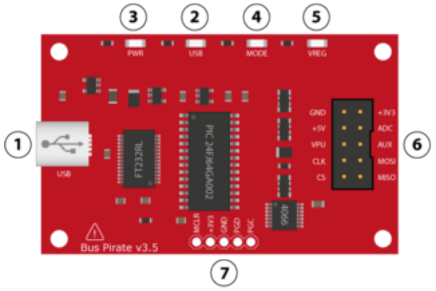
Hardware overview

SOIC Version

Last Version



1. Mini-B USB port. Connects the Bus Pirate to a PC. The Bus Pirate draws power from the USB port, and uses the data connection to communicate with the PC.

2. USB transmit indicator. This LED flashes when there's traffic from the PIC to the PC.

3. Power indicator. This LED lights when the Bus Pirate is powered by the USB supply.

4. Mode indicator. This LED lights when the Bus Pirate is configured for a protocol mode from the user terminal (menu 'm'). The I/O pins might be active when the mode indicator is on. The pins should be in a safe, non-powered, high-impedance state when the mode LED is off.

5. Voltage regulator indicator. This LED lights when the on-board power supplies have been activated from the user terminal (command capital 'W').

6. I/O pins. This 2x5 block of 0.1" pin header connects the Bus Pirate to external circuits. See the pinout table below, or the Bus Pirate manual.

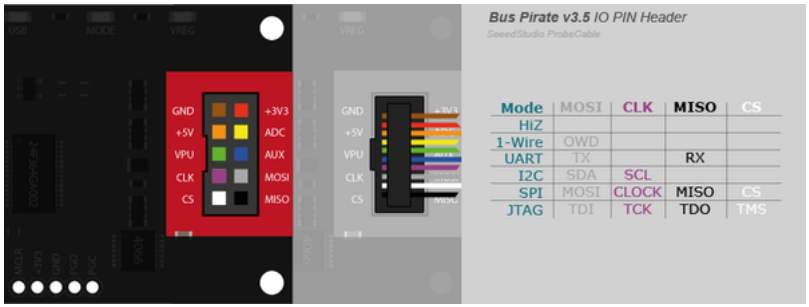


Image shows Colors from BusPirate ProbeKit available at Seeed Studio on latest Version

Bus Pirate - IO Pins

Pin Name	Description (Bus Pirate is the master)

MOSI	Master data out, slave in (SPI, JTAG), Serial data (1-Wire, I2C, KB), TX* (UART)
CLK	Clock signal (I2C, SPI, JTAG, KB)
MISO	Master data in, slave out (SPI, JTAG) RX (UART)
CS*	Chip select (SPI), TMS (JTAG)
AUX	Auxiliary IO, frequency probe, pulse-width modulator
ADC	Voltage measurement probe (max 6volts)
Vpu	Voltage input for on-board pull-up resistors (0-5volts).
+3.3v	+3.3volt switchable power supply
+5.0v	+5volt switchable power supply
GND	Ground, connect to ground of test circuit

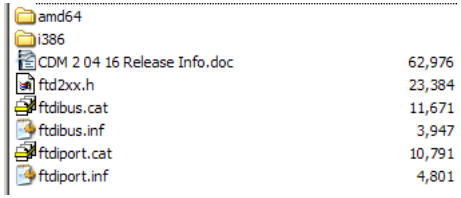
*Notes: * TX moved from CS to MOSI in firmware v0g*

7. In circuit serial programming (ICSP) header. This 1x5 block of 0.1" pin header is the programming connection for the PIC 24FJ64GA002 microcontroller. These pins can be used to write new firmware to the microcontroller with a programmer like the PICKIT2 or ICD2 . The Bus Pirate firmware can also be updated over the USB connection using a bootloader, so the ICSP header is normally only used to program it the first time at the factory. Put a jumper between the PGC and PGD pins to trigger the on-board bootloader for firmware updates.

8. Serial terminal (ST) header. Version v2go only. This unpopulated header is a tap into the UART connection between the PIC microcontroller and the FTDI 232BL chip that provides the USB connection. The Bus Pirate firmware defaults to a 115200bps/8/N/1 UART.

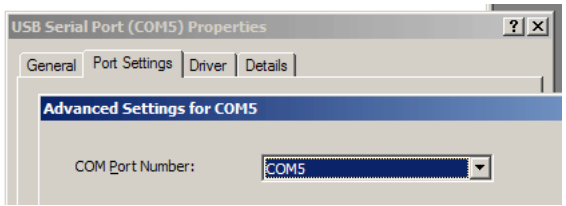
Retrieved from "http://dangerousprototypes.com/docs/Hardware_overview"

FTDI driver install and configuration



Windows will request a driver the first time the Bus Pirate connects to a PC. Extract the 2.08.28 virtual com port drivers from FTDI into a folder and browse to them using the 'Found New Hardware' wizard. Note: the 2.08.30 drivers gave some people connection errors between the Windows terminal and the Bus Pirate. For now we advise to install the 2.08.28 drivers. Install guides and drivers for other systems are also available on the FTDI driver download page.

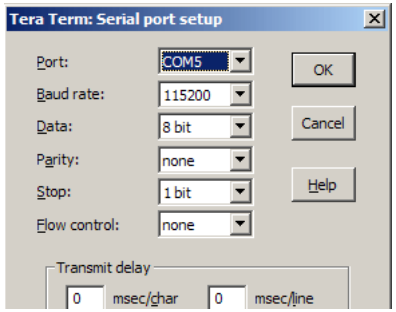
To find the COM port number assigned to the Bus Pirate go to the Windows device manager (*Start->Settings->Control panel->System->Hardware->Device manager*). Look in 'Ports (COM & LPT)' for 'USB Serial Port', ours is COM5.



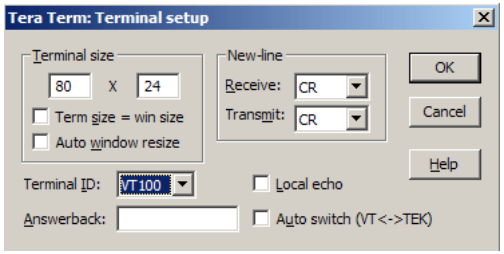
You can change the serial port assigned to the FTDI chip. Go to *USB Serial Port properties->Port settings tab->Advanced*, change the COM port in the drop-down box.

Terminal setup

Windows terminal is cranky, but it appears to work with the Bus Pirate when VT100 emulation is enabled. We highly recommend a better terminal, we like Tera Term Profor Windows.



First, configure the correct COM port and settings. The Bus Pirate operates at 115200bps/8/N/1 on the COM port assigned by Windows. *Flow control is no longer required. Disable it!*



Next, check the terminal setup. Turn off local echo and use a VT100 terminal type. The Bus Pirate should work with any type of new-line character, but we use the CR setting.

Get to know the terminal interface

The Bus Pirate is controlled by text commands entered through the serial terminal. If the terminal is blank, press *enter* to show the command prompt. Press '?', followed by*enter*, to show the help menu.

Menus configure various Bus Pirate options like pull-up resistors, terminal speed, data display format (DEC, HEX, BIN), etc. Type the menu command, followed by*enter*, to display the options.
Syntax is used to interact with a device connected over a bus. Commands are mostly single characters, such as 'r' to read a byte. Enter up to 4000 characters of syntax, press *enter* to execute the sequence.

Each menu and syntax option is documented fully in the Bus Pirate manual.

Most menus have a default option shown in () before the prompt:

Output type:
1. High-Z outputs (H=input, L=GND)
2. Normal outputs (H=Vcc, L=GND)
(1) > **<<< option 1 is the default**

Press *enter* to select the default option.

Bus modes, protocol libraries

```
?
General                         Protocol interaction
---------------------------------------------------------------------------
?       This help               (0)     List current macros
=X/|X   Converts X/reverse X            (x)     Macro x
```

```
~       Selftest                    [       Start
#       Reset                       ]       Stop
$       Jump to bootloader          {       Start with read
&/%     Delay 1 us/ms               }       Stop
a/A/@   AUXPIN (low/HI/READ)        "abc"   Send string
b       Set baudrate                123
c/C     AUX assignment (aux/CS)     0x123
d/D     Measure ADC (once/CONT.)    0b110   Send value
f       Measure frequency           r       Read
g/S     Generate PWM/Servo          /       CLK hi
h       Commandhistory              \       CLK lo
i       Versioninfo/statusinfo      ^       CLK tick
l/L     Bitorder (msb/LSB)          -       DAT hi
m       Change mode                 _       DAT lo
o       Set output type             .       DAT read
p/P     Pullup resistors (off/ON)   !       Bit read
s       Script engine               :       Repeat e.g. r:10
v       Show volts/states           .       Bits to read/write e.g. 0x55.2
w/W     PSU (off/ON)        <x>/<x= >/<0>   Usermacro x/assign x/list all
HiZ>m <<< bus mode menu
1. HiZ
2. 1-WIRE
3. UART
4. I2C
5. SPI
6. 2WIRE
7. 3WIRE
8. LCD
9. DIO
x. exit(without change)
```

The 'bus mode' menu (M) configures the Bus Pirate for a specific protocol, like 1-Wire, I2C, SPI, etc. The default start-up mode is HiZ, all pins are inputs and all power supplies are off.

```
(1)>5 <<< enter SPI bus mode
Set speed:
 1. 30KHz
 2. 125KHz
 3. 250KHz
 4. 1MHz

(1)>1
Clock polarity:
 1. Idle low *default
 2. Idle high

(1)>1
Output clock edge:
 1. Idle to active
 2. Active to idle *default

(2)>2
Input sample phase:
 1. Middle *default
 2. End

(1)>1
CS:
 1. CS
 2. /CS *default
```

```
(2)>2
Select output type:
 1. Open drain (H=Hi-Z, L=GND)
 2. Normal (H=3.3V, L=GND)

(1)>2  <<< option 1 is the default
Ready
SPI>W
Power supplies ON
SPI>w
Power supplies OFF
SPI>V
Syntax error at char 1
SPI>v
Pinstates:
1.(BR)  2.(RD)  3.(OR)  4.(YW)  5.(GN)  6.(BL)  7.(PU)  8.(GR)  9.(WT)  0.(Blk)
GND     3.3V    5.0V    ADC     VPU     AUX     CLK     MOSI    CS      MISO
P       P       P       I       I       I       O       O       O       I
GND     0.00V   2.43V   0.00V   0.00V   L       L       L       H       L
SPI>W
Power supplies ON
SPI>
```

Power supplies

3.3volt and 5volt on-board power supplies can provide up to 150mA for your project. Activate them with the w/W command from any mode except HiZ mode. HiZ mode is a safe mode and all outputs are disabled.

```
I2C>w<<<power supplies off
POWER SUPPLIES OFF
I2C>v<<<voltage report
Voltage monitors: 5V: 0.0 | 3.3V: 0.0 | VPULLUP: 0.0 |
I2C>W<<<power supplies on
POWER SUPPLIES ON
I2C>v<<<voltage monitor report
Voltage monitors: 5V: 4.9 | 3.3V: 3.2 | VPULLUP: 0.0 |
I2C>
```

Capital 'W' activates the on-board supplies, small 'w' turns them off. Turn the power supplies on, then press v to show a power supply voltage report.

Note that W is syntax and not a menu option, it can be used with other syntax to toggle the power in the middle of complex bus operations.

The power supplies will try to protect your project with over current protection. Additionally, the Bus Pirate will measure the voltage shortly after the supplies are enabled. If a short is detected it will disable them and show a warning. This may help minimize damage to a project when there is a short or problem on a new board.

Demo with AT25080A/160A/320A/640A on Aardvark I2cC/SPI Activity Board

Instruction Set for the AT25080A/160A/320A/640A

Instruction Name	Instruction Format	Operation
WREN	0000 0110	Set Write Enable Latch
WRDI	0000 0100	Reset Write Enable Latch
RDSR	0000 0101	Read Status Register
WRSR	0000 0001	Write Status Register
READ	0000 0011	Read Data from Memory Array
WRITE	0000 0010	Write Data to Memory Array

A quick way to find the command to access a register:

Use the Bus Pirate to find the binary equivalent (= command)

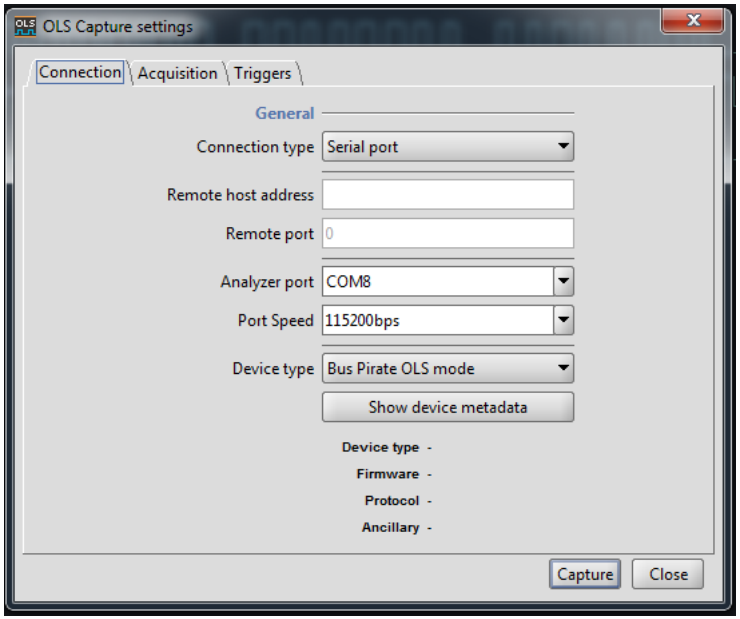
SPI>[0000 0101 r]

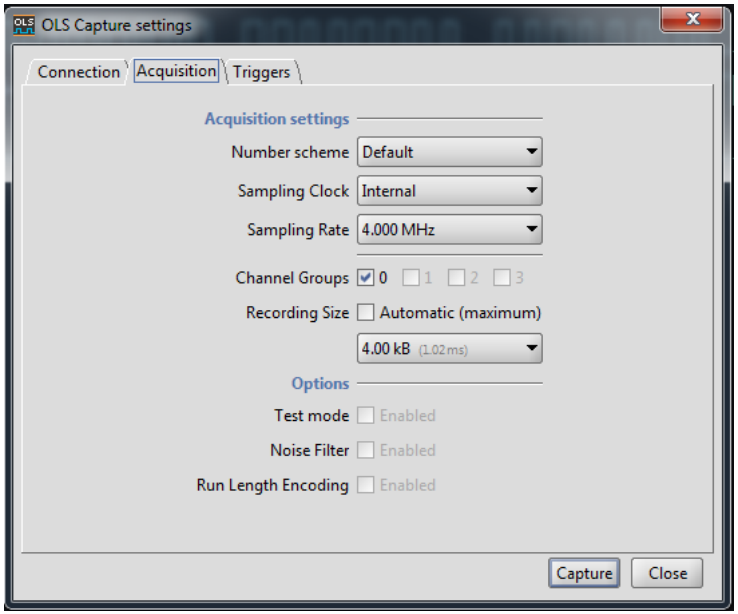
Lower chip select ([), then setup the register to access(0x05), read a byte (r), and raise chip select (]).

/CS ENABLED
WRITE: 0x00
WRITE: 0x4D
READ: 0x00
/CS DISABLED

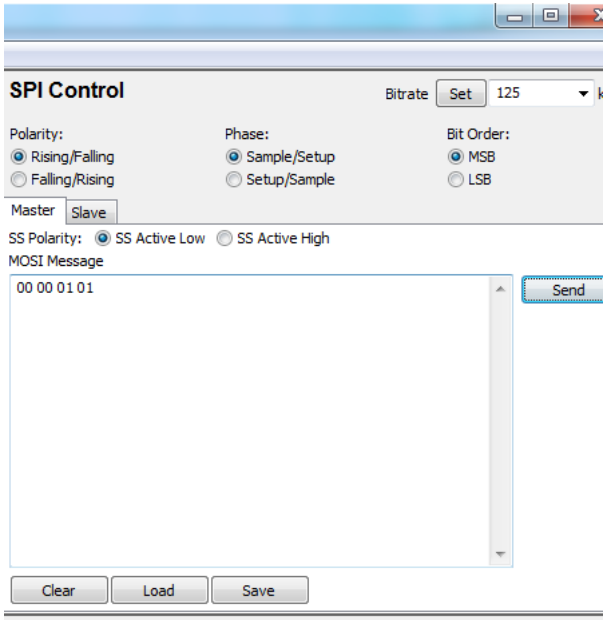
Open Logic Analyzer Client

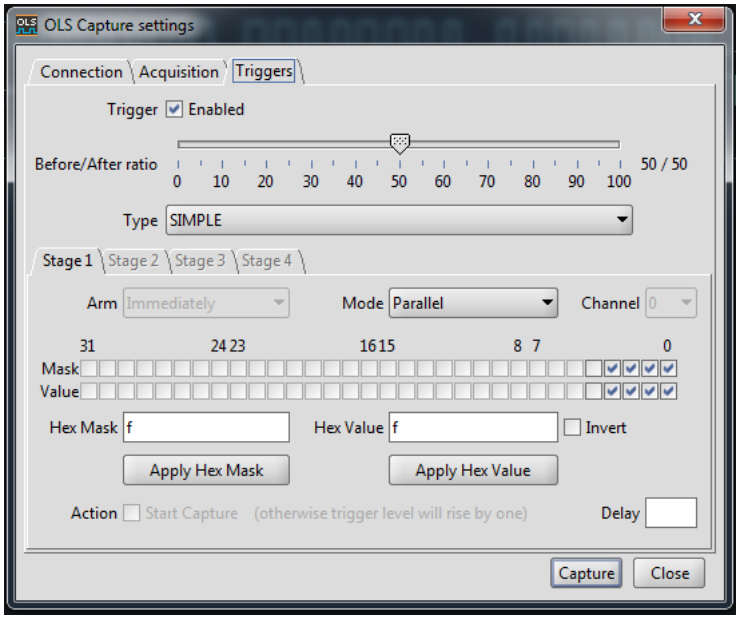
http://www.lxtreme.nl/ols/





The Bus Pirate V3 can store 4kB of data and can sample at a maximum rate of 1MHz. In this example we analyzed communication at 125kHz on the aardvark so we need to set the speed at a fast enough rate to be able to capture the transitions from high to low and low to high. It is recommended that you choose a sampling rate 10x your communication rate but in reality it can be lower.

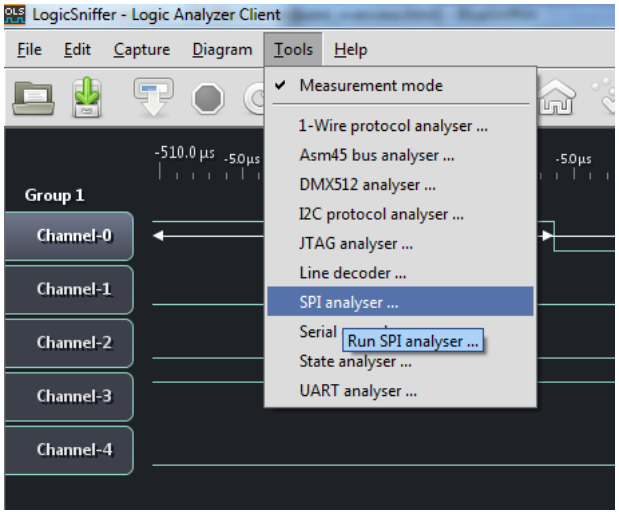


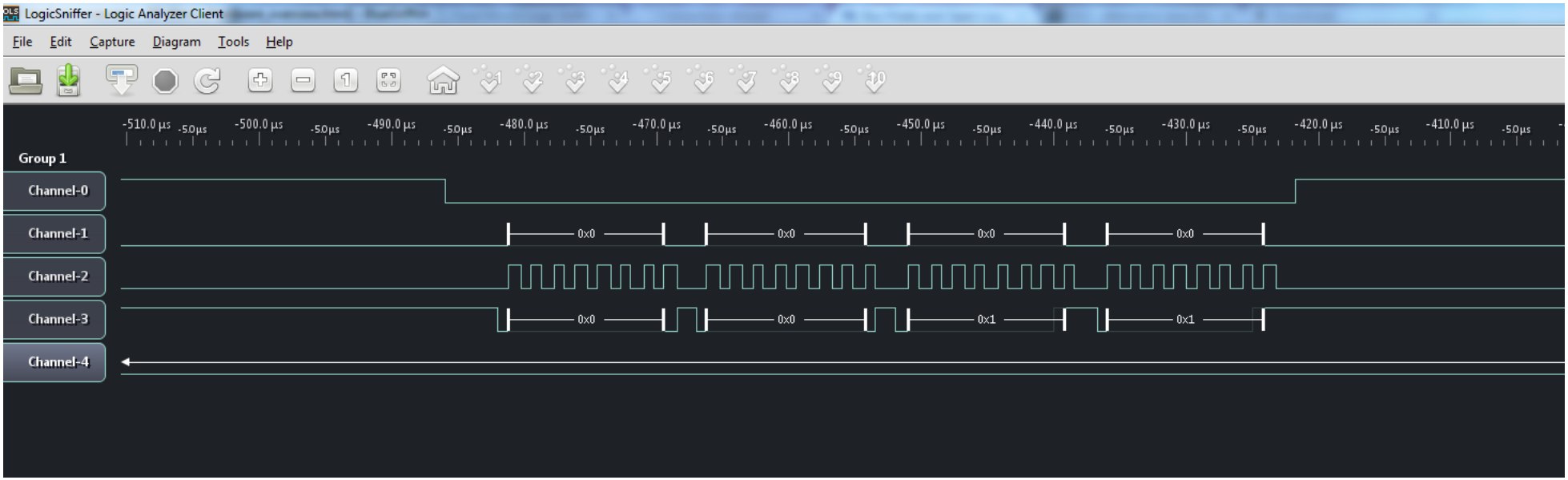


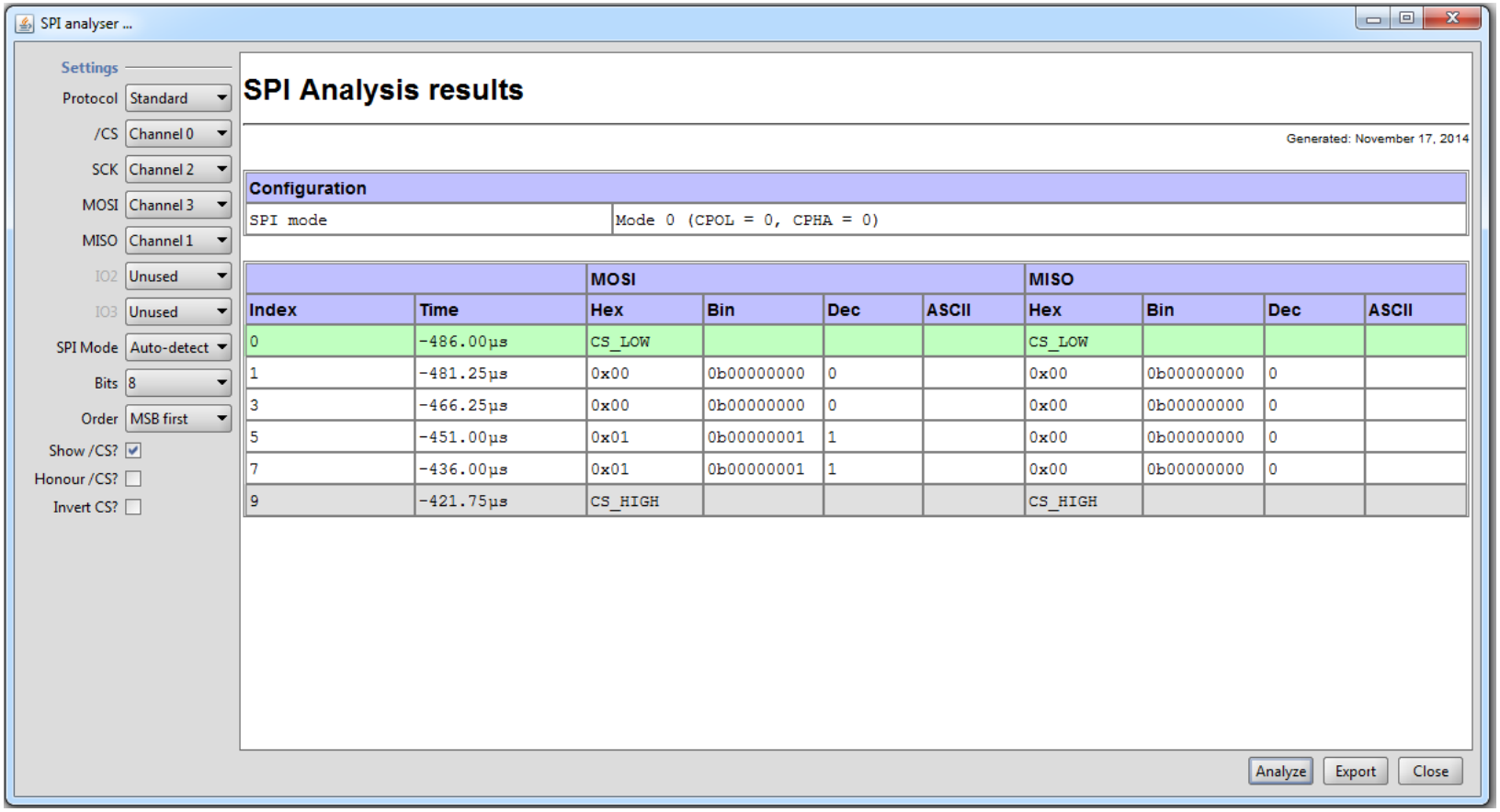
Click on the **Trigger Enabled** tick box to turn the triggers on.

At the bottom of this screen you will see Mask / Value tick boxes going from 0 to 31. Our Bus Pirate only has 5 logic inputs so 5 through to 31 are greyed out. Tick in the Mask boxes any inputs that you want a trigger enabled on. Value bits are currently ignored, any change triggers the capture.

For SPI select 0,1,2,3 where




LogicSniffer - Logic Analyzer Client
File Edit Capture Diagram Tools Help
-510.0 µs -50µs -500.0 µs -50µs -490.0 µs -50µs -480.0 µs -50µs -470.0 µs -50µs -460.0 µs -50µs -450.0 µs -50µs -440.0 µs -50µs -430.0 µs -50µs -420.0 µs -50µs -410.0 µs -50µs
Group 1
Channel-0
Channel-1
0x0
0x0
0x0
0x0
Channel-2
Channel-3
0x0
0x0
0x1
0x1
Channel-4



Bus Pirate binary SPI sniffer utility

This is a small program that displays SPI sniffer data from the Bus Pirate. It uses the binary mode SPI sniffer access, so it has a speed advantage over the terminal mode display. The computer takes the burden of converting raw byte values to HEX output.

Version

Clock edge and polarity did not work in v0.2. v0.3 fixes that, plus attempts to decode the output to human-readable format.

The old raw mode is still available with the -r flag.

Downloads

A Windows version and source is available here (v0.3).
The latest source is available in SVN and should compile on most systems with minimal tweaking.

Usage

```
SPIsniffer  -d <serial port> -e <clock edge> -p <clock polarity> -s <port speed> -r <output mode>
```

Where

-d device is port e.g. COM1 **(required)**
-e ClockEdge is 0 or 1 default is 1

-p Polarity is 0 or 1 default is 0
-s Speed is port Speed default is 115200
-r Rawoutput is 0 for decoded output, 1 for raw HEX output. Default is 0.

Only the serial port is required.

```
spisniffer -d COM3
spisniffer -d COM3 -e 0 -p 1
```

This is a Windows example. On Linux it would look like this:

```
spisniffer -d /dev/ttyUSB0
spisniffer -d /dev/ttyUSB0 -e 0 -p 1
```

On Linux, the usage of the Escape key for exiting seems to be broken, so one needs to re-attach the Bus Pirate after exiting with Ctrl-C.

Could you please add a Mac example if you're using this app on this platform?

Example

C:\>**spisniffer -d COM3 -r 1**
Bus Pirate binary mode SPI SNIFFER utility v0.3 (CC-0)
http://dangerousprototypes.com
Press escape to exit

Parameters used: Device = COM3, Speed = 115200, Clock Edge= 1, Polarity= 0

Opening Bus Pirate on COM3 at 115200bps...
Starting SPI sniffer...
Configuring Bus Pirate...
Entering binary mode...
(OK) Happy sniffing! Press ESC to stop.
5B 5C 01 00 5C 02 00 5C 03 00 5D
Esc key hit, stopping...
Clean up Bus Pirate...
(Bye for now!)

C:\>pause

Run the SPIsniffer utility from the command line. It configures the Bus Pirate and displays any values output by the sniffer mode.

Press escape to exit the program.

v0.3 attempts to decode the output to human-readable format. The old raw mode is still available with the -r flag.

Raw output decoding

```
v0.3 attempts to decode the output to human-readable format.
The old raw mode is still available with the -r flag.
```

See the SPI binary mode documentation for more about the SPI sniffer output. Here's a quick example.

```
5B 5C 01 00 5C 02 00 5C 03 00 5D
```

This output was sniffed from another Bus Pirate that sent *[0x01 0x02 0x03]*, which is also: CS low, 1, 2, 3, CS high.

5B - ASCII code for [(CS low).
5C - ASCII code for \, escapes each data pair. This is repeated with each byte pair so you can find your place in the datastream.
01 - Value sniffed on the Bus Pirate MOSI pin.
00 - Value sniffed on the MISO pin.
5C - \, escapes the next data pair.
02 - Value sniffed on the MOSI pin.
00 - Value sniffed on the MISO pin.
5C - \, escapes the next data pair.

03 - Value sniffed on MOSI pin.
00 - Value sniffed on MISO pin.
5D - ASCII code for] (CS high).

Limitations

Currently only sniffs when CS low with default SPI settings.
This can be changed by modifying the source, but should be added as a command line option.