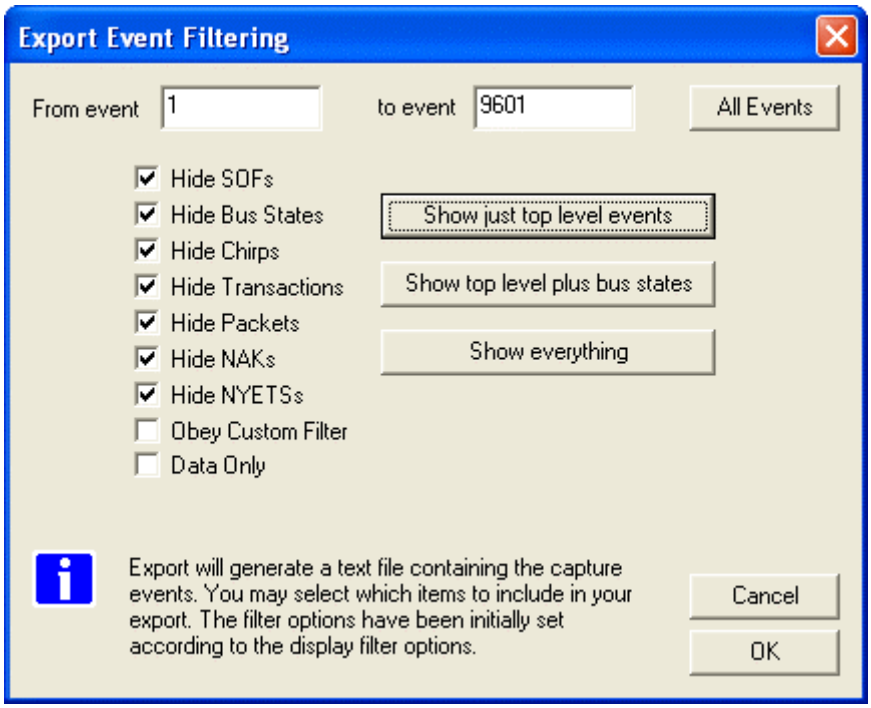
GraphicUSB allows various types of information to be exported to text based formats for further analysis or processing by the user. The following types of export are currently provided:

- 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.

4.20.1 Exporting Capture Events

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

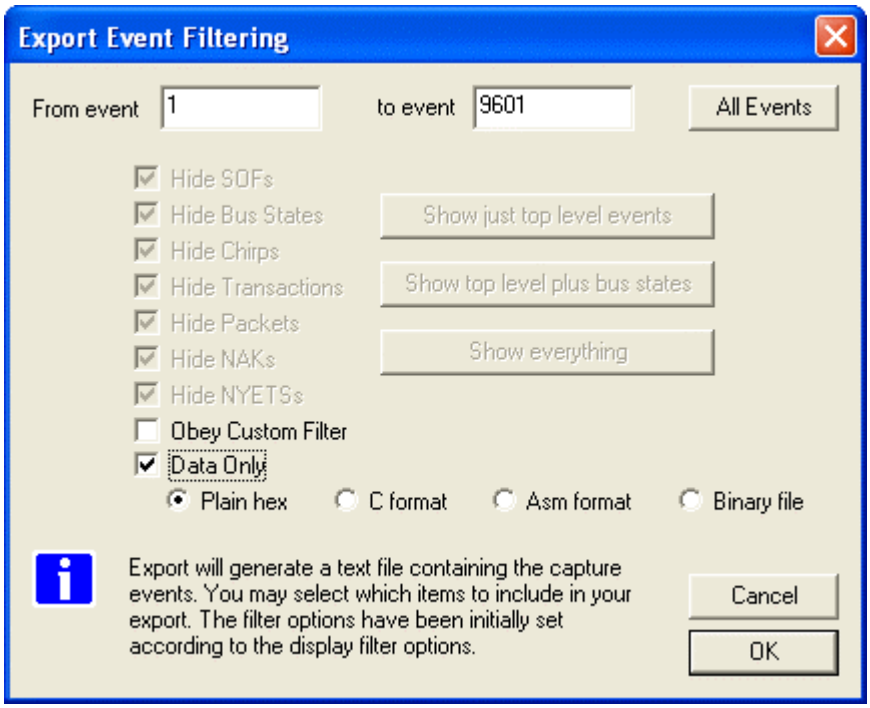


Select the event range, and the level of detail you wish to export, then click on OK. A typical output text file (showing just top level events) is shown below:

```
[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
```


4.20.2 Exporting Capture Events – Data Only

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

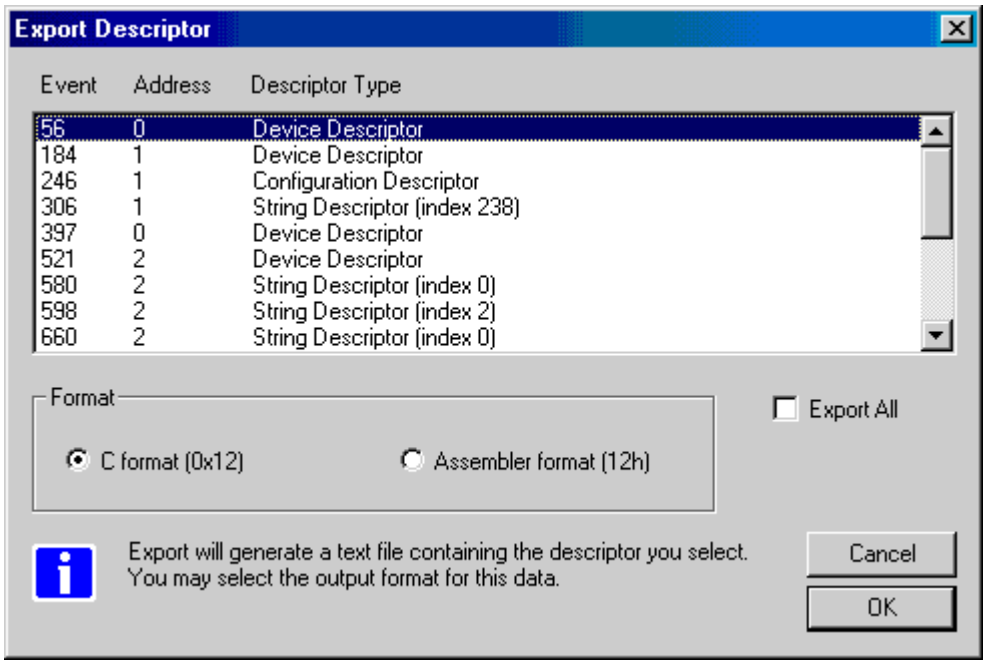


Selecting the ‘Data Only’ option allows data, as selected by the custom filter, and the event range, to be exported in a number of useful file formats.

4.20.3 Exporting Descriptors

4.20.3.1 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
};
```


4.20.3.2 Class Specific Descriptors

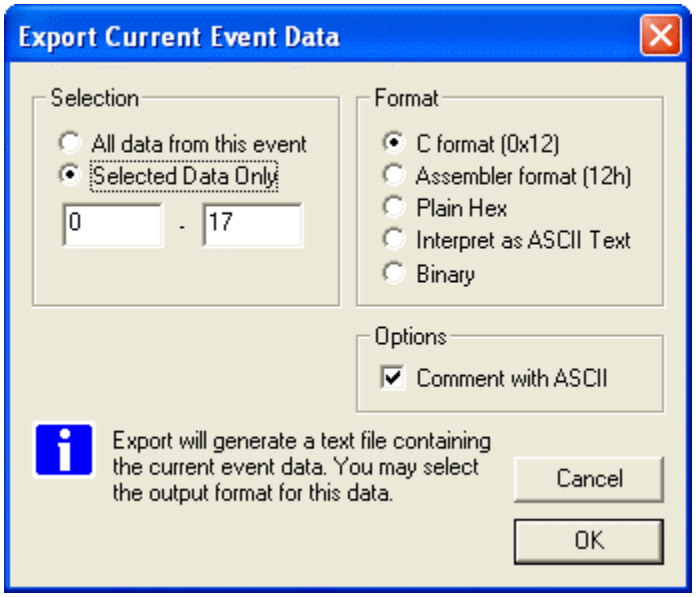
In some cases, where class analysis options have been installed, specific class descriptors are annotated, e.g.:

```
// 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,       //
```

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

4.20.4 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,
```


4.21 Text Editing

4.21.1 Introduction

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

- PET script file (*.mpet)
- exported event file (*.txt)
- exported data file (*.txt)
- exported descriptor file (*.c, *.asm)
- exported current event data file (*.c, *.asm)
- 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.

- PET script file (*.mpet)
- generator script
- vendor class information file
- device information file

This is used to display validation or compilation output.

4.21.2 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

4.21.3 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.

4.21.4 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.

4.22 USB Errors

4.22.1 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.

4.22.2 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.

4.22.3 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.

4.22.4 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.

4.22.5 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.

4.22.6 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.

4.22.7 Byte Error

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

4.22.8 Spurious Data

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

4.22.9 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.

4.22.10 Spurious End of Packet

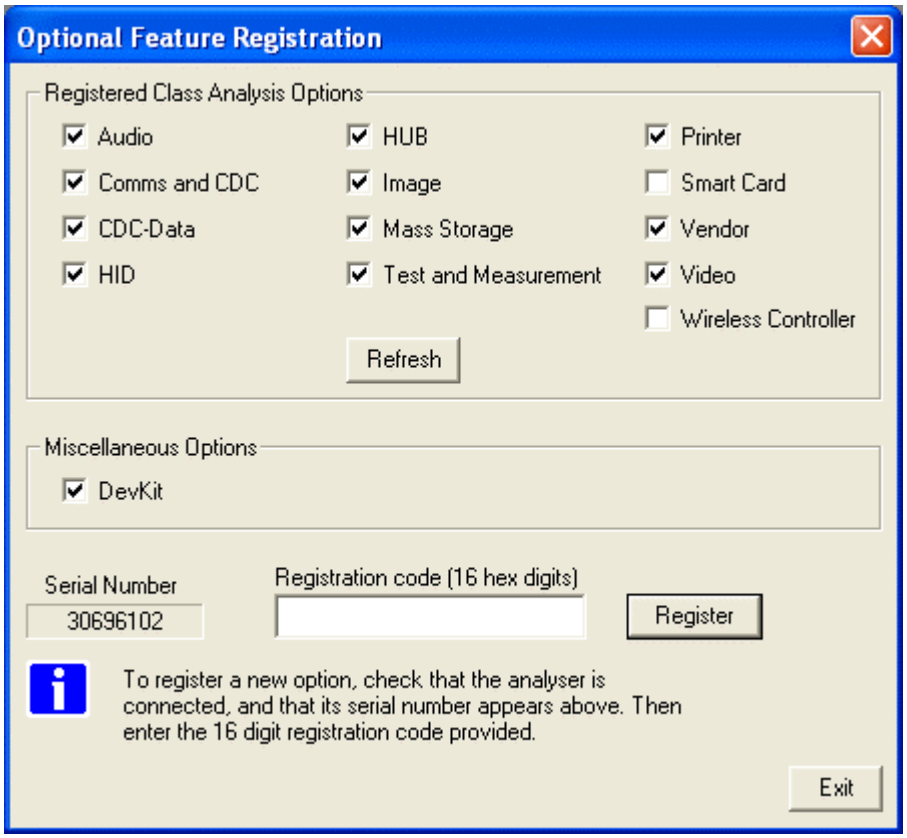
An End of Packet condition should only appear at the end of a data packet. If the condition appears at any other time it will be reported as an error.

4.23 Class Analysis Options

4.23.1 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.



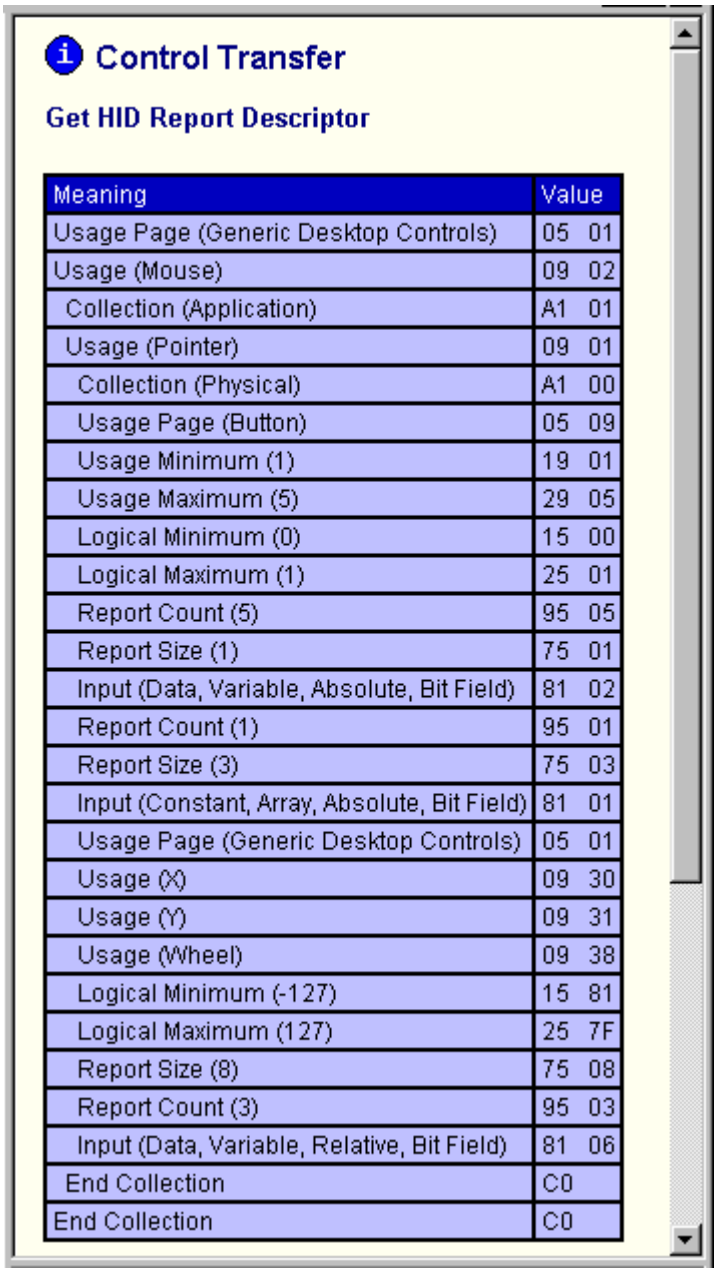
Enter the registration code provided and the corresponding option should become checked. Please store the registration code carefully in case you need to install the option on another host computer.

4.23.2 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.

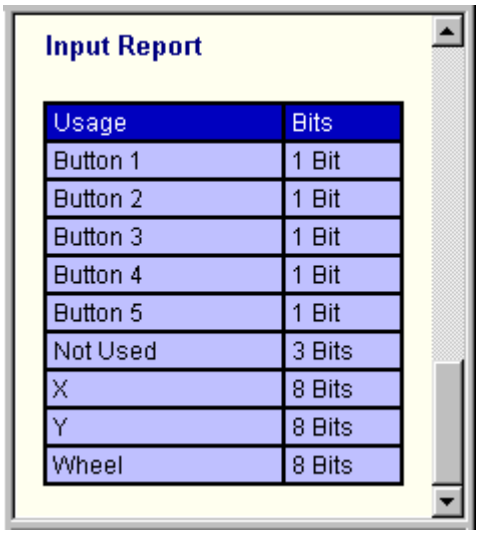


Control Transfer

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

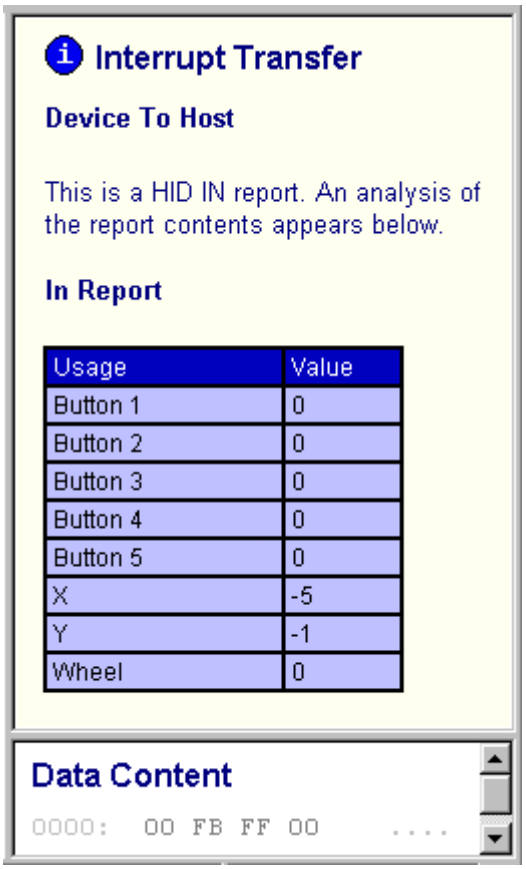
This shows a HID Report Descriptor, and below is the result of parsing it.



Input Report

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.



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 ....

4.23.3 Vendor Class Analysis

4.23.3.1 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

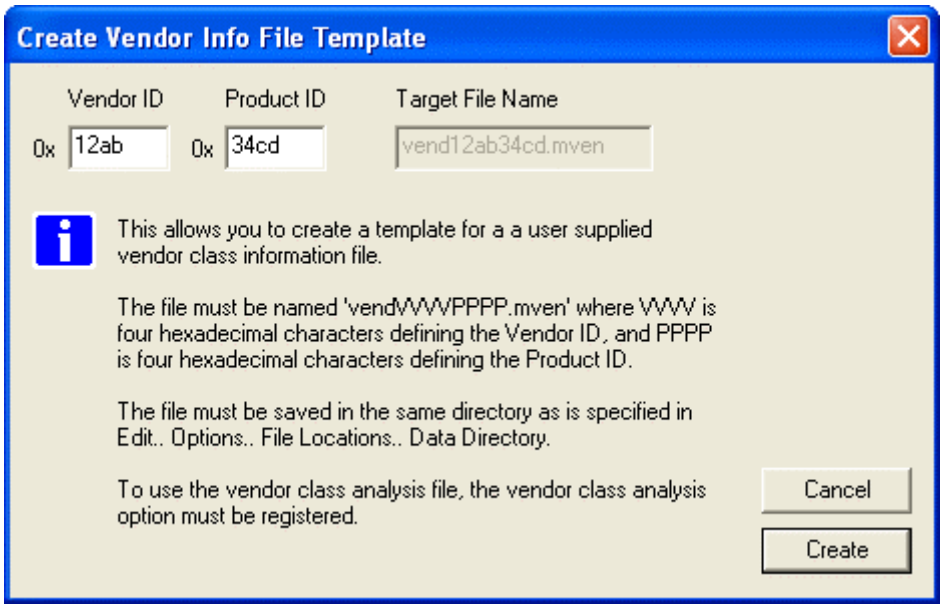
where VVVV is four hexadecimal characters defining the Vendor ID, and PPPP is four hexadecimal characters defining the Product ID.

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.

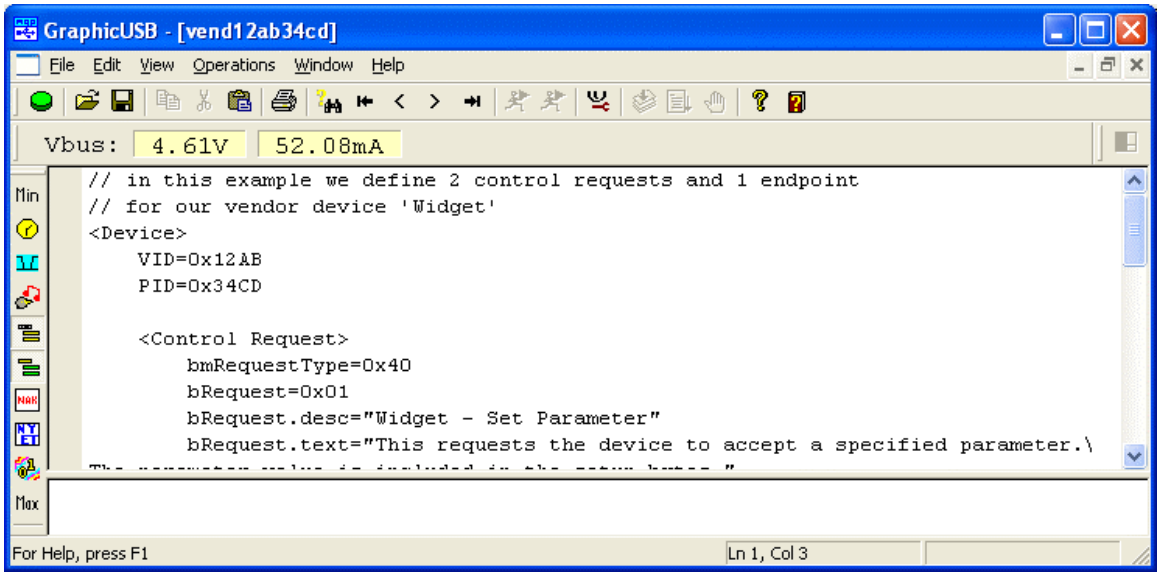
On running the new GraphicUSB for the first time, the application will offer to make these changes automatically for you.

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.\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\
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>
```


4.23.3.2 File Syntax

4.23.3.2.1 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.

4.23.3.2.2 Indentation

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

4.23.3.2.3 Numbers

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

4.23.3.2.4 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.

4.23.3.2.5 <Device> </Device>

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

4.23.3.2.6 VID= PID=

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

4.23.3.2.7 <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	**Must come first.** 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.

4.23.3.2.8 <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.

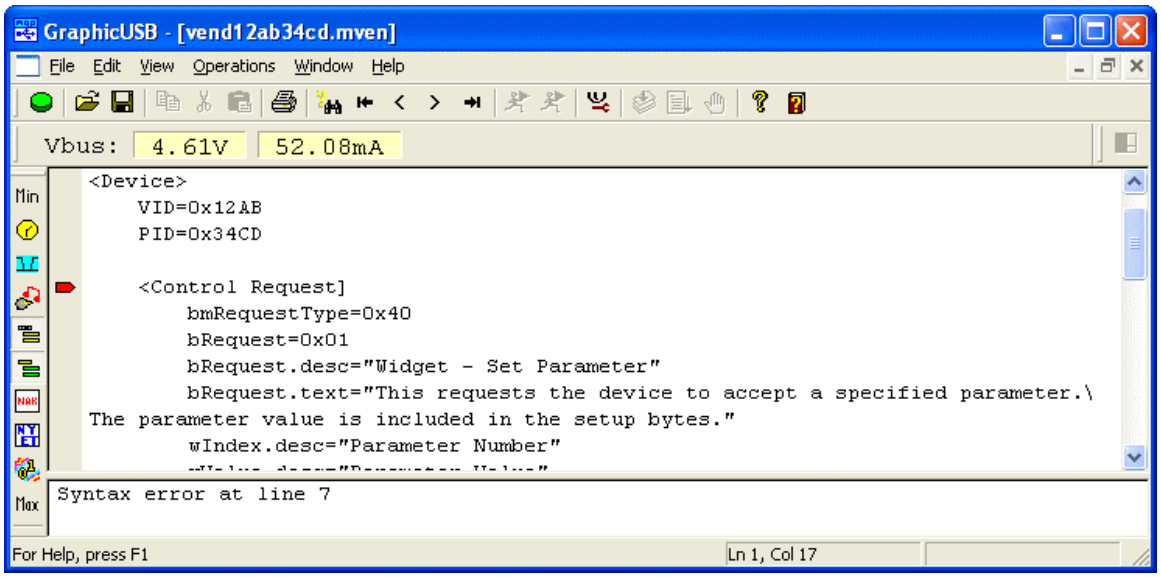
4.23.3.2.9 Syntax Checking

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.

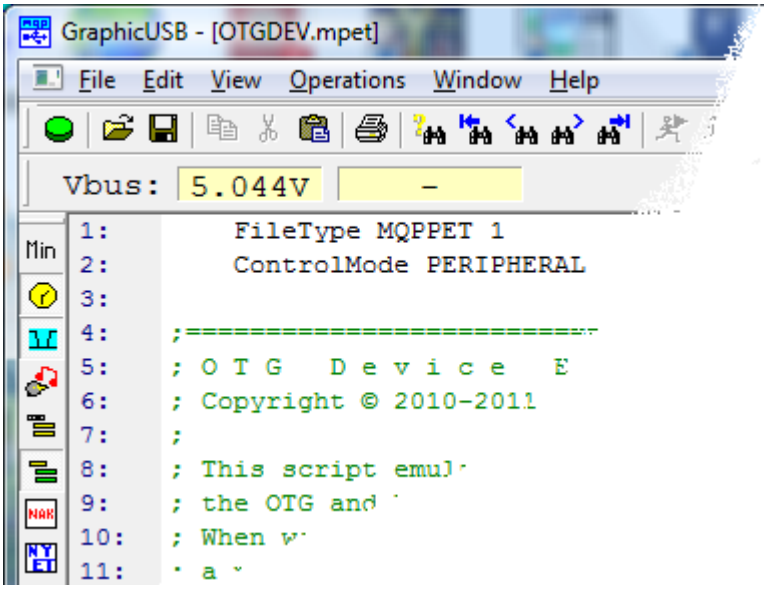
When satisfied, save the file, using the default filename, and ensuring that it is being saved in the data folder specified for the application.

4.24 V_{BUS} Voltage Measurement

The Packet-Master USB-PET has continuous V_{BUS} voltage monitoring circuitry. This is useful to give an early indication of hardware or software problems related to USB power supply.

See Technical Data section for accuracy information.

GraphicUSB will continuously display the voltage value on its toolbar.

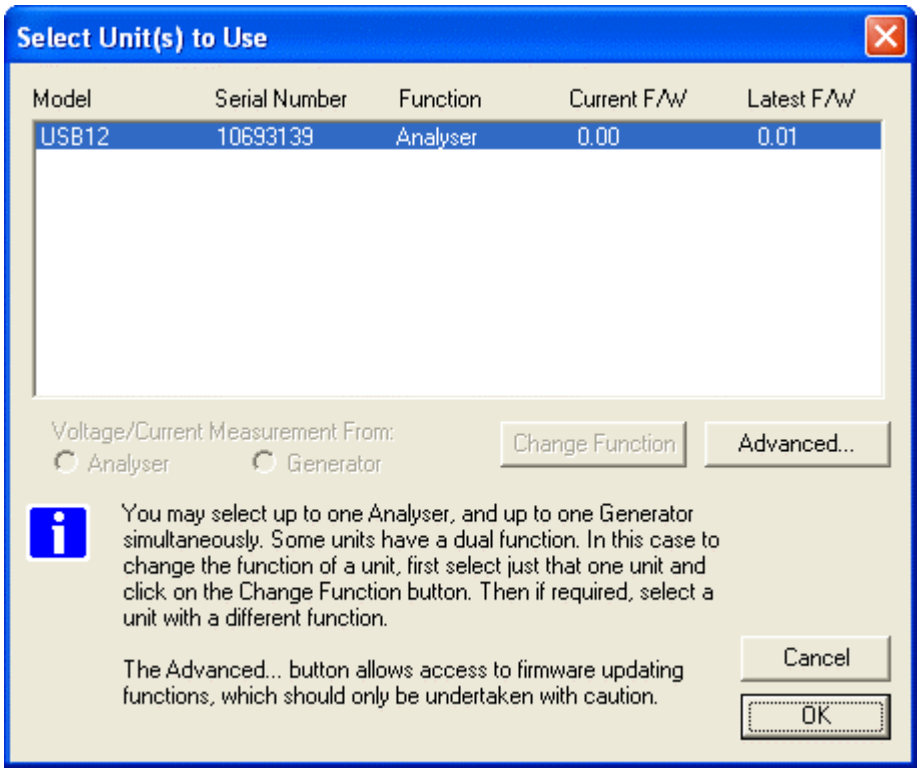


4.25 Firmware Updates

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

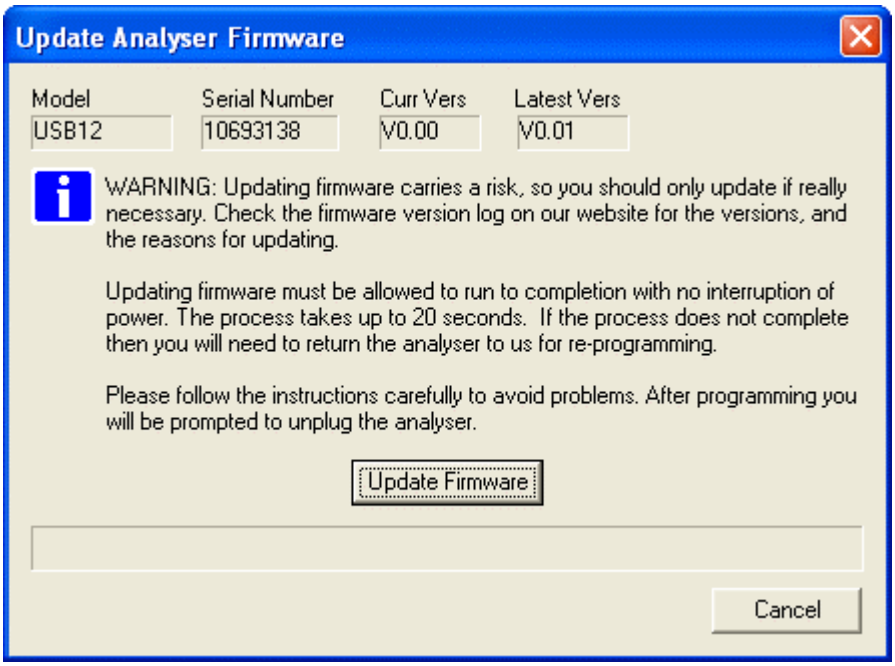
> ***Caution***
>
> Updating firmware is not without its risks. If the update process is interrupted by a power failure, USB cable disconnection or any other similar problem, then it is possible to leave the analyser unit in a non-working state. So the firmware should only be updated for a valid reason.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

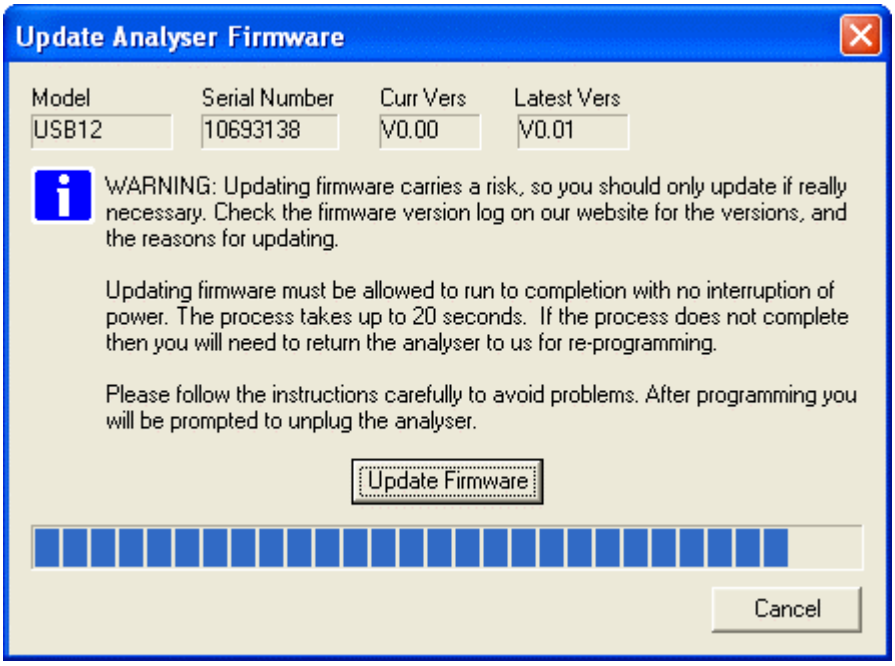


Each connected unit will be displayed. In the example above, the current version (in the unit) is 0.00 and the latest available version is 0.01. Checking on our website will reveal that this update is required from GraphicUSB V3.00 onwards, so we advise proceeding with the update.

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



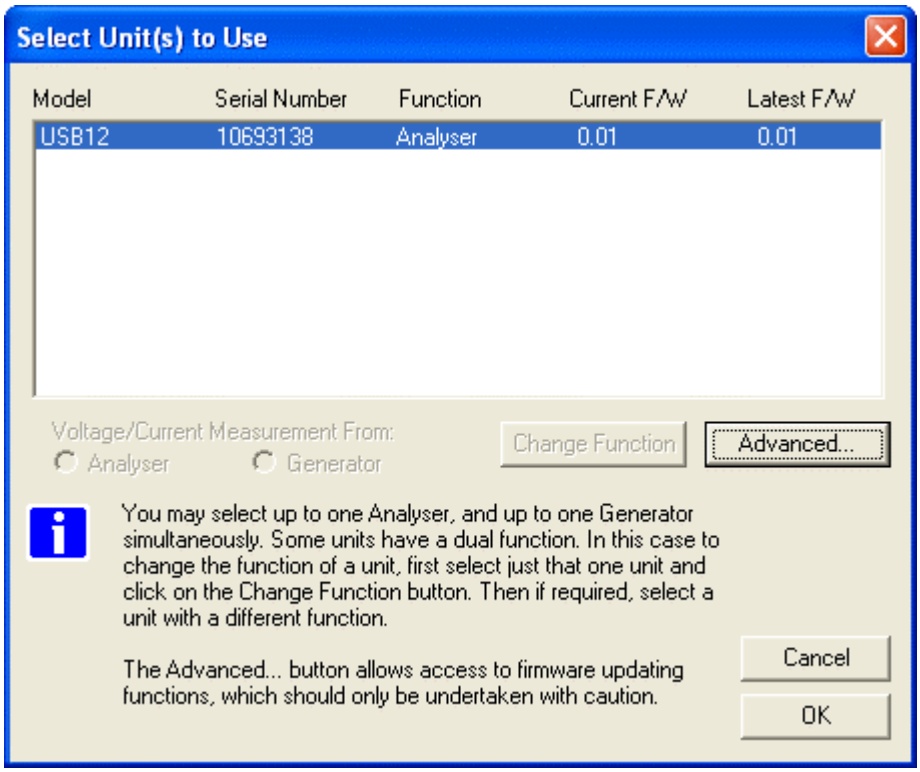
It is important to follow the instructions very carefully, step by step. First click on the Update Firmware button. The progress of the update process, which takes around 20 seconds, will be indicated on the progress bar.



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.

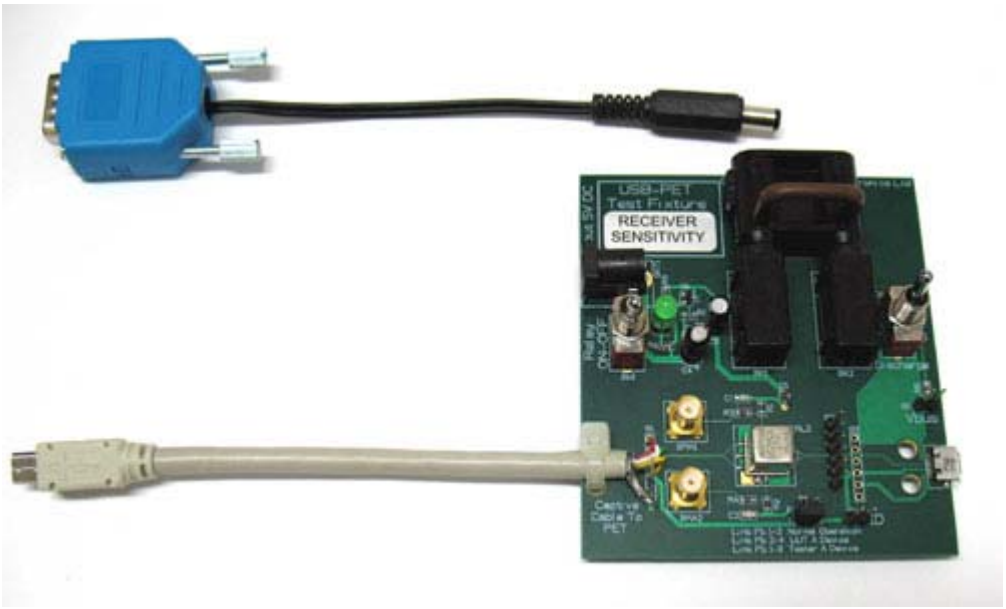


5 HIGH SPEED ELECTICAL TESTS

5.1 Test Fixtures for use with Packet-Master USB-PET

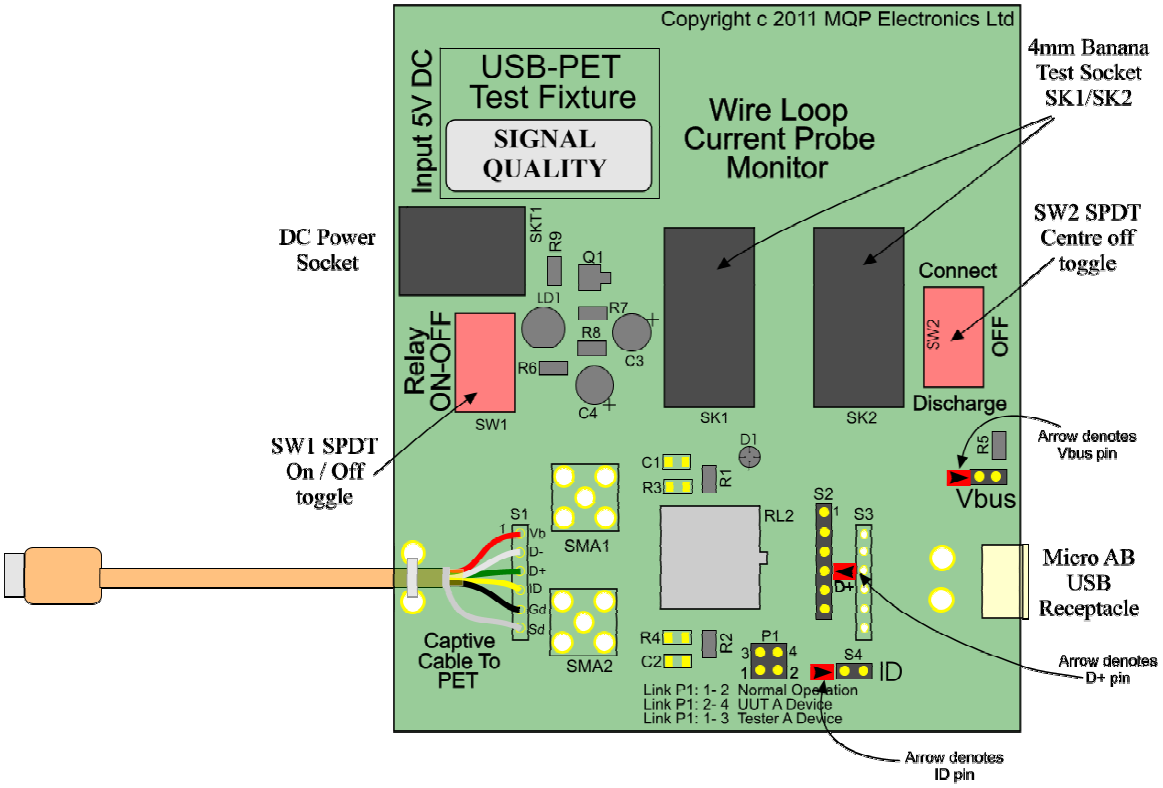
These fixtures are designed to allow the connection of third party test equipment such as oscilloscopes and data generators, to the cable between the USB-PET and the Unit Under Test (UUT). They are particularly designed for OTG 2.0, in that they include a fifth signal for controlling the resistance to ground on the ID pin of the UUT.

Each fixture, except for the Economy Breakout Board, is supplied with a special control cable, enabling the PET to automatically switch it to the terminated state at the appropriate point in the test.



5.1.1 High Speed Signal Quality (Host or Peripheral) plus Inrush Measurement Test Fixture

Reference: EL_2,3,4,5,6, 7, 8 and 9; B.4



USB PET Test Fixture SIGNAL QUALITY

The following table defines the pin numbering on the fixture header.

Table 5-1 Differential Probe Connector S2

Pin	Function	Comment
1	Ground	Alternative ground positions are provided for compatibility with different differential probes. Either ground pin at each end of the connector may be cut short in required.
2	Ground	As above.
3	D-	
4	D+	
5	Ground	As above.
6	Ground	As above.

The following table outlines the procedure for a high speed signal quality test. For more information see the oscilloscope manufacturer's operating instructions, and the USB-IF USB 2.0 Electrical Test Specification documents.

Table 5-2 Procedure for High Speed Signal Quality Test

Step	Action
1	Set the test fixture 'Relay ON/OFF' switch to the 'ON' position.
2	Set the test fixture 'Connect/OFF/Discharge' switch to the 'Connect' position.
3	Ensure that the jumper on P1 is connected to the 'Normal Operation' position.
4	Check that the banana plug wire loop is inserted into the banana test socket, to complete the VBUS circuit.
5	Connect PET to the provided power supply, and to an appropriate host PC with the latest version of the application GraphicUSB installed.
6	Plug test fixture micro-A plug into micro-AB receptacle on PET front panel.
7	Plug the D-connector of the test fixture control cable into the AUX connector on the PET front panel.
8	Plug the other end of the test fixture control cable into the DC power socket of the test fixture.
9	Plug the Special Test Cable (A or B as appropriate) into the test fixture micro-AB receptacle, and plug the other end into the UUT.
10	Attach the differential probe of the oscilloscope to S2 on the test fixture. Ensure D+ on probe lines up with D+ on fixture.
11	From the PET Test Suites dialog, select either the Host Test Packet or the Device Test Packet button as appropriate. The PET will provide the conditions for the UUT to start producing the test packet, and will then automatically switch in the correct test termination on the test fixture.
12	Measure transmitted waveform with high speed oscilloscope and differential probe.
13	Generate eye pattern diagrams from data.
14	Compare with USB 2.0 Specification eye diagrams.
15	Check rise/fall times to make sure they are not faster than minimum set in USB 2.0 Specification.
16	Check for non-monotonic transitions.

The following table outlines the procedure for a high speed J and K Voltage Level test. For more information see the USB-IF USB 2.0 Electrical Test Specification documents.

Table 5-3 Procedure for High Speed J and K Voltage Level Measurement Test

Step	Action
1	Set the test fixture 'Relay ON/OFF' switch to the 'ON' position.
2	Set the test fixture 'Connect/OFF/Discharge' switch to the 'Connect' position.
3	Ensure that the jumper on P1 is connected to the 'Normal Operation' position.
4	Check that the banana plug wire loop is inserted into the banana test socket, to complete the VBUS circuit.
5	Connect PET to the provided power supply, and to an appropriate host PC with the latest version of the application GraphicUSB installed.
6	Plug test fixture micro-A plug into micro-AB receptacle on PET front panel.
7	Plug the D-connector of the test fixture control cable into the AUX connector on the PET front panel.
8	Plug the other end of the test fixture control cable into the DC power socket of the test fixture.
9	Plug the Special Test Cable (A or B as appropriate) into the test fixture micro-AB receptacle, and plug the other end into the UUT.
10	Attach the negative lead of the voltmeter to a suitable ground point on the test fixture.
11	From the PET Test Suites dialog, select either the Host TEST_J or the Device TEST_J button as appropriate. The PET will provide the conditions for the UUT to enter the TEST_J test state and will then automatically switch in the correct test termination on the test fixture.
12	Measure D+ to be 400mV +/- 10%.
13	Measure D- to be 0V +/- 10mV.
14	Click 'OK' on GraphicUSB dialog to end test.
15	Close GraphicUSB Report document (and capture document if generated)
16	Take UUT out of test state by following vendor's instructions for doing so.
17	From the PET Test Suites dialog, select either the Host TEST_K or the Device TEST_K button as appropriate. The PET will provide the conditions for the UUT to enter the TEST_K test state and will then automatically switch in the correct test termination on the test fixture.
18	Measure D+ to be 0V +/- 10mV.
19	Measure D- to be 400mV +/- 10%.

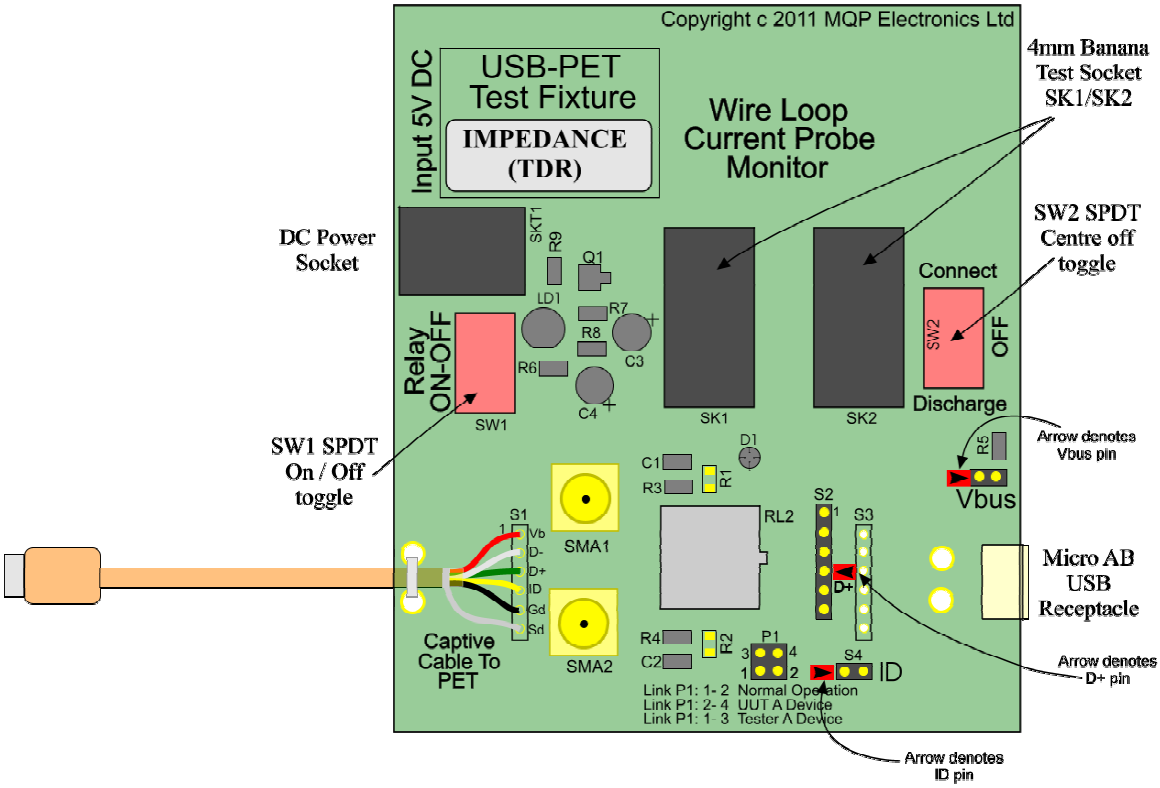
The following table outlines the procedure for an Inrush Current test. For more information see the oscilloscope manufacturer's operating instructions, and the USB-IF Full and Low Speed Compliance Test Procedure rev 1.3 document.

Table 5-4 Procedure for Inrush Current Test

Step	Action
1	Set the test fixture 'Relay ON/OFF' switch to the 'ON' position.
2	Set the test fixture 'Connect/OFF/Discharge' switch to the 'Connect' position.
3	Ensure that the jumper on P1 is connected to the 'Normal Operation' position.
4	Check that the banana plug wire loop is inserted into the banana test socket, to complete the VBUS circuit.
5	Connect PET to the provided power supply, and to an appropriate host PC with the latest version of the application GraphicUSB installed.
6	Plug test fixture micro-A plug into micro-AB receptacle on PET front panel.
7	Plug the Special Test Cable (A or B as appropriate) into the test fixture micro-AB receptacle, and plug the other end into the UUT.
8	Apply the current probe from the oscilloscope to the wire loop provided on the test fixture.
9	From the PET Test Suites dialog, select the Inrush Test button. The PET will turn on VBUS.
10	Follow the instructions in Section B4 of the 'USB-IF Full and Low Speed Compliance Test Procedure rev 1.3.

5.1.2 Termination Impedance Measurement (TDR), J and K Voltage Levels plus Inrush Measurement Test Fixture

Reference: EL_10, 19 and 20; B.4



USB PET Test Fixture IMPEDANCE (TDR)

The following table defines the pin numbering on the fixture header.

Table 5-5 Differential Probe Connector S2

Pin	Function	Comment
1	Ground	Alternative ground positions are provided for compatibility with different differential probes. Either ground pin at each end of the connector may be cut short in required.
2	Ground	As above.
3	D-	
4	D+	
5	Ground	As above.
6	Ground	As above.

The following table outlines the procedure for a high speed termination impedance measurement test. For more information see the TDR Measurement Equipment manufacturer's operating instructions, and the USB-IF USB 2.0 Electrical Test Specification documents.

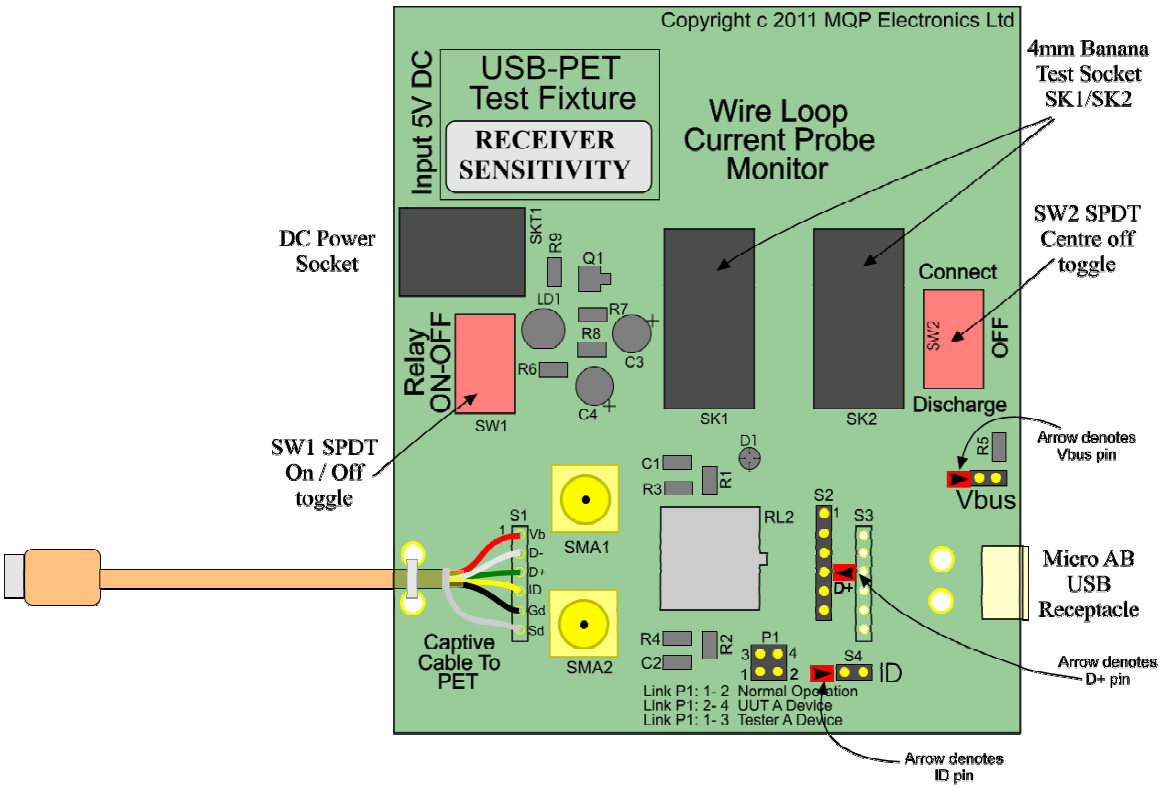
Table 5-6 Procedure for High Speed Termination Impedance Measurement Test

Step	Action
1	Set the test fixture 'Relay ON/OFF' switch to the 'ON' position.
2	Set the test fixture 'Connect/OFF/Discharge' switch to the 'Connect' position.
3	Ensure that the jumper on P1 is connected to the 'Normal Operation' position.
4	Check that the banana plug wire loop is inserted into the banana test socket, to complete the VBUS circuit.
5	Connect PET to the provided power supply, and to an appropriate host PC with the latest version of the application GraphicUSB installed.
6	Plug test fixture micro-A plug into micro-AB receptacle on PET front panel.
7	Plug the D-connector of the test fixture control cable into the AUX connector on the PET front panel.
8	Plug the other end of the test fixture control cable into the DC power socket of the test fixture.
9	Plug the Special Test Cable (A or B as appropriate) into the test fixture micro-AB receptacle, and plug the other end into the UUT.
10	Attach the SMA cables of the TDR Measuring Equipment to SMA1 and SMA2 on the test fixture.
11	From the PET Test Suites dialog, select either the Host SE0_NAK or the Device SE0_NAK button as appropriate. The PET will provide the conditions for the UUT to enter the SE0_NAK test state and will then automatically switch in the correct test termination on the test fixture.
12	Measure D+ and D- to be 0V +/- 10mV.
13	Drive a 400ps (nominal) edge rate step to the device.
14	Observe the resulting waveform, making sure that the termination impedance and through impedance meet the spec requirements.

For the Inrush Current Measurement Test see Table 5-4.

5.1.3 Receiver Sensitivity plus Inrush Measurement Test Fixture

Reference: EL_11, 12, 13, 14, 15, 16, 17 and 18; B.4



USB PET Test Fixture RECEIVER SENSITIVITY

The following table defines the pin numbering on the fixture header.

Table 5-7 Differential Probe Connector S2

Pin	Function	Comment
1	Ground	Alternative ground positions are provided for compatibility with different differential probes. Either ground pin at each end of the connector may be cut short in required.
2	Ground	As above.
3	D-	
4	D+	
5	Ground	As above.
6	Ground	As above.

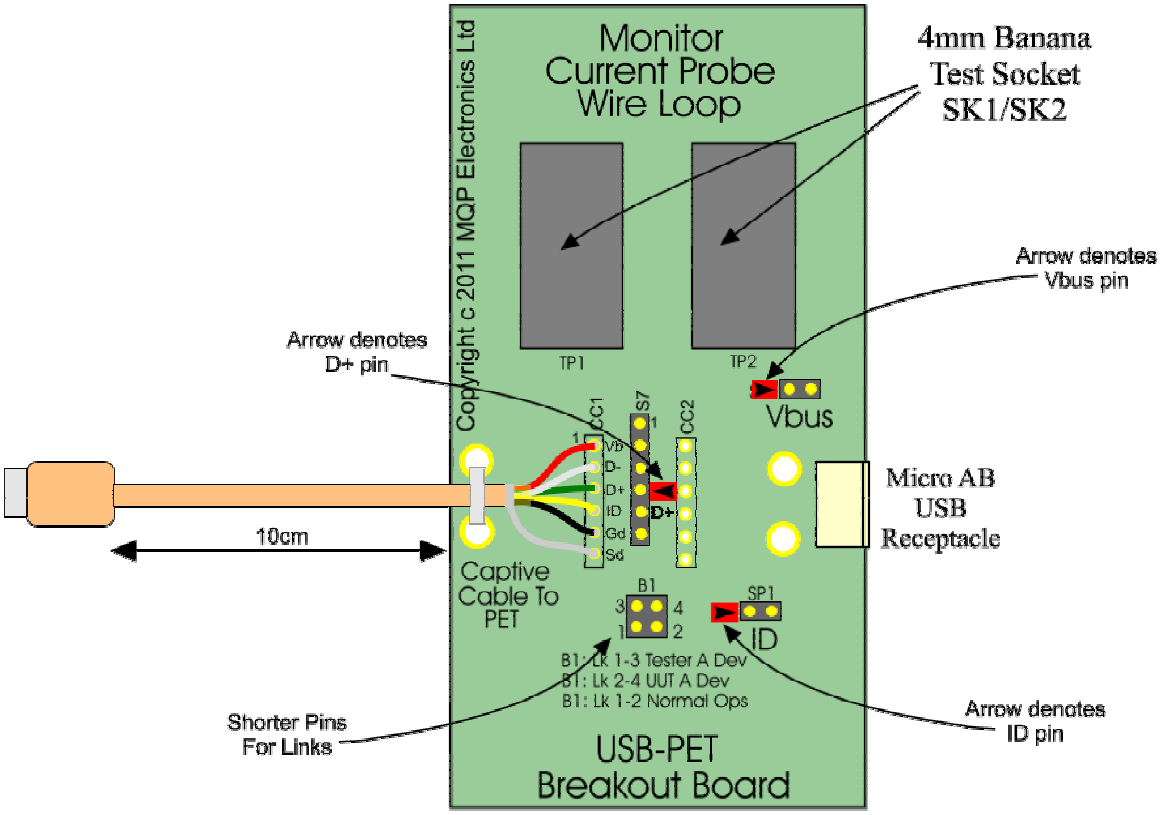
The following table outlines the procedure for a high speed upstream or downstream receiver sensitivity test. For more information see the Data Generator manufacturer's operating instructions, and the USB-IF USB 2.0 Electrical Test Specification documents.

Table 5-8 Procedure for Upstream Port Receiver Sensitivity Test

Step	Action
1	Set the test fixture 'Relay ON/OFF' switch to the 'ON' position.
2	Set the test fixture 'Connect/OFF/Discharge' switch to the 'Connect' position.
3	Ensure that the jumper on P1 is connected to the 'Normal Operation' position.
4	Check that the banana plug wire loop is inserted into the banana test socket, to complete the VBUS circuit.
5	Connect PET to the provided power supply, and to an appropriate host PC with the latest version of the application GraphicUSB installed.
6	Plug test fixture micro-A plug into micro-AB receptacle on PET front panel.
7	Plug the D-connector of the test fixture control cable into the AUX connector on the PET front panel.
8	Plug the other end of the test fixture control cable into the DC power socket of the test fixture.
9	Plug the Special Test Cable (A or B as appropriate) into the test fixture micro-AB receptacle, and plug the other end into the UUT.
10	Attach the SMA cables of the Data Generator to SMA1 and SMA2 on the test fixture.
11	Attach the differential probe of the oscilloscope to S2 on the test fixture. Ensure D+ on probe lines up with D+ on fixture.
12	From the PET Test Suites dialog, select the Host SE0_NAK or Device SE0_NAK button as appropriate. The PET will provide the conditions for the UUT to enter the SE0_NAK test state and will then automatically switch in the correct test termination on the test fixture.
13	Cause Data Generator to generate IN packets of minimum receivable amplitude (must meet appropriate receiver sensitivity template), with common mode voltage components ranging from -50mV to 500mV and with bit rate ranging from 480Mb/s -0.05% to 480Mb/s +0.05%.
14	Verify that all packets are NAKed while signalling is above the required voltage threshold.
15	Verify that no packets are NAKed when signalling amplitude is below the squelch level.
16	Generate IN packets (of compliant amplitude) with a 12-bit sync field.
17	Verify that device responds.

For the Inrush Current Measurement Test see Table 5-4.

5.1.4 Economy Breakout Board plus Inrush Measurement Test Fixture



The following table defines the pin numbering on the fixture header.

Table 5-9 Differential Probe Connector S2

Pin	Function	Comment
1	Ground	Alternative ground positions are provided for compatibility with different differential probes. Either ground pin at each end of the connector may be cut short in required.
2	Ground	As above.
3	D-	
4	D+	
5	Ground	As above.
6	Ground	As above.

This fixture allows third part equipment, in particular oscilloscopes, to be connected to the USB cable signals. It also supports the measurement of VBUS current.

In normal use the Current Probe Wire Loop should be plugged into the banana sockets to complete the VBUS connection. VBUS current can be monitored with a suitable DC current clamp. Alternatively a current meter can be interposed in the VBUS connection by removing the banana plug wire loop and plugging the meter into the 4mm banana sockets.

6 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.

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 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.

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.

7 WARRANTY

7.1 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.

7.2 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.

7.3 Warranty Period

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

7.4 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.

