# ENCE361 Heli Emulator

Draft: P HOF 15/7/2019

A new helicopter emulator is now available for you to debug your code and your signals. Please note that the emulator is in an early development stage.

## Wiring

Use this table to connect your Tiva board to the Nucleo emulator board:

|  |  |  |
| --- | --- | --- |
| Tiva Board | Wire\* | Nucleo Board |
| Main Rotor PWM: PC5 (out) | Yellow | D9 (in) |
| Tail Rotor PWM: PF1 (out) | Green | D8 (in) |
| Phase A: PB0 (in) | Blue | D7 (out) |
| Phase B: PB1 (in) | Purple | D6 (out) |
| Quadrature REF: PC4 (in) | Grey | D2 (out) |
| Altitude: PE4 (in) | White | A2 (out) |

\*Please do not change the wiring attached to the Nucleo board. If a wire breaks return the board to a technician for repair.

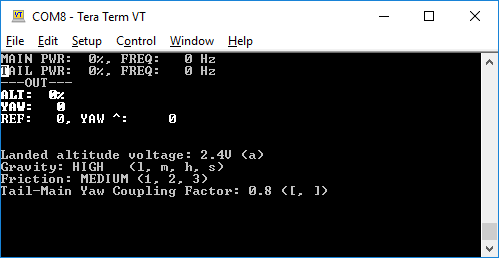
## Using the Emulator

The emulator uses VT100 commands and has only been tested with Tera Term. It is recommended to use Tera Term, which is installed in the ESL and ELEC LAB.

After opening the serial port, STMicroelectronics ST-Link2 Virtual Com Port, go to menu [Setup > Serial Port] and apply these settings:

115200baud, 8N1 (8 data bits, parity: none, 1 stop bit, flow control none)

Start the emulator by typing ‘s’ into the empty terminal. ‘s’ for start. The following screen should now be visible:



The actual screen may differ as the program is still under development.

Description:

* Line 1 is your main rotor duty cycle and PWM frequency.
* Line 2 is your tail rotor duty cycle and PWM frequency.
* Line 4 displays the altitude in percentage.
* Line 5 displays the yaw angle in degrees.
* Line 6 displays the REF count, the number of edges of the quadrature signal from the reference slot. The disc has 112 slots, defined internally as 4 \* 112 before it wraps around. Divide the REF count by 4 to get the number of slots.
* YAW ^: is the internal yaw delta value and here for debugging purposes of the emulator. A value alternating between plus and minus through 0 means your control algorithm is holding the helicopter in its yaw position. Also refer to the angle.

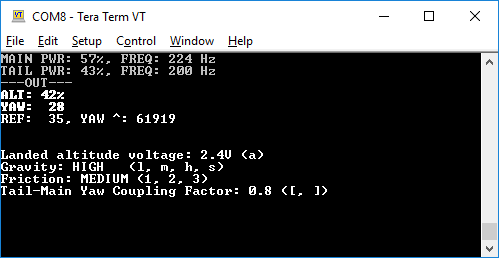
Press the keys in the parenthesis to change parameters in inflight. This allows you to test your control loop under various conditions. The defaults are probably a good starting point.

The altitude voltage should be changed while the helicopter is landed. This is to simulate the voltage variations from rig to rig. Changing these values allows you to test your calibration routine. While flying, the altitude voltage decreases up to 1V. A decrease of 1V equals to 100% height.

The emulator allows you to take off with the main rotor, completely ignoring yaw. The virtual helicopter will spin but it will not affect the lift. This allows you to work on the altitude control algorithm independently from yaw.

For the yaw control algorithm you will need to use both the main rotor and tail rotor.

Here we see the virtual helicopter flying at an altitude of 40% and with a yaw angle of 30 degrees:



The first two lines turn yellow if you exceed the maximum duty cycle (~80% for the main rotor, ~70% for the tail rotor). Unlike the real rig, the emulator will not shut down the virtual helicopter. On the real rig, the protection circuit will kick-in in ~5 seconds if those values are exceeded.

The PWM frequency should be kept between 80 to 400Hz. Most of the testing has been done at 200Hz and lower.

You can reset the emulator at any time using the black reset button on the Nucleo board. The serial connection will remain but you will need to press ‘s’ to start the emulator again.

## Testing the Wiring & Emulator

I recommend running a small test program each time when you start using the emulator to ensure that the wiring is correct and that the emulator or wires have not been damaged.

TIP: Remove the Orbit display from the Tiva if you want to use an oscilloscope. You can then hook the probes on the appropriate pins. Use the serial port to output data of interest to a terminal.

Applying a low duty cycle PWM to either the tail or main rotor will spin the virtual helicopter and produce a quadrature signal you can read with the Tiva or watch on an oscilloscope. (Using the ‘Time Mode Roll’ on the oscilloscope makes the slow changing quadrature signals easier to observe).

Use the information in the terminal window to see how much the virtual helicopter has turned.

Applying a large duty cycle (~60%) on the main rotor should give you a lift and reduce the altitude voltage. Alternatively, you could just cycle through the various output voltages with the ‘a’ key and see if your program samples them correctly.

## Cheating the Emulator

No attempt has been made to randomise values. For example applying the right amount of duty cycle to both the main and tail rotor would allow you to take-off straight without using a control loop. Trying to ‘time’ things may work on the emulator but will fail on the real rigs.

Test your control loop under various conditions by changing the parameters.

## Known limitations:

* The altitude output voltage is linear as opposed to the output of the real sensor:

<http://www.sharp-world.com/products/device/lineup/data/pdf/datasheet/gp2y0a41sk_e.pdf>

* The simulation model is not modelled after the helicopter.
* The rate of change of any signal coming from the emulator may differ greatly from the real thing.
* ~~The number of slots between two measurements of the reference signal have not been verified yet.~~
* The virtual reference slot width differs from the reference slot on the rigs.
* The terminal does not allow fast update rates, making rapid changes difficult to spot.

However:

The emulator works quite well with the original Heli Demo program, which is also the test program that runs on the real rigs.

## Control Loop

Assume that the tuning of the control loop on the real helicopter rigs will be much different from the emulator. Therefore, once you have something running more or less satisfactory on the emulator move on to the real rigs.

Your feedback on how tough it was to move from the emulator to the rigs would be welcomed.

## Using VT100 Commands

If you want, you can use VT100 commands in your own code. The sequence is always ESC + COMMAND:

Example:

//VT100 commands

#define VT100\_ESC 0x1B

#define VT100\_HOME "[H"

#define VT100\_CLS "[2J"

#define VT100\_CLR "[K"

#define VT100\_FG\_YELLOW "[33m"

#define VT100\_FG\_WHITE "[37m"

#define VT100\_BG\_BLACK "[40m"

#define VT100\_BG\_MAGENTA "[45m"

In your code:

printf("%c%s", VT100\_ESC, VT100\_HOME);

## Example Test Setup

Note: You will have to plug the wires directly into the Tiva board.

