**Algoritmo para conexión de Arduino con mission planner para poder conectar directamente a USB y realizar las rutas establecidas:**

using System;

using System.Collections.Generic;

using System.Reflection;

using System.Text;

using System.Threading;

using log4net;

using MissionPlanner.Comms;

// Written by Michael Oborne

namespace MissionPlanner.Arduino

{

public class ArduinoSTK : SerialPort, IArduinoComms

{

private static readonly ILog log = LogManager.GetLogger(MethodBase.GetCurrentMethod().DeclaringType);

public event ProgressEventHandler Progress;

public new void Open()

{

// default dtr status is false

//from http://svn.savannah.nongnu.org/viewvc/RELEASE\_5\_11\_0/arduino.c?root=avrdude&view=markup

base.Open();

base.DtrEnable = false;

base.RtsEnable = false;

System.Threading.Thread.Sleep(50);

base.DtrEnable = true;

base.RtsEnable = true;

System.Threading.Thread.Sleep(50);

}

/// <summary>

/// Used to start initial connecting after serialport.open

/// </summary>

/// <returns>true = passed, false = failed</returns>

public bool connectAP()

{

if (!this.IsOpen)

{

return false;

}

int a = 0;

while (a < 50) // 50 tries at 50 ms = 2.5sec

{

this.DiscardInBuffer();

this.Write(new byte[] { (byte)'0', (byte)' ' }, 0, 2);

a++;

Thread.Sleep(50);

log.InfoFormat("connectap btr {0}", this.BytesToRead);

if (this.BytesToRead >= 2)

{

byte b1 = (byte)this.ReadByte();

byte b2 = (byte)this.ReadByte();

if (b1 == 0x14 && b2 == 0x10)

{

return true;

}

}

}

return false;

}

/// <summary>

/// Used to keep alive the connection

/// </summary>

/// <returns>true = passed, false = lost connection</returns>

public bool keepalive()

{

return connectAP();

}

/// <summary>

/// Syncs after a private command has been sent

/// </summary>

/// <returns>true = passed, false = failed</returns>

public bool sync()

{

if (!this.IsOpen)

{

return false;

}

this.ReadTimeout = 1000;

int f = 0;

while (this.BytesToRead < 1)

{

f++;

System.Threading.Thread.Sleep(1);

if (f > 1000)

return false;

}

int a = 0;

while (a < 10)

{

if (this.BytesToRead >= 2)

{

byte b1 = (byte)this.ReadByte();

byte b2 = (byte)this.ReadByte();

log.DebugFormat("bytes {0:X} {1:X}", b1, b2);

if (b1 == 0x14 && b2 == 0x10)

{

return true;

}

}

log.DebugFormat("btr {0}", this.BytesToRead);

Thread.Sleep(10);

a++;

}

return false;

}

/// <summary>

/// Downloads the eeprom with the given length - set Address first

/// </summary>

/// <param name="length">eeprom length</param>

/// <returns>downloaded data</returns>

public byte[] download(short length)

{

if (!this.IsOpen)

{

throw new Exception();

}

byte[] data = new byte[length];

byte[] command = new byte[] { (byte)'t', (byte)(length >> 8), (byte)(length & 0xff), (byte)'E', (byte)' ' };

this.Write(command, 0, command.Length);

if (this.ReadByte() == 0x14)

{ // 0x14

int step = 0;

while (step < length)

{

byte chr = (byte)this.ReadByte();

data[step] = chr;

step++;

}

if (this.ReadByte() != 0x10) // 0x10

throw new Exception("Lost Sync 0x10");

}

else

{

throw new Exception("Lost Sync 0x14");

}

return data;

}

public byte[] downloadflash(short length)

{

if (!this.IsOpen)

{

throw new Exception("Port Not Open");

}

byte[] data = new byte[length];

this.ReadTimeout = 1000;

byte[] command = new byte[] { (byte)'t', (byte)(length >> 8), (byte)(length & 0xff), (byte)'F', (byte)' ' };

this.Write(command, 0, command.Length);

if (this.ReadByte() == 0x14)

{ // 0x14

int read = length;

while (read > 0)

{

//Console.WriteLine("offset {0} read {1}", length - read, read);

read -= this.Read(data, length - read, read);

//System.Threading.Thread.Sleep(1);

}

if (this.ReadByte() != 0x10) // 0x10

throw new Exception("Lost Sync 0x10");

}

else

{

throw new Exception("Lost Sync 0x14");

}

return data;

}

public bool uploadflash(byte[] data, int startfrom, int length, int startaddress)

{

if (!this.IsOpen)

{

return false;

}

int loops = (length / 0x100);

int totalleft = length;

int sending = 0;

for (int a = 0; a <= loops; a++)

{

if (totalleft > 0x100)

{

sending = 0x100;

}

else

{

sending = totalleft;

}

//startaddress = 256;

if (sending == 0)

return true;

setaddress(startaddress);

startaddress += sending;

byte[] command = new byte[] { (byte)'d', (byte)(sending >> 8), (byte)(sending & 0xff), (byte)'F' };

this.Write(command, 0, command.Length);

log.Info((startfrom + (length - totalleft)) + " - " + sending);

this.Write(data, startfrom + (length - totalleft), sending);

command = new byte[] { (byte)' ' };

this.Write(command, 0, command.Length);

totalleft -= sending;

if (Progress != null)

Progress((int)(((float)startaddress / (float)length) \* 100),"");

if (!sync())

{

log.Info("No Sync");

return false;

}

}

return true;

}

/// <summary>

/// Sets the eeprom start read or write address

/// </summary>

/// <param name="address">address, must be eaven number</param>

/// <returns>true = passed, false = failed</returns>

public bool setaddress(int address)

{

if (!this.IsOpen)

{

return false;

}

if (address % 2 == 1)

{

throw new Exception("Address must be an even number");

}

log.Info("Sending address " + ((ushort)(address / 2)));

address /= 2;

address = (ushort)address;

byte[] command = new byte[] { (byte)'U', (byte)(address & 0xff), (byte)(address >> 8), (byte)' ' };

this.Write(command, 0, command.Length);

return sync();

}

/// <summary>

/// Upload data at preset address

/// </summary>

/// <param name="data">array to read from</param>

/// <param name="startfrom">start array index</param>

/// <param name="length">length to send</param>

/// <param name="startaddress">sets eeprom start programing address</param>

/// <returns>true = passed, false = failed</returns>

public bool upload(byte[] data, short startfrom, short length, short startaddress)

{

if (!this.IsOpen)

{

return false;

}

int loops = (length / 0x100);

int totalleft = length;

int sending = 0;

for (int a = 0; a <= loops; a++)

{

if (totalleft > 0x100)

{

sending = 0x100;

}

else

{

sending = totalleft;

}

if (sending == 0)

return true;

setaddress(startaddress);

startaddress += (short)sending;

byte[] command = new byte[] { (byte)'d', (byte)(sending >> 8), (byte)(sending & 0xff), (byte)'E' };

this.Write(command, 0, command.Length);

log.Info((startfrom + (length - totalleft)) + " - " + sending);

this.Write(data, startfrom + (length - totalleft), sending);

command = new byte[] { (byte)' ' };

this.Write(command, 0, command.Length);

totalleft -= sending;

if (!sync())

{

log.Info("No Sync");

return false;

}

}

return true;

}

public Chip getChipType()

{

byte sig1 = 0x00;

byte sig2 = 0x00;

byte sig3 = 0x00;

byte[] command = new byte[] { (byte)'u', (byte)' ' };

this.Write(command, 0, command.Length);

System.Threading.Thread.Sleep(20);

byte[] chr = new byte[5];

int count = this.Read(chr, 0, 5);

log.Debug("getChipType read " + count);

if (chr[0] == 0x14 && chr[4] == 0x10)

{

sig1 = (byte)chr[1];

sig2 = (byte)chr[2];

sig3 = (byte)chr[3];

}

foreach (Chip item in Arduino.Chip.chips)

{

if (item.Equals(new Chip("", sig1, sig2, sig3, 0)))

{

log.Debug("Match "+item.ToString());

return item;

}

}

return null;

}

public new bool Close()

{

try

{

byte[] command = new byte[] { (byte)'Q', (byte)' ' };

this.Write(command, 0, command.Length);

}

catch { }

try

{

if (base.IsOpen)

base.Close();

}

catch { }

this.DtrEnable = false;

this.RtsEnable = false;

return true;

}

}

}

**Script de ejemplo para una ruta preestablecida en un punto determinado:**

# cs.???? = currentstate, any variable on the status tab in the planner can be used.

# Script = options are

# Script.Sleep(ms)

# Script.ChangeParam(name,value)

# Script.GetParam(name)

# Script.ChangeMode(mode) - same as displayed in mode setup screen 'AUTO'

# Script.WaitFor(string,timeout)

# Script.SendRC(channel,pwm,sendnow)

#

print 'Start Script'

for chan in range(1,9):

Script.SendRC(chan,1500,False)

Script.SendRC(3,Script.GetParam('RC3\_MIN'),True)

Script.Sleep(5000)

while cs.lat == 0:

print 'Waiting for GPS'

Script.Sleep(1000)

print 'Got GPS'

jo = 10 \* 13

print jo

Script.SendRC(3,1000,False)

Script.SendRC(4,2000,True)

cs.messages.Clear()

Script.WaitFor('ARMING MOTORS',30000)

Script.SendRC(4,1500,True)

print 'Motors Armed!'

Script.SendRC(3,1700,True)

while cs.alt < 50:

Script.Sleep(50)

Script.SendRC(5,2000,True) # acro

Script.SendRC(1,2000,False) # roll

Script.SendRC(3,1370,True) # throttle

while cs.roll > -45: # top hald 0 - 180

Script.Sleep(5)

while cs.roll < -45: # -180 - -45

Script.Sleep(5)

Script.SendRC(5,1500,False) # stabalise

Script.SendRC(1,1500,True) # level roll

Script.Sleep(2000) # 2 sec to stabalise

Script.SendRC(3,1300,True) # throttle back to land

thro = 1350 # will decend

while cs.alt > 0.1:

Script.Sleep(300)

Script.SendRC(3,1000,False)

Script.SendRC(4,1000,True)

Script.WaitFor('DISARMING MOTORS',30000)

Script.SendRC(4,1500,True)

print 'Roll complete'