The Black Box ToolKit

Serious about science: Serious about timing

The Black Box ToolKit

USB TTL Module v1 Guide

Rev. RC2 (20151101) Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

Credits:

Author: Dr. Richard R. Plant, C.Psychol, CSci, AFBPsS

Covers the following hardware:

The Black Box ToolKit USB TTL Module v1

For the following platforms:

Microsoft Windows XP SP3, Vista SP2 (32/64), Windows 7 SP1 (32/64)
Windows 8 (32/64), Windows 8.1 (32/64), Windows 10 (32/64)

Mac OS 8/9, OS-X

Linux 2.4 and greater

Contact details:

The Black Box ToolKit Ltd
PO Box 3802
Sheffield
S25 9AG
UK

Phone: +44 (0)114 303 00 56
Fax: +44 (0)114 303 46 56

Email: info@blackboxtoolkit.com
support@blackboxtoolkit.com
sales@blackboxtoolkit.com

Web address: www.blackboxtoolkit.com

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

Contents

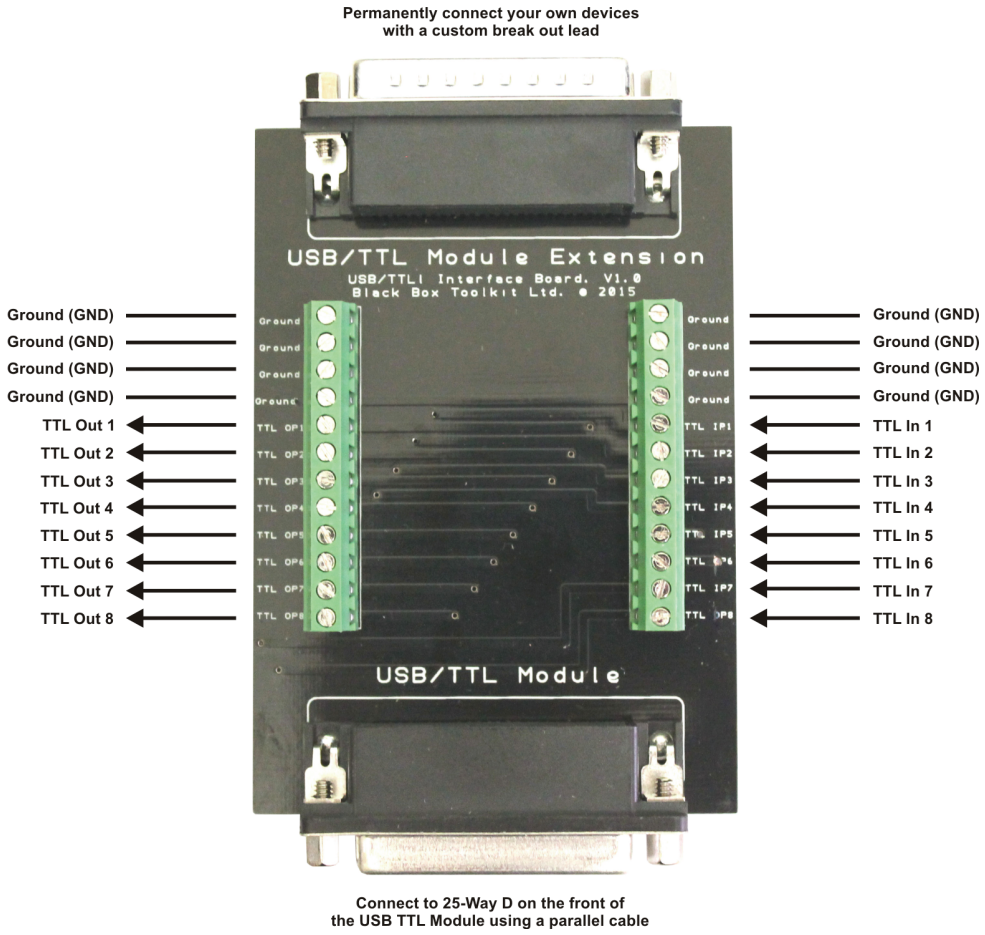
 Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

1. Introduction

The Black Box ToolKit USB TTL Module enables you to quickly and easily TTL event mark from any Experiment Generator or other software that supports a standard serial port under Microsoft Windows. It can be used in any environments where there is a need to event mark, e.g. EEG, eye tracking etc. You should bear in mind that the TTL Module is locked to the time domain of your Experiment Generator, i.e. when you instruct it to produce a TTL signal you are relying on your Experiment Generator to output the serial bytes required at the time you request them. For an example of what this may mean for your experiment you are advised to consult section 5 of this guide.

The Module itself is designed to act as a parallel port replacement for PCs that have only USB ports and has 8 TTL input lines and 8 TTL output lines.

You have the option of using existing parallel cables to interface with your own equipment via the male 25-way D connector. Alternatively you can make use of the optional BBTK USB TTL Module breakout board (shown below) which is connected to the Module by the supplied 25-way parallel cable.



Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

Using the Module to event mark is simple. Once installed all you need do is open the linked serial port and send a two byte hexadecimal string. For example to event mark on TTL Output Line 1, open the serial port and send the hex bytes 01. To stop send 00.

To receive incoming TTL signals sent from your equipment on the TTL Input Lines you would monitor the serial port for two bytes being received, e.g. 01 indicates that your equipment has sent TTL Input Line 1 high (+5V).

Examples of event marking a stimulus onset and duration are shown below for several popular packages. Note that these are simple examples and that you are responsible for proper implementation.

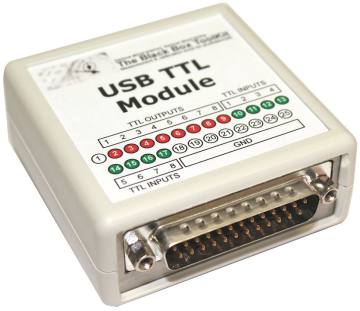
Before each use you should ensure that the serial port driver is installed and configured correctly (see section 2) and that timing validation checks have been carried out (see section 3).

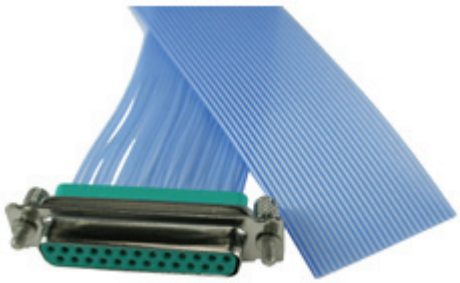
PsychoPy	E-Prime®	MATLAB

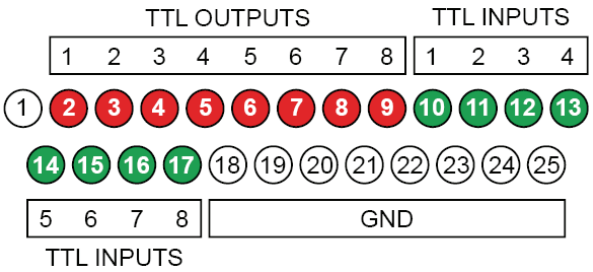
import serial
ser = serial.Serial(0, 115200, timeout=1)
ser.write('01')
<My Stimulus Image Shown Here>
ser.write('00')
ser.close()
	Serial.WriteString "01"
	<My Stimulus Image Shown Here>
	Serial.WriteString "00"
		s = serial('COM1');
		set(s, 'BaudRate', 115200, 'DataBits', 8, 'StopBits', 1, 'Parity', 'none');
		fprintf(s, '01');
		<My Stimulus Image Shown Here>
		fprintf(s, '00');
		fclose(s);

1.1 USB TTL Module Pinouts

If you have chosen not to purchase the optional breakout board you may need to make up your own parallel cable which maps to your specific devices TTL Input and Output pins based on the Modules front facing 25-way D pinouts below.







Pins 2~9 correspond to TTL Output Lines 1~8 and pins 10~17 to TTL Input Lines 1~8.

Pins 18~25 are grounds.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

The rear of the module has a standard USB type B connector by which it is connected to the PC doing the event marking.

Green and Red activity LEDs show Input and Output activity respectively.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

2. Using the USB TTL Module for the First Time

2.1 Installing the Serial Port Driver

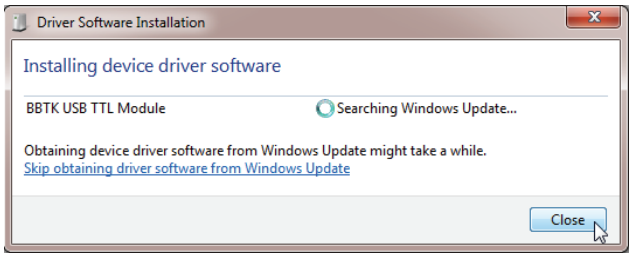
Before you can use the USB TTL Module you must install a simple driver and configure it correctly. This ensures that the Module appears to your system as a standard serial or COM port.

The exact process will vary depending on your Operating System. The example below covers Windows 7 and assumes that Windows Update will automatically supply the required drivers. For Windows 8, 8.1 and 10 the process is virtually identical. For Windows XP and Vista you will find a Drivers folder on the installation CD. A Drivers folder is also installed with the Modules configuration utility.

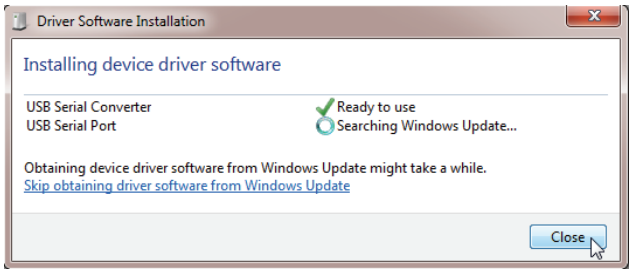
Before installation ensure that you are logged in with local administrator rights.

Using the supplied USB cable plug the USB TTL Module into a primary USB port. That is, a USB port that is not shared with another device or is part of a hub or switch. This should be a full speed USB 2 or 3 port.

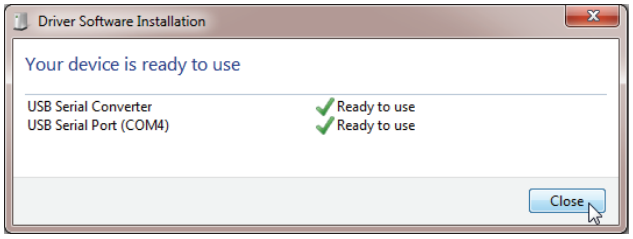
The Green and Red LEDs on the rear of the Module will illuminate for 10 seconds as it initialises each time it is plugged in.



The very first time you plug in the Module it will be identified and drivers should be automatically downloaded for it from Windows Update as shown.



As the drivers are downloaded a tick will appear next to each component.

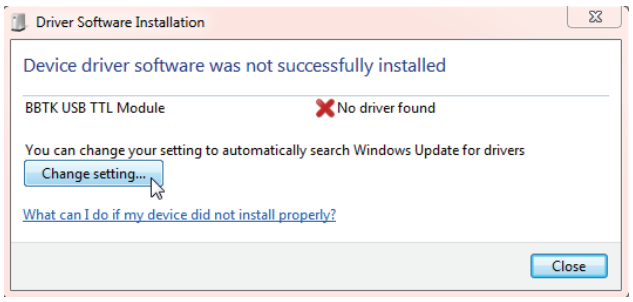


As part of the final step the Module will be assigned a USB Serial Port COM number. In this example, COM4. This is the serial port you would connect to the USB TTL Module through in order to event mark.

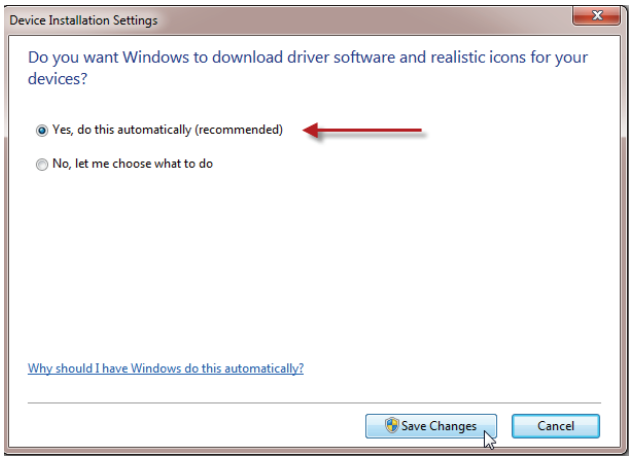
Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

2.2 Troubleshooting Installation of the Serial Port Driver

Depending on how your PC is setup you may need to manually tell Windows to use Windows Update to install drivers for the USB TTL Module.



Do this by clicking on the Change setting… button.



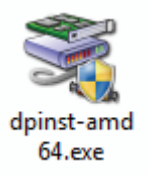
Next click on Yes, do this automatically (recommended). The required drivers will then be automatically downloaded as shown in section 2.1.

You also have the option to click on No, let me choose what to do, and select a one off driver installation from Windows Update.

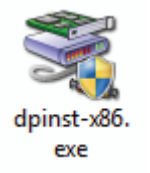
If you are not connected to the Internet you can install the drivers from the installation CD. A Drivers folder is also installed with the Modules configuration utility.

Your Operating System may automatically prompt you to select the folder where the drivers are located in which case you should select the Drivers folder.

Alternatively you can run a standalone driver package from the Drivers folder if your Operating System does not automatically prompt you.



Run the 64 bit version for 64 bit Operating Systems.



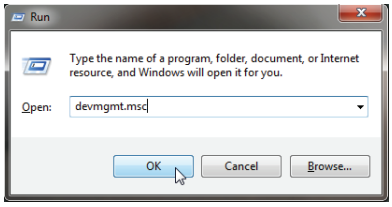
Run the 32 bit version for 32 bit Operating Systems.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

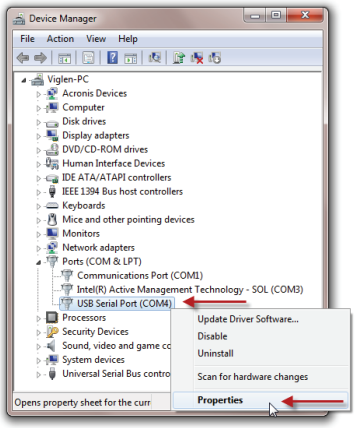
2.3 Configuring the Serial Port Driver

Before you make use of the USB TTL Module for the first time you need to set the latency at which it operates. By default it sends and receives in 16mS busts which is too slow for event marking and needs to be manually set to 1mS.

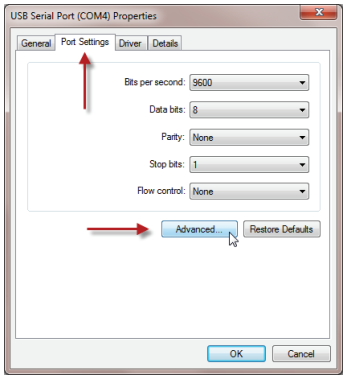
If you have not configured the Serial Port Driver correctly the Black Box ToolKit Ltd cannot be held responsible for inaccurate event marking. Once configured you should install and run our Configuration Utility which validates how quickly you can event mark using your particular PC and the USB TTL Module.



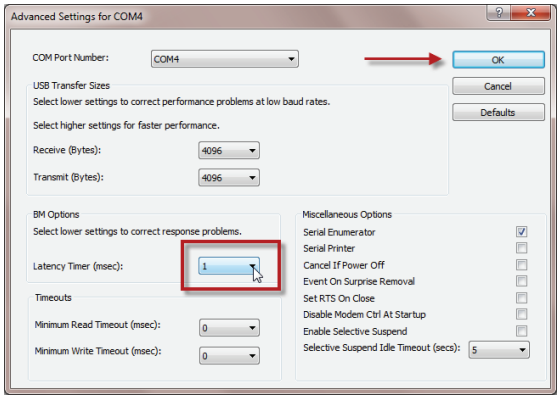
To configure the Serial Port Driver, press the Windows Key+R, and type “devmgmt.msc”. Then click on OK. This will open the Windows Device Manager.



Right click on the USB Serial Port where the BBTK USB TTL Module was installed. Then click on Properties.



Click on the Port Settings tab and then on Advanced…



Under Advanced Setting for COMx you should select a Latency Timer (msec) of 1mS using the drop down box.

To apply the new settings click OK. To close the Properties dialog click on OK again.

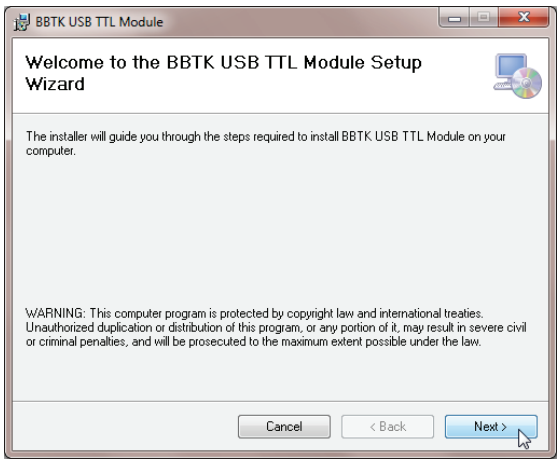
Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

3. The BBTK USB TTL Module Configuration Utility

3.1 Installing the Configuration Utility

Before you carry out any event marking you should validate the timing latency on your particular PC. To do this you will need to make use of the BBTK USB TTL Configuration Utility.

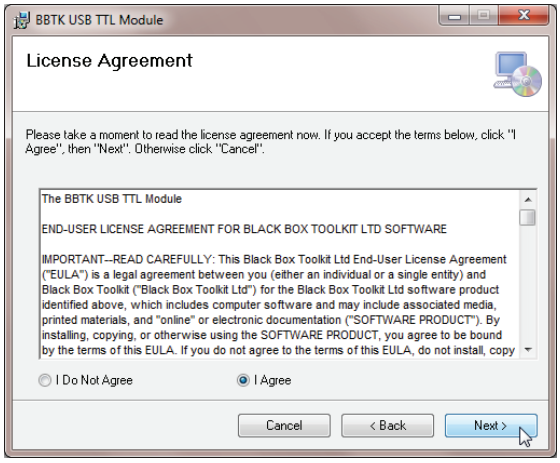
This utility can determine if the Serial Port Drivers have been installed and configured correctly as shown in section 2. Crucially it carries out internal tests which determine the timing latency of your PC and Operating System when using the USB TTL Module.



Ensure you are logged into the local PC with Administrator rights and insert the CD into your CD drive. The setup should automatically run. If not you may need to manually run setup.exe.

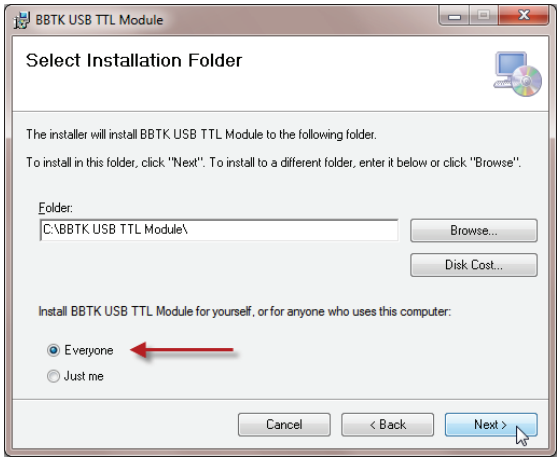
If the Microsoft .NET 4 Framework is not installed on your PC you may be prompted to download and install it.

An offline version is included on the CD under the DotNetFX40 folder. If you wish to use this press Cancel, install the Framework and the rerun the BBTK USB TTL Module setup and continue as below.



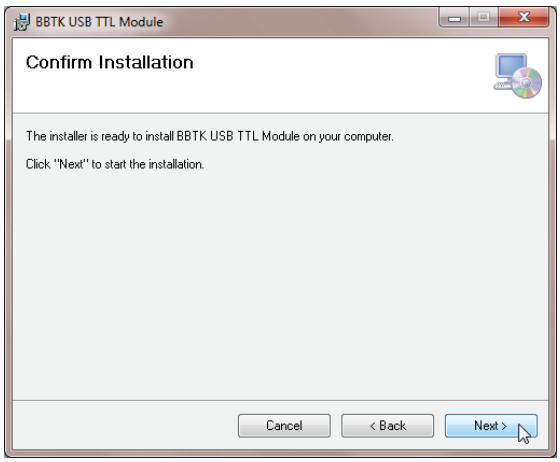
Before you can install you will need to Agree to the License Agreement.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.



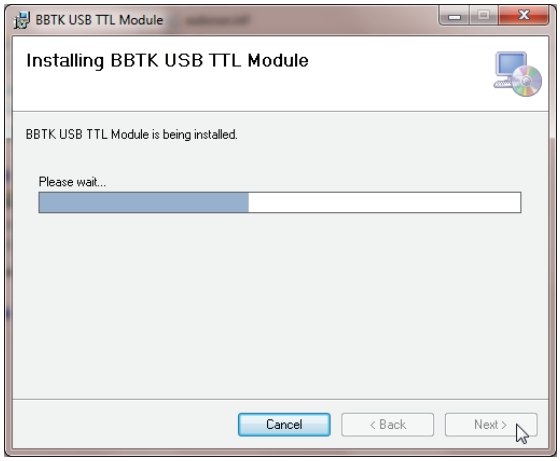
Select the folder where you want to install the BBTK USB TLL Module.

In general you should select Everyone who uses this PC so that everyone who has an account on the PC can use the Configuration Utility.

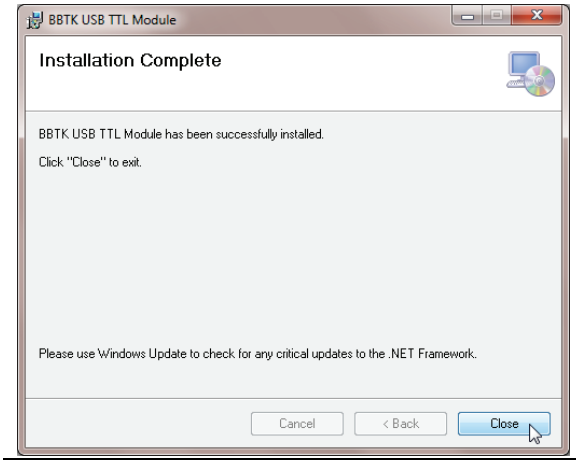


Next confirm installation.

At this stage you may be prompted by UAC to confirm the application Publisher. Click on Yes to continue.

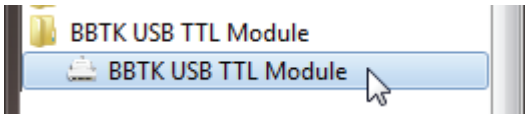


The BBTK USB TTL Module Configuration Utility will now be installed.



To finalise the installation click on close.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.



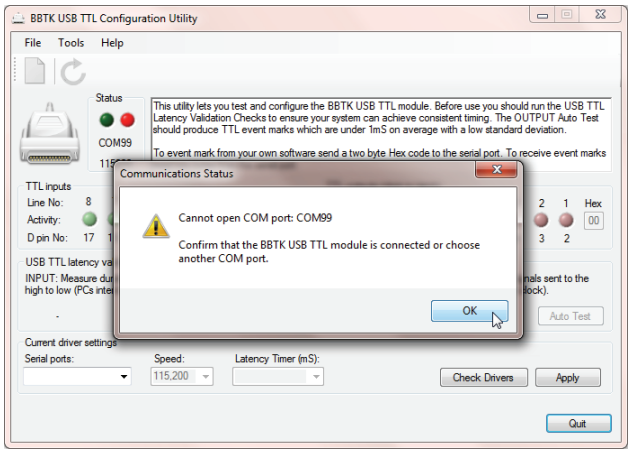
To start the BBTK USB TTL Module Configuration Utility click on its icon from the Start Menu.

3.2 Validating Event Marking Latency

Before using the USB TTL Module for event marking you should use the Configuration Utility to carry out USB TTL Latency Validation Checks. These checks help determine how quickly the Module can event mark from your particular PC over USB.

If you have not configured the Serial Port Driver correctly or if you have not run USB TTL Latency Validation Checks The Black Box ToolKit Ltd cannot be held responsible for inaccurate event marking.

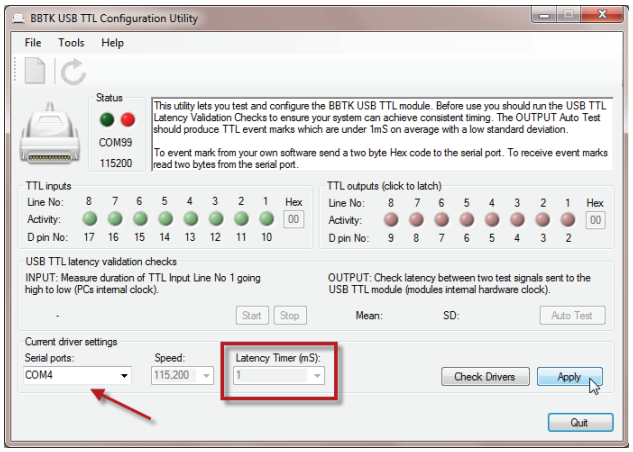
The OUTPUT Auto Test determines the potential latency of your event marks for the PC you are hosting the USB TTL Module on. This sends a series of event mark pairs to the Module with the Modules hardware timing recording the latency between each one in microseconds (uS). This hardware based elapsed time is returned to the Configuration Utility after each pair. In total 50 event marks are sent to the Module in sequence as rapidly as possible.



To begin start the Configuration Utility.

When first run the COM port will be set to COM99 and you will need to choose the correct serial COM port on your system.

To choose the COM port click OK.

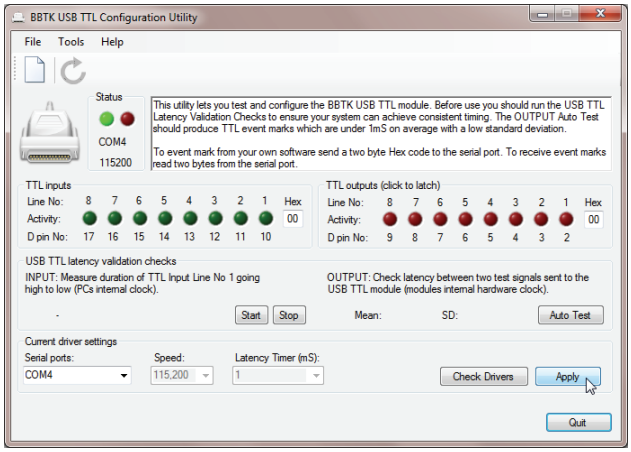


Under Current driver settings the COM port with the USB TTL Module attached should be automatically selected (red arrow).

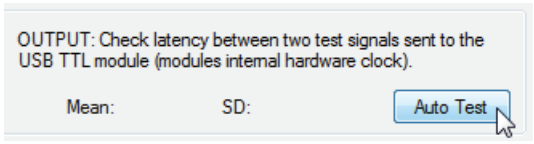
Note that the Latency Timer (mS) should be set to 1mS as detailed in section 2.3. If it set to any other number you should carry out the procedure to set it to 1mS.

If the correct COM port is not selected pick it from the drop down box and then click Apply.

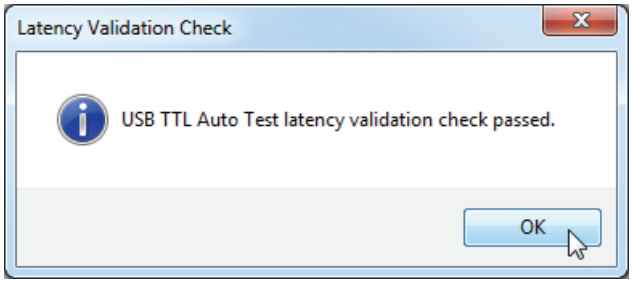
Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.



If you have chosen the correct COM port and the USB TTL Module is attached then the Status LED will change from red to green as shown and the interface will be unlocked.

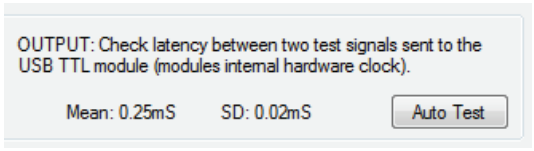


To carry out the latency check click on Auto Test.



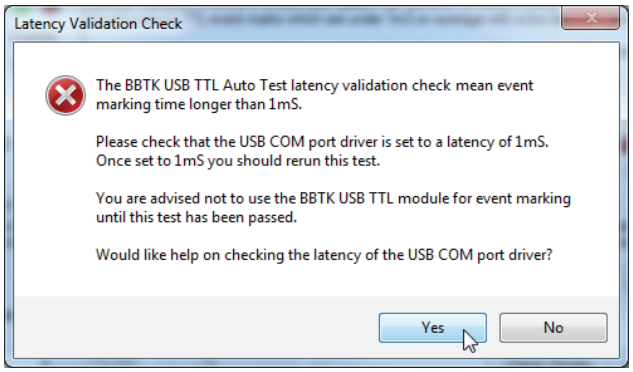
The test will be flagged as being passed if the mean latency is consistently under 1mS.

If the test is not passed on your PC you are advised not to use the USB TTL Module for live event marking as timing consistency cannot be reliably attained.



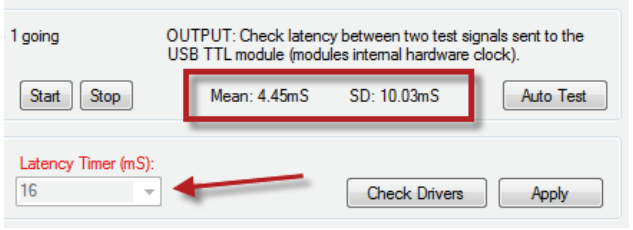
In this case the mean latency to event mark is:

M = 0.25mS, SD = 0.02mS



If the mean event marking time is longer than 1mS you will be warned and advised to check the COM Port Driver latency and offered help on how to do this.

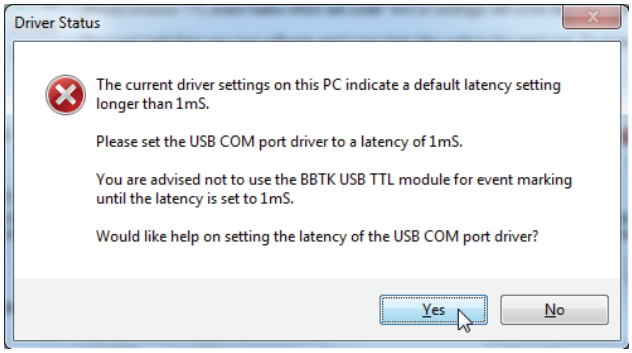
An example of a test failure is shown below.



In this example the test has been failed as the Latency Timer (mS) has been left set at the default driver installation value of 16mS.

The Latency Timer drop down label has also turned red to indicate a value greater than 1mS.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.



If the COM Port Driver is set to a value greater than 1mS a warning dialog will be displayed.

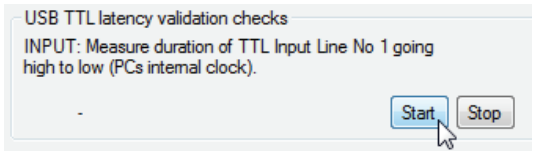
If you have not configured the Serial Port Driver correctly or if you have not run USB TTL Latency Validation Checks The Black Box ToolKit Ltd cannot be held responsible for inaccurate event marking.

Should the Serial Port Driver latency be set to 1mS and your PC still doesn't pass the latency Auto Test you are advised not to use the Module for time critical event marking.

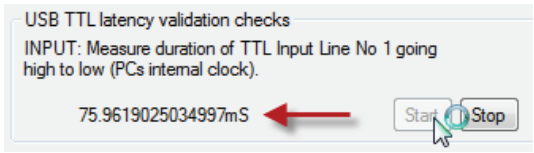
3.3 Checking Input Latency

The Configuration Utility also allows you to check input duration timing. This test measures the time between TTL signals on TTL Input Line 1 going high to low, e.g. a EG pulse from a Black Box ToolKit v2, button press, on|off, +5V to 0V.

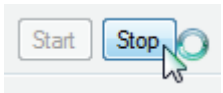
You should note that this Configuration Utility uses Windows based timing for this test in addition to displaying timing measures live therefore its accuracy is subject to some variability as a result.



To start the input duration test click on Start.



As each input is received its duration is displayed.



To stop testing input durations click on Stop.

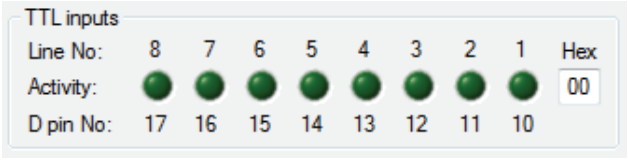
Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

3.4 Checking TTL Inputs Coming From Connected Equipment

Often it's useful to check that your equipment is connected to the USB TTL Module correctly and is outputting TTL signals correctly, e.g. start sequence signals. Remember TTL Inputs on the Module are +5V TTL outputs **FROM** your equipment.

For you to be able to check TTL Inputs the green status LED should be lit bright green and the 8 green TTL Input LEDs should not be greyed out.

The Configuration Utility is simply designed to show if the TTL Input Lines are connected correctly. The green Input LEDs may not be a real-time representation.



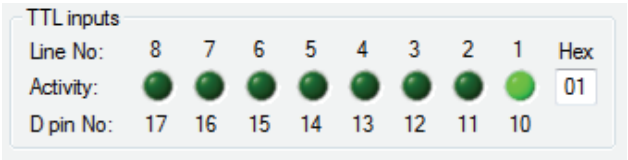
By default the 8 green TTL Input Line LEDs are off as long 0V is being input.

Lines are displayed in binary order from right to left, i.e.

128, 64, 32, 16, 8, 4, 2, 1

Each line corresponds to the D pin numbers shown, e.g. Input Line 1 corresponds to D pin number 10 and so on.

The hexadecimal value of the Input port is also shown in the Hex box. This shows the value of the two bytes been sent from the Module itself. It is these two bytes you should monitor in your own software when detecting TTL Inputs being fed into the Module.



In this example Input Line 1 has gone high with a hex Input Port value of 1 and the appropriate green LED has illuminated (D pin 10). In addition the green input LED on the Module itself will also illuminate.

Hex two byte values can range from 00~FF (hex), 0~255 (dec), 00000000~11111111 (bin) to represent the status of each Input Line on the Input Port.

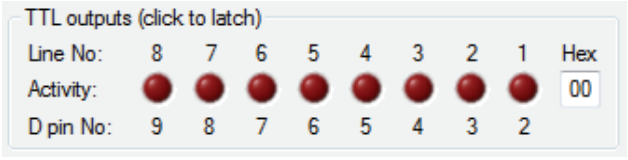
Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

3.5 Checking TTL Outputs Going To Connected Equipment

Often it's useful to check that your equipment is connected to the USB TTL Module correctly and is receiving TTL signals correctly, e.g. event marks. Remember TTL Outputs on the Module are +5V TTL outputs **TO** your equipment.

For you to be able to check TTL Outputs the green status LED should be lit bright green and the 8 red TTL Output LEDs should not be greyed out.

The Configuration Utility is simply designed to show if the TTL Output Lines are connected correctly. The red Output LEDs may not be a real-time representation.



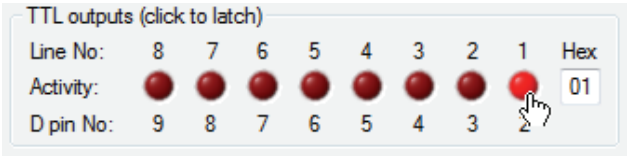
By default the 8 red TTL Output Line LEDs are off as long 0V is being output.

Lines are displayed in binary order from right to left, i.e.

128, 64, 32, 16, 8, 4, 2, 1

Each line corresponds to the D pin numbers shown, e.g. Output Line 1 corresponds to D pin number 2 and so on.

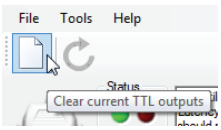
The hexadecimal value of the Output port is also shown in the Hex box. This shows the value of the two bytes being sent to the Module. It is this two byte value you should send to the Module from your own software when you want to event mark on those lines.



To send a given TTL Output Line high (+5V) click on it. To send it low (0V) click on it again so that the LED is not illuminated.

In this example Output Line 1 has been sent high with a hex Output Port value of 01 and the appropriate red LED has illuminated (D pin 2). In addition the red output LED on the Module itself will also illuminate.

Two byte hex Output values can range from 00~FF (hex), 0~255 (dec), 00000000~11111111 (bin) to represent the status of each Output Line on the Output Port.

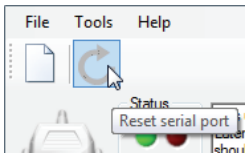


To clear all lines at once (0V) click on the New toolbar button or press CTRL+N.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

3.6 Resetting the USB TTL Module

Sometimes it's useful to be able to reset the Module without unplugging the USB lead.



To reset the Module and Serial Port click on the Reset toolbar icon or press CTRL+R.

Doing this is equivalent to sending a Break in standard serial communications.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

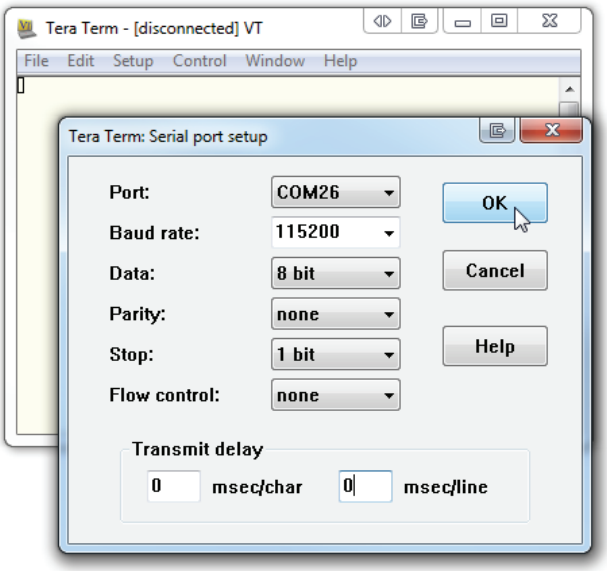
4. Using the USB TTL Module Programmatically From An Experiment Generator Or Your Own Software

4.1 Basic Output Event Marking Using a Serial Terminal

Using the USB TTL Module from a serial terminal provides an opportunity for you to get a feel for how you read inputs and output event marks.

We recommend the free Tera Term terminal software for this purpose however any terminal software that supports a serial port should work.

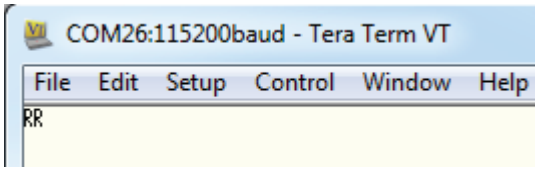
Before you attempt to connect to the USB TTL Module ensure that all other software using the serial port it is assigned to be closed as only one application can use that port at a time. This also includes the configuration utility.



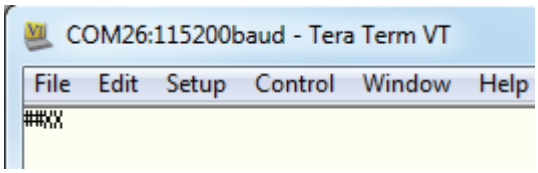
Before you can use the USB TTL Module you will need to connect to it using the COM port which you installed it on. In this example COM26.

You will need to configure the terminal software to use the communication settings shown:

Baud rate:	115,200
Data:	8 bit
Parity:	None
Stop:	1 bit
Flow control:	None



Once connected you will need to initialise it by sending the Module two RR bytes which resets the serial port and clears all TTL Output Lines.

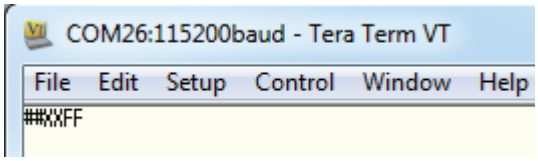


To check that the Module has been initialised and reset correctly send type ## bytes. The Module should reply with XX as shown.

You'll note that as the Module responds to two bytes as you press the second # it responds. You do not need to press return.

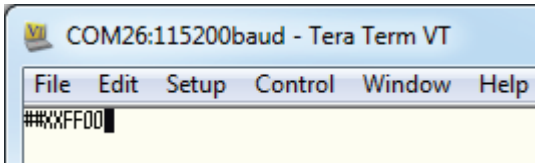
The Module should only need to be initialised when it is first connected by sending it RR. However you can also use RR when ending an experiment to clear all lines and reset the Module.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.



Once initialised to turn on all TTL Output Lines, i.e. 1~8 type the hex value FF as shown.

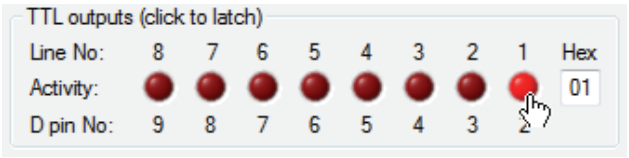
The red LED on the Module will then illuminate and any equipment connected will show a TTL +5V signal on all lines.



Note that the Output Lines stay on, or latch. To turn them off you need to send another hex value to the Module, e.g. 00. Again this has to be a hex byte pair.

It's crucial to ensure that you send hex pairs of byes, or characters, as otherwise the Module may get out of sync and will only respond when a second byte is sent. All hex byte pairs should be sent in capitals.

To work out which lines you wish to turn on you can use the Configuration Utility to construct a hex value.

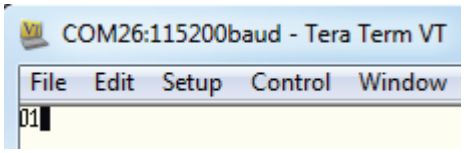


Two byte hex Output values can range from 00~FF (hex), 0~255 (dec), 00000000~11111111 (bin) to represent the status of each Output Line on the Output Port.

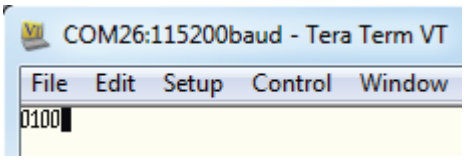
To convert from a binary on|off bit pattern simply convert to hex using a calculator.

4.2 Basic TTL Input Using a Serial Terminal

Receiving TTL input from the USB TTL Module is straightforward. You should bear in mind that your own software will need to constantly monitor the serial port for input as this will happen asynchronously. That is, it can happen at any time. If you are simply event marking using the Module, i.e. outputting from your experiment generator or own software you do not need to monitor for inputs at all.



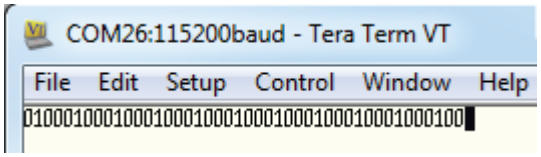
Once connected and initialised (see section 4.1) the Module is ready to send you TTL Input Port Values encoded as two byte hex strings. Here TTL Input Line 1 has been sent high (+5V).



Unless there is a line change no other byte pairs will be sent from the Module. When there is a change on the Input Port a new hex pair will be sent.

In this example 00 indicates that all Input Ports lines have all gone to 0V or no inputs.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.



If there is a pulse on a given Input Line then a series of hex byte pairs will be sent. In this example Input Line 1 is going from 1 to 0 or +5v to 0V repeatedly.

4.3 Tips for Using the USB TTL Module With an Experiment Generator or Your Own Software

Using the USB TTL Module should be fairly straight forward for those with a background in programming.

The information below summarises a series of tips for using the USB TTL Module effectively for simple event marking.

WARNING: The Black Box ToolKit Ltd cannot be held liable for any damage caused to devices through use of the USB TTL Module howsoever caused. By using the USB TTL Module you accept sole responsibility for any subsequent damage. The end user should evaluate fitness for purpose prior to using the USB TTL Module with any given device. The Module is designed for use in research environments.

Physical connection:

- Use a USB port which is directly connected to the motherboard
- Try and always use the same USB port
- Do not use USB hubs or switches to connect the Module
- Do not use high bandwidth USB devices together with the Module and do not leave them connected to any USB ports on your system, e.g. USB memory sticks, external hard drives etc.
- Use the supplied high quality USB cable

Before use you should to ensure that:

- The Serial Port Driver has been installed (section 2.1) and you know what COM port the Module has been assigned to
- The Serial Port Driver has been configured and the latency set to 1mS (section 2.3)
- You have validated Event Marking Latency using the Configuration Utility (section 3.2)

When using you should bear in mind that:

- The Module should be opened with the following parameters: 115200, 8, N, 1

Baud rate:	115,200
Data:	8 bit
Parity:	None
Stop:	1 bit
Flow control:	None

- Only one application can access one serial port at a time (this includes the Configuration Utility). If two are open you will receive an error (section 4.1)

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

- Once connected you will need to initialise it by sending the Module RR. Ideally you should do this at the start of each experiment and optionally at the end of each experiment to ensure that all TTL Output Lines are cleared on exit
- For event marking send two hex bytes in capitals, e.g. FF to turn all output lines on (section 4.1)
- Output Lines stay on or latch when activated. To turn them off you need to send another hex value to the Module, e.g. 00. Again this has to be a hex byte pair (section 4.1)
- Only TTL Input Line changes are reported in order to reduce latency to below 1mS (section 4.2)
- It is possible to use the Module asynchronously for both input and output. However any two byte output must be completed before input can resume, i.e. outputs are blocking
- If you are monitoring the serial port for input, i.e. TTL signals being fed INTO the Module you will need to do so asynchronously as two byte reports can come at any time (section 4.2)
- Ensure that you always send and receive two hex bytes otherwise this may cause the Module to hang until you send or receive a second byte
- Once the COM port is opened you should not open and close it again. You should only open it once and close it once in the course of an event marking session regardless of how many event marking hex byte pairs you send
- To retrieve the firmware version number send the byte pair VE to the module. Firmware updates may be available from the USB TTL Module area of our support website

Examples of event marking a stimulus onset and duration are shown below for several popular packages. Note that these are simple examples and that you are responsible for proper implementation as we cannot provide detailed programming advice for third party systems.

PsychoPy	E-Prime®	MATLAB

import serial
ser = serial.Serial(0, 115200, timeout=1)
ser.write('01')
<My Stimulus Image Shown Here>
ser.write('00')
ser.close()
	Serial.WriteString “01”
	<My Stimulus Image Shown Here>
	Serial.WriteString “00”
		s = serial('COM1');
		set(s, 'BaudRate', 115200, 'DataBits', 8, 'StopBits', 1, 'Parity', 'none')
		fprintf(s, '01');
		<My Stimulus Image Shown Here>
		fprintf(s, '00');
		fclose(s)

You are also advised to consult the appropriate documentation for writing byte pairs in hex, e.g. WriteByte(0x01) in MATLAB.

When developing it is advisable to check that the Module is working correctly in a serial terminal such as Tera Term (section 4).

You should bear in mind that the TTL Module is locked to the time domain of your Experiment Generator, i.e. when you instruct it to produce a TTL signal you are relying on your Experiment Generator to output the serial bytes required at the time you request them. For an example of what this may mean for your experiment you are advised to consult section 5 of this guide.

 Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

5. Worked Examples Using E-Prime®

5.1 Event Marking Using Task Events

E-Prime® 2.0 and above allows you to easily event mark using Task Events through E-Primes standard property dialogs. Psychology Software Tools describes Task Events as follows:

> *"Task Events improve upon E-Prime 2.0 Professional's ability to communicate with external devices by offering a variety of trigger signals to be sent when specific time-critical events occur during an experiment. Previous versions of E-Prime provided some limited support for this type of communication, but only by way of complex design and/or advanced E-Basic scripting techniques. These capabilities can now be achieved using the E-Studio graphical user interface, with greater precision and without the need to write InLine script."*

You should bear in mind that the TTL module is time-locked, or synchronous, to the E-Prime® time domain, i.e. when you instruct it to produce a TTL signal you are relying on E-Prime® to output the serial bytes required at the time you request them. This also applies to any other Experiment Generator or software that you have written yourself. The TTL Module is simply designed to offer a method to produce and receive TTL signals. The TTL Module has no in-built timing mechanisms and it should be thought of as an alternative to using a parallel port.

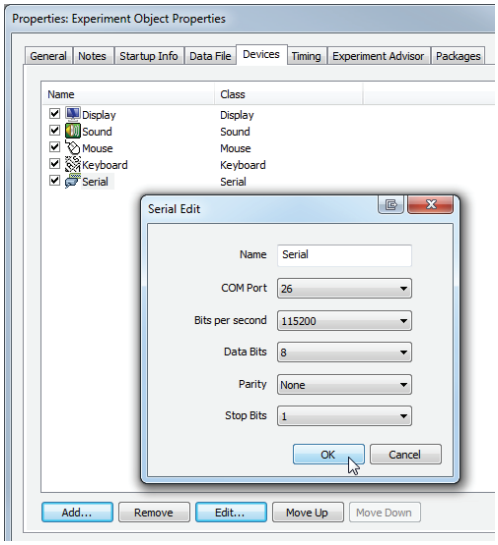
In practical terms this means that you are reliant on your software producing an event mark at the correct time just as you would be if you were using a parallel port. For example, if you choose to event mark an image onset you are reliant on your software sending the required serial bytes to the TTL Module immediately prior to displaying the image.

If you wish to base your event marking on the physical appearance of stimuli in the real world rather than the time you requested them be shown you will need to make use of timing validation hardware such as the Black Box ToolKit v2 or the mBBTK v1 which is specifically designed for event marking physical stimuli across multiple modalities and TTL I/O channels.

The Black Box ToolKit Ltd cannot accept any responsibility for inaccurate event marking that are as a result of bugs in your chosen software, poor scripting practices or Operating System issues.

THE FOLLOWING EXAMPLE SCRIPT IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE EXAMPLE SCRIPT OR THE USE OR OTHER DEALINGS IN THE EXAMPLE SCRIPT.

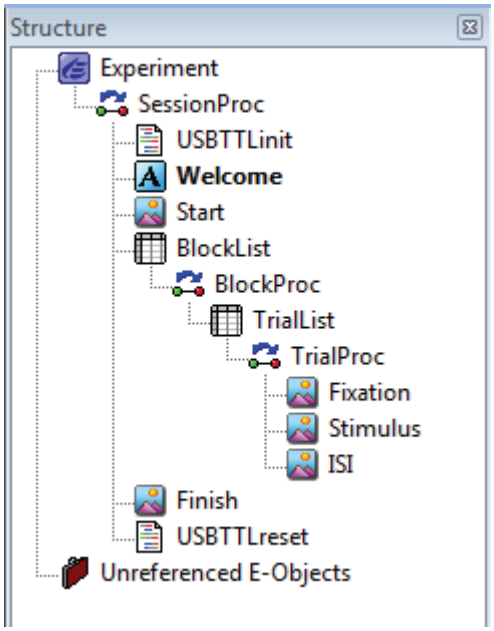
Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.



Before you can begin you will need to ensure you have added your USB TTL Modules serial port to the list of available devices. For consistency we have called the device Serial.

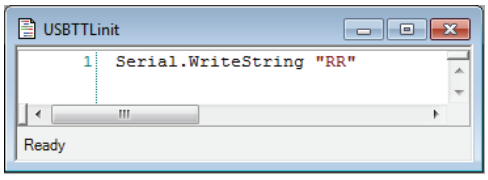
You will need to ensure that the selected COM port matches that of the USB TTL Module on your own system.

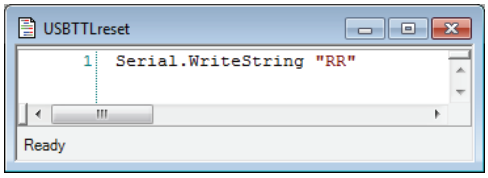
CTRL+E will bring up the Experiment Object Properties Dialog.



In this examples structure view we can see a simple experiment that displays a stimulus image a number of times.

Each stimulus is proceeded by a fixation and followed by an ISI image.





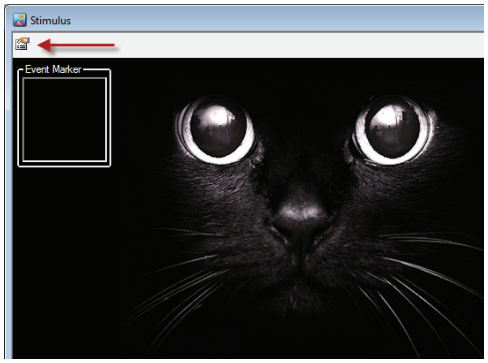
You should note at the top and the tail of the experiment there are InLine scripts to initialise and reset the TTL Module.

Both pieces of script reset the TTL Module and clear any TTL inputs or outputs.

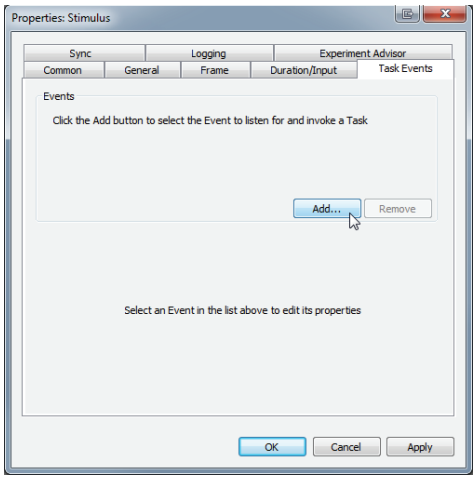
The TTL Module is reset by sending the two bytes RR:

```
Serial.WriteString "RR"
```

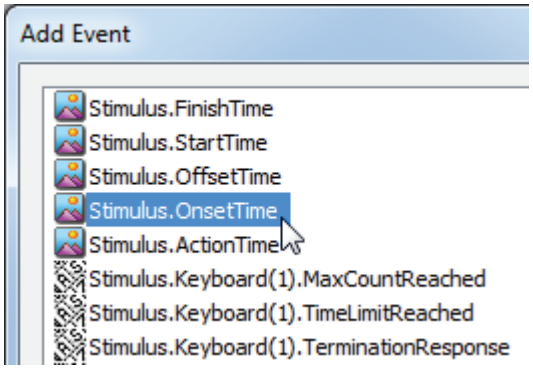

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.



To use a Task Event to event mark click on the stimulus images properties button.

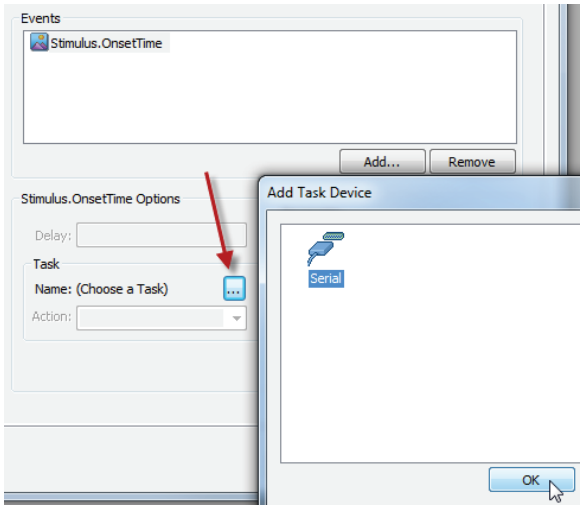


Click on the Task Events tab and then on the Add… button to add a new event.



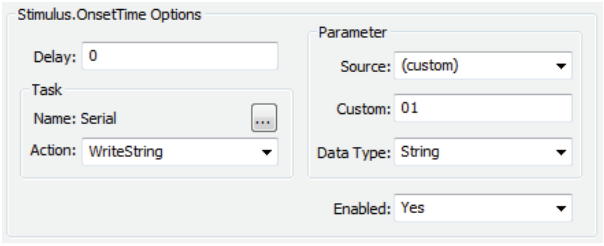
Typically you would select the OnsetTime option for the Stimulus image object.

For further information on the available events you should consult Psychology Software Tools website.



Next click on the more button … and select the Serial port object so that you can write to the USB TTL Module.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

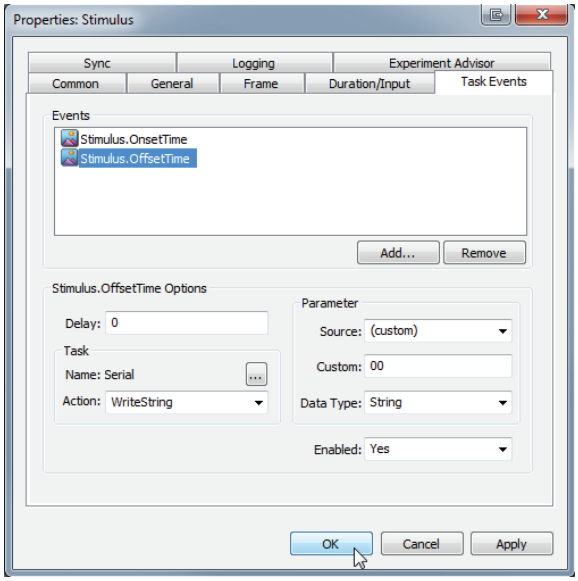


Finally fill in the Stimulus.OnsetTime Options as shown.

In this example:

Custom: 01

means that TTL Output Line will be high (+5V), or on, while the stimulus is displayed.



To turn off the event marking signal you should add a second Task Event as shown but under custom set it to 00:

Custom: 00

This will clear all the TTL Output event marking lines at the stimulus images offset.

When your experiment runs there will now be an event mark on TTL Out 1 whenever the stimulus image is requested to be shown by E-Prime®.

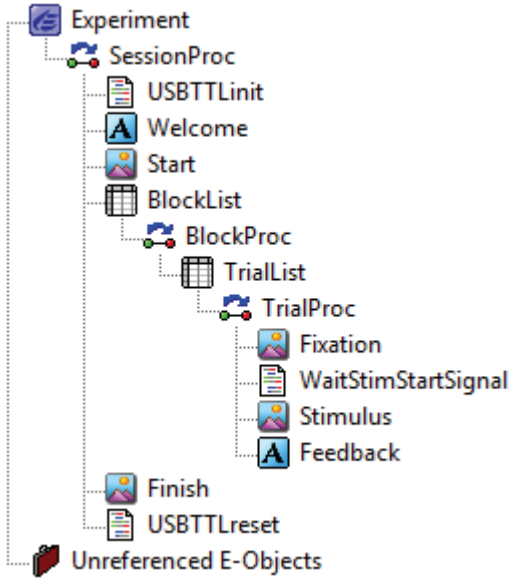
For more details on Task Events and event marking using InLine script calls you should consult the E-Prime® User Guide and online Knowledge Base, e.g. http://www.pstnet.com/support/kb.asp?TopicID=1318.

Please note that the Black Box ToolKit Ltd cannot provide support for third party software such as E-Prime. Example scripts can be downloaded from our support website but are provided on an "as is" basis.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

5.2 Input Triggering Using Your Own Scripts

The TTL Module allows you to wait for an incoming TTL input trigger and then perform an action. Typically this feature is used to show a stimulus when a trigger is received, e.g. in fMRI studies.

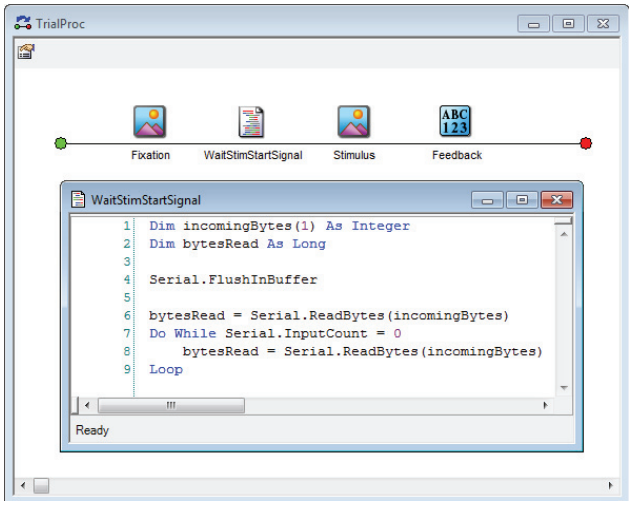


In this example as before there are two pieces of InLineScript that reset the USB TTL Module and are called at the start and end of the experiment.

Note that just after the fixation object there is a piece of InLine script that we have called:

WaitStimStartSignal

Experiment execution will wait here until there is a TTL input on the USB TTL module.



In this example execution will pause at the InLine code as the Do While Loop waits for any input change signalled by two incoming serial bytes from the USB TTL Module.

So for example if TTL In 1 went high (+5V) then bytes 01 would be sent via serial to E-Prime, the holding loop exited and the stimulus shown.

```
Dim incomingBytes(1) As Integer
Dim bytesRead As Long

Serial.FlushInBuffer

bytesRead =
Serial.ReadBytes(incomingBytes)
Do While Serial.InputCount = 0

      'look for incoming bytes on the
serial port that indicates a trigger
signal
      bytesRead =
Serial.ReadBytes(incomingBytes)
Loop
```

A typical example of a holding loop is shown opposite.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

For more details on using InLine script calls you should consult the E-Prime® User Guide and online Knowledge Base, e.g. http://www.pstnet.com/support/kb.asp?TopicID=1318.

Please note that the Black Box ToolKit Ltd cannot provide support for third party software such as E-Prime. Example scripts can be downloaded from our support website but are provided on an "as is" basis.

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

6. Technical Specifications

6.1 Hardware Specifications

- Dimensions (W x H x D): 67.1 x 28.2 x 67.1 mm
- Weight: approx 100 g
- Shipping carton dimensions (W x H x D): 100 x 60 x 80 mm
- Shipping carton weight: approx 200 g
- 28-Pin, High-Performance, Enhanced Flash, USB 2.0 Microcontroller with nanoWatt Technology
- Single chip USB to asynchronous serial data transfer interface
- USB 2.0 Full Speed compatible
- Low USB bandwidth consumption
- Operating Temperature: 0 °C to 50 °C
- Operating Voltage Range: 3.3 V to 5.25 V
- High-Current Sink/Source: 25 mA/25 mA
- 100,000 Erase/Write Cycle Enhanced Flash Program Memory Typical for firmware updates
- Flash/Data EEPROM Retention: > 40 Years

- 16 Digital +5 V TTL Lines
    - 8 TTL Input
    - 8 TTL Output
- Accessed via standard Virtual COM Port (VCP)
- TTL Input Lines configured as an 8 bit port
- TTL Output Lines configured as an 8 bit port
- Change detection on TTL Input Lines
- TTL Input to 2 hex bytes conversion representing 255 possible states
- 2 hex bytes to TTL output across 8 bit port representing 255 possible states

- Custom designed firmware
    - Checks for a TTL event mark or trigger event 109,000 times a second (109 kHz sampling rate)
    - Typical time to send a TTL output or event mark from a two byte hex code from the PC, 254 µs#
    - Typical time to respond to a TTL input/trigger pulse or sync signal and send a two byte hex code to the PC, 350 µs#
    - Typical time for 50 TTL outputs or event marking pairs (on|off), M = 290 µs, SD = 10 µs, measured using our configuration and latency validation software for Microsoft Windows (Win 7 64 SP1, USB 2) and modules internal hardware timer#
    - Supplied with configuration and latency validation software for Windows XP, Vista, Win 7, Win 8/8.1 and Win 10
    - Built-in hardware timing checks based on elapsed time between two sequential event marks

#Dependent on specific USB subsystem and using the USB TTL Configuration utility to check correct for correct installation.

- Moulded grey ABS plastic enclosure
- Stainless Steel front and rear panels with DP1 finish

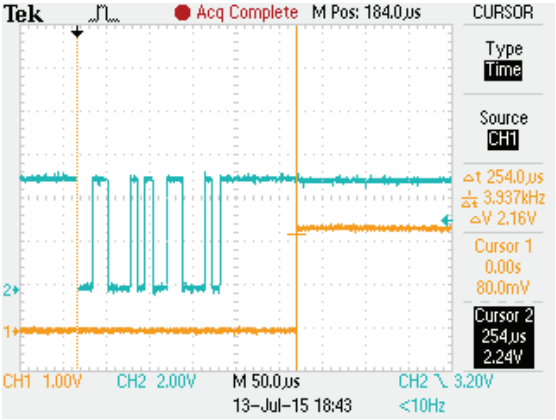
- Custom designed PCB
    - Material: FR4
    - Solder Resist Coating: CAWN_1331 green resist and CAWN_1291 green hardener
- Lead free solder: MULTICORE / LOCTITE - 96SC 400 5C 1.00MM - SOLDER WIRE (Henkel 60EN 362 5C)

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.

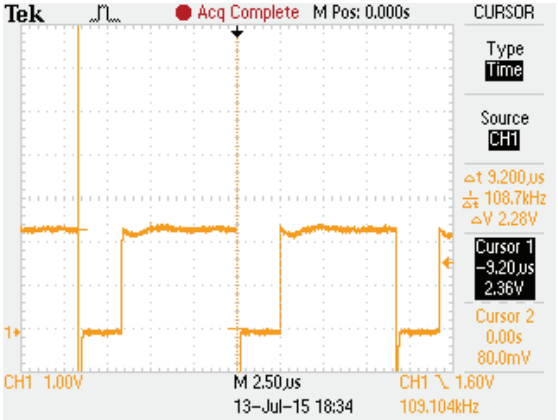
6.2 Timing Specifications

- Checks for a TTL event mark or trigger event 109,000 times a second (109 kHz sampling rate)
- Typical time to send a TTL output or event mark from a two byte hex code from the PC, 254 µs#
- Typical time to respond to a TTL input/trigger pulse or sync signal and send a two byte hex code to the PC, 350 µs#
- Typical time for 50 TTL outputs or event marking pairs (on|off), M = 290 µs, SD = 10 µs, measured using our configuration and latency validation software for Microsoft Windows (Win 7 64 SP1, USB 2) and modules internal hardware timer#
- Supplied with configuration and latency validation software for Windows XP, Vista, Win 7, Win 8/8.1 and Win 10
- Built-in hardware timing checks based on elapsed time between two sequential event marks

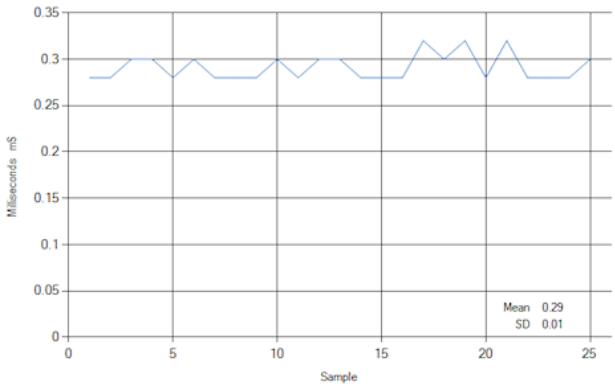
#Dependent on specific USB subsystem and using the USB TTL Configuration utility to check correct for correct installation.



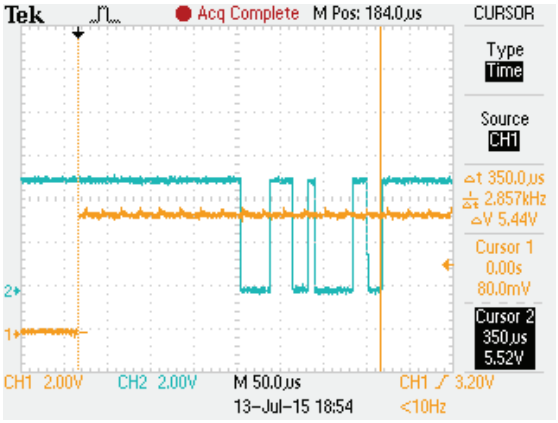
Typical time to send a TTL output or event mark (orange) from a two byte hex code from the PC (blue)



TTL sampling rate, i.e. how many times a second TTL I/O status is checked



Typical time for 50 TTL outputs or event marking pairs (on|off) measured using our configuration and latency validation software for Microsoft Windows (Win 7 64 SP1, USB 2)



Typical time to respond to a TTL input/trigger pulse or sync signal (orange) and send a two byte hex code to the PC (blue)

Copyright © 2015 The Black Box ToolKit Ltd. All rights reserved.