How to use the BrainVision TriggerBox application interface

Once the TriggerBox software is installed, a serial port is available for each application which is able to handle serial port communications. The COM port number can be found in the “Ports (COM & LPT)” section of the device manager. Search the entry “TriggerBox VirtualSerial Port (COMx)” and take the COM port number.

From your application open the serial port. The serial port settings have no influence on the transmission speed of the interface because it’s a pure virtual port and data is always transmitted as fast as possible. So you can either leave the default port settings or, if your application requires those parameters, you can use dummy values like:

- Bits per second: 9600
- Data bits: 8
- Parity: None
- Stop bits: 1
- Flow control: None

Each byte written to the COM port is transmitted to the eight output lines (Bit 0 – Bit 7) of the TriggerBox “Output (to Amp)” connector.
Starting with driver version 1.2.0 the input is event driven. A new byte is available for reading when the state of the TriggerBox “Input (8-15)” lines has changed.

If you are finished, the output line should be reset to their default levels by writing a 0xFF to the serial port and the port must be closed.

Examples

Below are examples for the TriggerBox usage in

- C++
- Python
- C#
- MATLAB®
- MATLAB® + Psychophysics Toolbox.

Please note, for all examples you need at least the TriggerBox driver version 1.2.0.

C++ Example

```
#include "stdafx.h"
#include <windows.h>
#include <iostream>

HANDLE hPort;
BOOL   Terminate = FALSE;

// The serial port read thread
ULONG RxThread(void *arg)
{
   BYTE buffer[10000];
   DWORD bytesRead;
   BOOL ok;
   BOOL WaitingOnRead;
   DWORD dwRes;

   try
   {
      OVERLAPPED ovFile = { 0 };
      ovFile.hEvent = CreateEvent(0, true, 0, 0);

      OVERLAPPED ov = { 0 };
      DWORD dwEventMask;
      ov.hEvent = CreateEvent(0, true, 0, 0);
      ok = SetCommMask(hPort, EV_RXCHAR);
      WaitingOnRead = FALSE;

      while (!Terminate)
      {
         if (!WaitingOnRead)
         {
            WaitCommEvent(hPort, &dwEventMask, &ov);
            dwRes = WaitForSingleObject(ov.hEvent, 20);

            // issue a new read request
            ok = ReadFile(hPort, buffer, sizeof(buffer), &bytesRead, &ovFile);
            if (!ok)
            {
               DWORD lastError = ::GetLastError();
               if (lastError != ERROR_IO_PENDING)
               {
                  std::cout << "error " << lastError << " in read thread" << std::endl;
                  Terminate = true;
               }
               else
                  // read operation delayed
                  WaitingOnRead = TRUE;
            }
            else
            {
               // read completed immediately
               if (bytesRead)
               {
                  for (DWORD b = 0; b < bytesRead; b++)
                     std::cout << (int)buffer[b] << std::endl;
               }
            }
         }
         else
         {
            // delayed read operation
            dwRes = WaitForSingleObject(ovFile.hEvent, 20);
            switch (dwRes)
            {
            case WAIT_OBJECT_0:
               // Read completed.
               if (!GetOverlappedResult(hPort, &ovFile, &bytesRead, FALSE))
               {
                  // Error in communications; report it.
                  DWORD lastError = ::GetLastError();
                  std::cout << "error " << lastError << " in read thread" << std::endl;
                  Terminate = true;
               }
               else
               {
                  // Read completed successfully.
                  if (bytesRead)
                  {
                     for (DWORD b = 0; b < bytesRead; b++)
                        std::cout << (int)buffer[b] << std::endl;
                  }

                  //  Reset flag so that another opertion can be issued.
                  WaitingOnRead = FALSE;
               }
               break;

            case WAIT_TIMEOUT:
               // Operation isn't complete yet.
               // This is a good time to do some background work.
               break;

            default:
               // Error in the WaitForSingleObject; abort.
               // This indicates a problem with the OVERLAPPED structure's
               // event handle.
               DWORD lastError = ::GetLastError();
               std::cout << "error " << lastError << " in read thread" << std::endl;
               Terminate = true;
               break;
            }
         }
      }

      // clean up
      CloseHandle(ovFile.hEvent);
   }
   catch (...)
   {
      std::cout << "exception in read thread" << std::endl;
   }
   ExitThread(0);
   return(0);
}


int _tmain(int argc, _TCHAR* argv[])
{
   BYTE data;
   HANDLE hRcv = 0;
   DWORD  ThreadId = 0;
   int    *ThreadIx = 0;
   OVERLAPPED ov = { 0 };
   ov.hEvent = CreateEvent(0, true, 0, 0);

   // Open the Windows device manager,
   // search for the "TriggerBox VirtualSerial Port (COMx)"
   // in "Ports /COM & LPT)" and enter the COM port number as file name.
   // For the usage of serial ports larger than COM9
   // see https://support.microsoft.com/en-us/kb/115831
   hPort = CreateFile(_T("\\\\.\\COM6"),
      GENERIC_WRITE | GENERIC_READ, 0, NULL, OPEN_EXISTING, FILE_FLAG_OVERLAPPED, NULL);
   if (hPort == INVALID_HANDLE_VALUE)
   {
      std::cout << "failed to open COM port" << std::endl;
      return -1;
   }
```

```
    // Create and start the read thread
    hRcv = CreateThread(0, 0, (LPTHREAD_START_ROUTINE)&RxThread, &ThreadIx, (DWORD)NULL, &ThreadId);

    // Set the port to an initial state
    data = 0x00;
    WriteFile(hPort, &data, 1, NULL, &ov);
    WaitForSingleObject(ov.hEvent, 20);
    Sleep(10);

    // Set Bit 0, Pin 2 of the Output(to Amp) connector
    data = 0x01;
    WriteFile(hPort, &data, 1, NULL, &ov);
    WaitForSingleObject(ov.hEvent, 20);
    Sleep(10);

    // Reset Bit 0, Pin 2 of the Output(to Amp) connector
    data = 0x00;
    WriteFile(hPort, &data, 1, NULL, &ov);
    WaitForSingleObject(ov.hEvent, 20);
    Sleep(10);

    // Reset the port to its default state
    data = 0xFF;
    WriteFile(hPort, &data, 1, NULL, &ov);
    WaitForSingleObject(ov.hEvent, 20);
    Sleep(10);

    // Terminate the read thread
    Terminate = TRUE;
    WaitForSingleObject(hRcv, (DWORD)2000);
    CloseHandle(hRcv);


    // Close the serial port
    CloseHandle(hPort);

    return 0;
}
```

Python Example

```python
import serial
import time
import threading

Connected = True
PulseWidth = 0.01

def ReadThread(port):
    while Connected:
        if port.inWaiting() > 0:
            print ("0x%X"%ord(port.read(1)))


# Open the Windows device manager, search for the "TriggerBox VirtualSerial Port (COM6)"
# in "Ports /COM & LPT)" and enter the COM port number in the constructor.
port = serial.Serial("COM6")

# Start the read thread
thread = threading.Thread(target=ReadThread, args=(port,))
thread.start()

# Set the port to an initial state
port.write([0x00])
time.sleep(PulseWidth)

# Set Bit 0, Pin 2 of the Output(to Amp) connector
port.write([0x01])
time.sleep(PulseWidth)

# Reset Bit 0, Pin 2 of the Output(to Amp) connector
port.write([0x00])
time.sleep(PulseWidth)

# Reset the port to its default state
port.write([0xFF])
time.sleep(PulseWidth)

# Terminate the read thread
Connected = False
thread.join(1.0)

# Close the serial port
port.close()
```

C# Example

```
using System;
using System.Collections.Generic;
using System.IO.Ports;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Threading;

namespace TriggerBox
{
    class Program
    {
        static void Main(string[] args)
        {
            Byte[] data = { (Byte)0 };

            // Open the Windows device manager, search for the "TriggerBox VirtualSerial Port (COMx)"
            // in "Ports /COM & LPT)" and enter the COM port number in the constructor.
            SerialPort TriggerBox = new SerialPort("COM6");
            TriggerBox.Open();
            TriggerBox.ReadTimeout = 5000;
            // Attach the data received event handler
            TriggerBox.DataReceived += TriggerBox_DataReceived;

            // Set the port to an initial state
            data[0] = 0x00;
            TriggerBox.Write(data,0,1);
            Thread.Sleep(10);

            // Set Bit 0, Pin 2 of the Output(to Amp) connector
            data[0] = 0x01;
            TriggerBox.Write(data, 0, 1);
            Thread.Sleep(10);

            // Reset Bit 0, Pin 2 of the Output(to Amp) connector
            data[0] = 0x00;
            TriggerBox.Write(data, 0, 1);
            Thread.Sleep(10);

            // Reset the port to its default state
            data[0] = 0xFF;
            TriggerBox.Write(data, 0, 1);
            Thread.Sleep(10);

            // Close the serial port
            TriggerBox.DataReceived -= TriggerBox_DataReceived;
            TriggerBox.Close();
        }

        static void TriggerBox_DataReceived(object sender, SerialDataReceivedEventArgs e)
        {
            SerialPort sp = (SerialPort)sender;
            // get all available bytes from the input buffer
            while (sp.BytesToRead > 0)
            {
                Console.WriteLine(sp.ReadByte());
            }
        }
    }
}
```

MATLAB® Example

```
%%% Sample code to write to and read from TriggerBox from Brain Products GmbH

%%% Important note: Make sure the toggle switches on the TriggerBox
%%% are set to 'PCx' (and NOT 'Inx'; x=BitNo) for all used bits
%%% i.e. if e.g. only bits 0 to 3 are used, switches 0 to 3 need to be set

%%% Further information: See Matlab Help "Getting Started with Serial I/O"

% !!! Important Note: When using virtual serial ports
% !!! remember to restart MATLAB, once you plug in the TriggerBox,
% !!! for the new port to show up in MATLAB.
% !!! This is to allow the underlying serial API to update the list
% !!! of available serial ports.


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%         START of sample code         %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% To construct the right serial port object,
% open the Windows device manager and search for the "TriggerBox VirtualSerial Port (COMx)"
% in section "Ports (COM & LPT)" and adjust/enter the COM port number x in the following constructor
SerialPortObj=serial('COM4', 'TimeOut', 1); % in this example x=4
SerialPortObj.BytesAvailableFcnMode='byte';
SerialPortObj.BytesAvailableFcnCount=1;
SerialPortObj.BytesAvailableFcn=@ReadCallback;

% To connect the serial port object with serial port hardware
fopen(SerialPortObj);

% Set the port to zero state 0
fwrite(SerialPortObj, 0,'sync');
pause(0.01);

% Set Bit 0 (Pin 2 of the Output(to Amp) connector)
fwrite(SerialPortObj, 1,'sync');
pause(0.01);

% Reset the port to zero state 0
fwrite(SerialPortObj, 0,'sync');
pause(0.01);

% Reset the port (i.e. bit 0 to 7) to its resting state 255
fwrite(SerialPortObj, 255,'sync');
pause(0.01);

% Then disconnect/close the serial port object from the serial port
fclose(SerialPortObj);

% Remove the serial port object from memory
delete(SerialPortObj);

% Remove the serial port object from the MATLAB® workspace
clear SerialPortObj;

% Read callback function
function ReadCallback(src, event)
    %disp(event.Type);
    if(src.BytesAvailable > 0)
        disp(fread(src, src.BytesAvailable));
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%          END of sample code          %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

MATLAB® Psychophysics Toolbox Example

```
%%% Sample code to write to and read from TriggerBox from Brain Products GmbH
%%% using the Psychophysics Toolbox functions

%%% Important note: Make sure the toggle switches on the TriggerBox
%%% are set to 'PCx' (and NOT 'Inx'; x=BitNo) for all used bits
%%% i.e. if e.g. only bits 0 to 3 are used, switches 0 to 3 need to be set

%%% Further information: See http://docs.psychtoolbox.org/IOPort

% !!! Important Note: When using virtual serial ports
% !!! remember to restart MATLAB, once you plug in the TriggerBox,
% !!! for the new port to show up in MATLAB.
% !!! This is to allow the underlying serial API to update the list
% !!! of available serial ports.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%         START of sample code         %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% To construct and open the right serial port object,
% open the Windows device manager and search for the "TriggerBox VirtualSerial Port (COMx)"
% in section "Ports (COM & LPT)" and adjust/enter the COM port number x in the following constructor
TB = IOPort('OpenSerialPort', 'COM6'); % in this example x=6
% Read data from the TriggerBox
Available = IOPort('BytesAvailable', TB);
if(Available > 0)
    disp(IOPort('Read', TB, 0, Available));
end

% Set the port to zero state 0
IOPort('Write', TB, uint8(0), 0);
pause(0.01);
% Read data from the TriggerBox
Available = IOPort('BytesAvailable', TB);
if(Available > 0)
    disp(IOPort('Read', TB, 0, Available));
end

% Set Bit 0 (Pin 2 of the Output(to Amp) connector)
IOPort('Write', TB, uint8(1), 0);
pause(0.01);
% Read data from the TriggerBox
Available = IOPort('BytesAvailable', TB);
if(Available > 0)
    disp(IOPort('Read', TB, 0, Available));
end

% Reset the port to zero state 0
IOPort('Write', TB, uint8(0), 0);
pause(0.01);
% Read data from the TriggerBox
Available = IOPort('BytesAvailable', TB);
if(Available > 0)
    disp(IOPort('Read', TB, 0, Available));
end

% Reset the port (i.e. bit 0 to 7) to its resting state 255
IOPort('Write', TB, uint8(255), 0);
pause(0.01);
% Read data from the TriggerBox
Available = IOPort('BytesAvailable', TB);
if(Available > 0)
    disp(IOPort('Read', TB, 0, Available));
end

% Then disconnect/close the serial port object from the serial port
IOPort('Close', TB);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%          END of sample code          %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```