Replay_bbox_ltm.rb

Jonathan Hudson `<https://github.com/stronnag/mwptools>`

Table of Contents

`replay_bbox_ltm.rb` is a ruby scrip to transpose Blackbox / INAV flight data recordings into LTM (LightTelemetry) messages over UDP or serial (Bluetooth) devices.

1. Introduction

Blackbox is an invaluable tool for development and flight analysis for Cleanflight and its derivatives such as INAV. This program (`replay_bbox_ltm.rb`) provides a means to replay a Blackbox recording in a geospatial context using one of the well know ground stations mwp[1] (Linux) and ezgui[2] (Android). The LTM (LightTelemetry) protocol is used to replay the Blackbox log data into the ground station; thus any ground station or OSD that supports LTM can be used for flight visualisation.

Note that `replay_bbox_ltm.rb` is integrated with mwp[3] and replay is available as a menu option. It is not necessary to read this document beyond Dependencies to use `replay_bbox_ltm.rb` with mwp[4].

2. Platforms

`replay_bbox_ltm.rb` is intended to run on most popular platforms (Linux, OSX, Windows). It is developed on Arch Linux and tested on Arch, Fedora, Ubuntu and occasionally, Windows 10.

3. Dependencies

There are a number of dependencies that must be satisfied to run the `replay_bbox_ltm.rb` script:

[1] https://github.com/stronnag/mwptools
[2] http://ez-gui.com/
[3] https://github.com/stronnag/mwptools
[4] https://github.com/stronnag/mwptools

- A Ruby Interpreter;
- The Ruby 'rubyserial' gem, and for Windows, the win32api gem;
- The Blackbox utility 'blackbox_decode'.

The installation of these dependencies is described below.

3.1. Dependency Installation

Ruby

Version 2.0 or greater is required.

- Linux: Install via your distribution package manager.
- OSX: you can use third-party tools (rbenv and RVM).
- Windows: Install the latest version from rubyinstaller.org[5] (aka MRI). Alternately, the cygwin[6] version may be used, however this requires cygwin gcc is also installed to build ffi (a rubyserial dependency otherwise satisfied by the MRI install). For MRI, set the options to associate `.rb` extensions with ruby.

> On Windows, run the environment specific interpreter; running the MRI (rubyinstaller) interpreter under cygwin may not work as expected. Using the cygwin ruby is unlikely to work for a serial device.

> On Ubuntu 14.04 LTS, you must specfically install Ruby 2.0, the default is 1.9.

```
sudo apt install ruby2.0
sudo update-alternatives --install /usr/bin/ruby ruby /usr/bin/ruby2.0 10
sudo update-alternatives --install /usr/bin/gem gem /usr/bin/gem2.0 10
```

> rubyserial depends on ffi. If your package manager does not offer this (it is part of the Windows install, but not default elsewhere), then you may need to install that as well, either as an explicit OS package, or via the `gem` command. rubyserial is an optional dependency.

For further details, please refer to https://www.ruby-lang.org/en/documentation/installation/.

Ruby Serial Gem

This is optional and is only necessary to replay blackbox files over bluetooth (e.g. it visualise in EzGui); it is not necessary to replay within mwp[7]. Once you have ruby installed, in a console (CLI) window:

```
$ gem install rubyserial
```

[5] http://rubyinstaller.org/downloads/
[6] https://www.cygwin.com/
[7] https://github.com/stronnag/mwptools

```
$ gem install win32api # Windows, excluding cygwin
```

Blackbox Utilities

Blackbox is hosted at github[8]. Instructions are included for all major operating systems. Binary releases for Windows and OSX are available from the releases page[9].

`blackbox_decode` must be on the user's PATH when the `replay_bbox_ltm.rb` script is invoked.

4. Installation

Installation is as simple as copying the `replay_bbox_ltm.rb` script somewhere convenient. The mwp[10] installer will install it for you, by default.

5. Communication with the GS

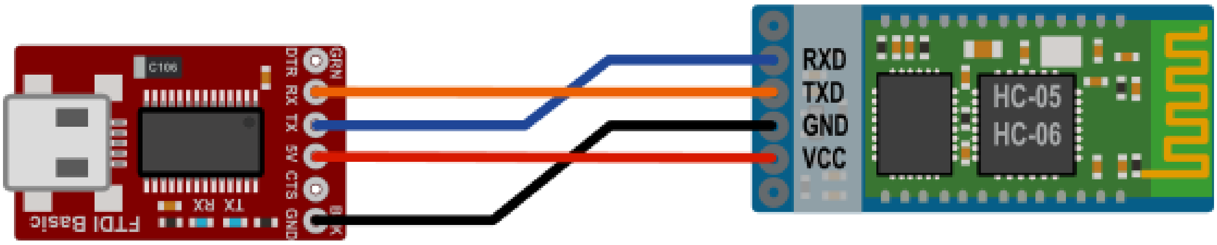
`replay_bbox_ltm.rb` can communicate with the ground station using a UDP socket or a serial device, typically a Bluetooth device for communications. At the time of publication, UDP is only supported by mwp[11].

5.1. Setting up serial communications

The easiest way to set up serial comms is to use a USB-TTY adaptor on the desktop computer running `replay_bbox_ltm.rb`, with a cross-over connection to a Bluetooth TTY device, which can then communicate with EZGUI (or mwp[12], though UDP is easier in this case).

USB	BT
5V	5V
Gnd	Gnd
RX	TX
TX	RX

The image below (from the librepilot wiki[13]) shows the connection.



Usage for ezgui and mwp[14] is decribed later.

[8] https://github.com/iNavFlight/blackbox-tools/

[9] https://github.com/iNavFlight/blackbox-tools/releases

[10] https://github.com/stronnag/mwptools

[11] https://github.com/stronnag/mwptools

[12] https://github.com/stronnag/mwptools

[13] http://opwiki.readthedocs.org/en/latest/user_manual/oplink/bluetooth.html

[14] https://github.com/stronnag/mwptools

Note that it is *possible* to use just the computer’s Bluetooth device (without a hardware adapter); setting this up is somewhat OS dependent and beyond the scope of this document. On Linux, one would use the `sdptool add channel=N SP` and `rfcomm listen` commands.

6. Usage

It is **essential** that `blackbox_decode` is on the PATH, so either install it on the extant PATH or extend the PATH to cover the directory containing `blackbox_decode`.

6.1. Verify the installation

You can verify the install by running ruby in 'check' mode:

```
$ ruby -cw ./replay_bbox_ltm.rb
Syntax OK
```

You should see the 'Syntax OK' message.

Next, verify that the application runs:

```
$ ./replay_bbox_ltm.rb --help
replay_bbox_ltm.rb [options] file
Replay bbox log as LTM
    -u, --udp=ADDR                   udp target (localhost:3000)
    -s, --serial-device=DEV
    -i, --index=IDX
    -t, --vehicle-type=TYPE
    -d, --declination=DEC            Mag Declination (default -1.3)
    -g, --force-gps-heading          Use GPS course instead of compass
    -4, --force-ipv4
    -?, --help                       Show this message
```

The [options] are:

--udp, -u

When using UDP as the transport for LTM messages, this defines the UDP host and port. If a host name is given, it is assumed that the specified host has bound to the defined socket. If the host name is blank, then `replay_bbox_ltm.rb` will bind to the socket. The UDP address is defined as:

```
udp://host:port
```

The udp:// part may be omitted, so valid examples are:

```
-u udp://:4321
--udp :4321
--udp udp://somehost:4321
-u somehost:4321
```

The first two examples are equivalent, and the last two are equivalent. By default, `replay_bbox_ltm.rb` prefers IPv6 where available. You can force IPv4 with the `--force-ipv4` option.

--serial-device, -s

Defines the serial device, where this is used as the LTM transport. For example:

```
-s /dev/ttyUSB0
--serial-device /dev/rfcomm1
-s COM6
```

The default baud rate is 115200, this may be changed by appending @rate to the serial definition:

```
-s /dev/rfcomm2@57600
-s COM7@38400
```

NOTE

Only one of UDP and serial definitions should be given.

--index, -i

Defines the index of the recording in the the blackbox file. If this is not given, the first recording (index 1) is used.

--declination, -d

The magnetic declination in decimal degrees. If not given, a value appropriate to the New Forest area of southern England is used.

You can preset the declination as decimal degrees in a JSON file `$HOME/.config/mwp/replay_ltm.json`, as (for the New Forest):

```
{"declination":-1}
```

--vehicle-type, -t

The vehicle type defines the vehicle icon shown in mwp or ezgui. Standard MultiWii values are used (mwp interpretation).

0. Undefined (an arrow in mwp, useful for checking the compass)
1. Tricopter
2. Quad+
3. QuadX
4. Bicopter
5. QuadX
6. Y6
7. Hex6+
8. Flying_Wing
9. Y4
10. Hex6X
11. OctoX8
12. OctoFlat+

13. OctoFlatX
14. Aeroplane
15. Heli
16. Heli
17. V-Tail4
18. Hex6+

----force-gps-heading, -g

Uses the GPS ground course for vehicle orientation (instead of compass). For vehicles without a compass (e.g. fixed wing).

--force-ipv4, -4

Forces IPv4 (for dual stack systems without full IPv6 resolvers).

`replay_bbox_ltm.rb` expects the ground station to poll it (this is normal mwp[15] and ezgui behaviour; both these applications will poll using MultiWii / Cleanflight identification messages when first invoked).

Typically, when using the ground station with a flight controller, one first powers up the FC, thus enabling its communications port then connects from the ground station: `replay_bbox_ltm.rb` emulates this behaviour. `replay_bbox_ltm.rb` provides basic INAV identification messages to the ground station before broadcasting the blackbox file as LTM messages.

It is also possible to bypass the polling expectation for serial devices and when invoked as a UDP listener. In these circumstances, on startup, `replay_bbox_ltm.rb` displays the message:

```
Waiting for GS to start (RETURN to continue) :
```

If you press return before any poll from the ground station, `replay_bbox_ltm.rb` will continue; this may be acceptable for serial connections, however it will cause the application to exit for UDP, as it does not know the address of the client system. This is intended to support any client such as an OSD that does not initially poll.

6.2. Example Command lines

```
# replay bbox log, index 2, flying wing, UDP listener
./replay_bbox_ltm.rb -i 2 -t 8 -u :3000 LOG0042.TXT
# replay bbox log, index 1 (default), quadX (default), USB (for ezgui/BT)
./replay_bbox_ltm.rb -s /dev/ttyUSB0 LOG0042.TXT
```

6.3. Connection examples

mwp -a -4 -s udp://host:3456	replay_bbox_ltm.rb -4 -u :3456	Emulates real world, mwp connects to "FC".
mwp -4 -s udp://:3456	replay_bbox_ltm.rb -4 -u host:3456	Invoke mwp first

[15] https://github.com/stronnag/mwptools

ezgui, connect to BT device xx:xx:xx:xx:xx:xx	replay_bbox_ltm.rb -s /dev/ttyUSB0	Assumes hardware USB / BT "bridge", start replay_bbox_ltm.rb first
mwp -s /dev/rfcomm0	replay_bbox_ltm.rb -s /dev/ttyUSB0	Assumes hardware USB / BT "bridge", start replay_bbox_ltm.rb first