

# Processor in the Loop Simulation Guide

## UAV Research Group

### University of Minnesota, Twin Cities

This document describes how to run the UMN UAV Processor in the Loop (PIL) simulation whose main goal is to verify the proper operation of the flight software before flight testing. This is achieved by running the flight computer in the loop with the nonlinear UAV simulation. A schematic of the current setup is presented as follows:

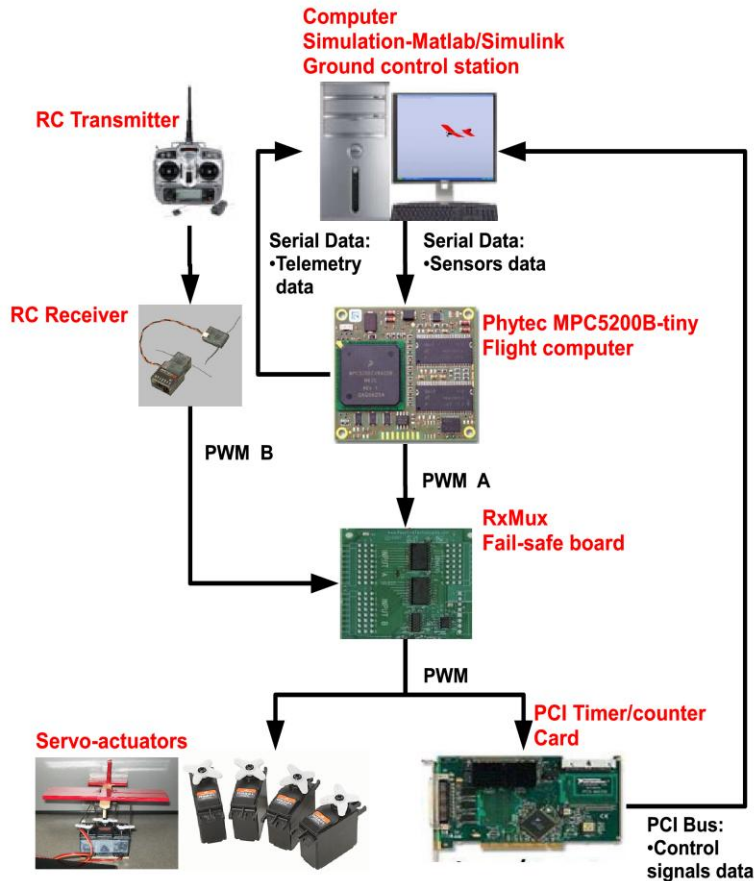


Figure 1. Processor in the loop layout

## Overall System Description:

The connection of the overall system is depicted in Figure 1. The simulation computer simulates the UAV dynamics and transmits sensor values through the serial port of the computer to the flight computer (MPC5200b-tiny). The flight computer reads the sensor inputs and generates a set of actuator commands (elevator, aileron, rudder and throttle). These commands are sent back to the simulation computer via the serial port. The simulation model is executed in real time on Matlab/Simulink using Real Time Windows Target. The hardware setup is shown in Figure 2.

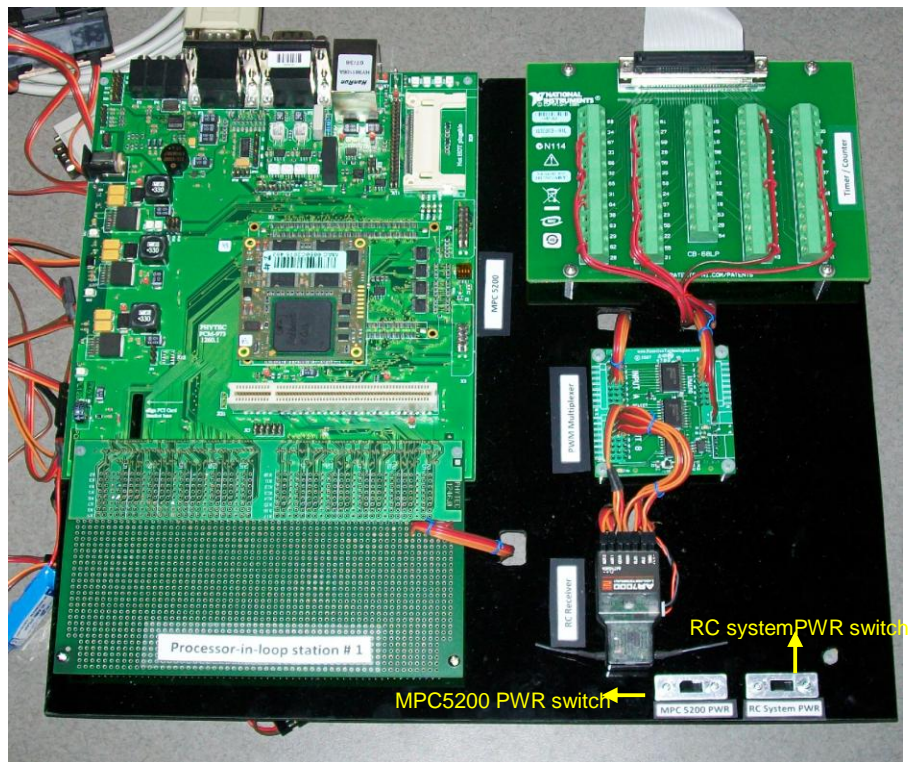


Figure 2. Hardware setup

## Components Description

**Flight computer and development board:** The Phytex-MPC5200b-tiny is a single board computer where the control, navigation and data acquisition algorithms are executed. Figure 3 shows this device:



Figure 3. phyCORE-MPC5200B-tiny System on module

This board is plugged into the daughter board of the UAV during flight testing. For the processor in the loop simulation, the board is mounted on the development board shown in Figure 4.

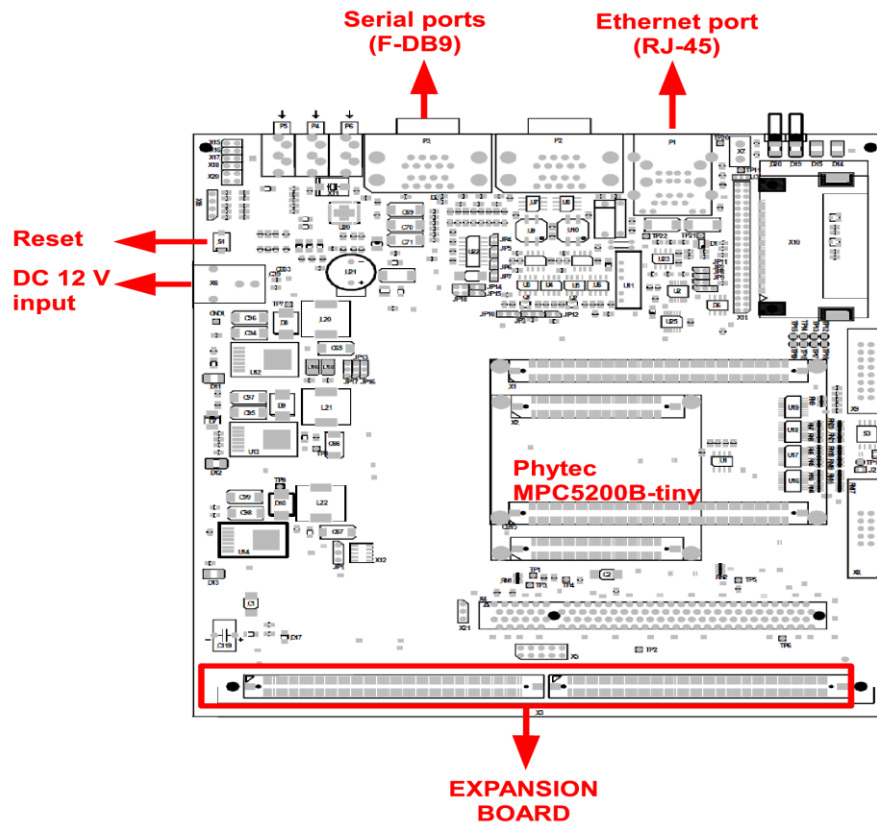


Figure 4. Development board for MPC5200B-tiny

The development board connects all the input and output ports of the MPC5200b-tiny to external connectors and hardware interfaces.

**Simulation computer:** This computer is used to run the simulation software and hosts the development tools required to compile and load the flight code onto the embedded system.

#### System interconnection:

A list of hardware interconnections is given as follows:

- Serial port 1 of the development board (Top Female-DB9 port) connected to serial port 1 of the computer (COM1)
- Serial port 2 of the development board (Bottom Female-DB9 port) connected to serial port 2 of the computer (COM2).
- Ethernet cable connected to either the network or the computer. **Note: If the development board is connected directly to the computer a crossover Ethernet cable is necessary.** The network option is recommended.

## Software Description:

#### Simulation Software:

The following software has been installed in the computers located in the UAV Laboratory.

**Operating System:** 32-bit Windows is required for compatibility with Real-Time Windows Target.

**Matlab/Simulink:** 32-bit Matlab/Simulink release 2010a is required to run the UMN UAV simulation. See “UAV\_sim\_quickstart.pdf” for a complete list of toolbox requirements.

**FlightGear:** Shows the simulation outputs using a 3D model of an aircraft.

**OpenUGS Ground Station:** Utilized to display navigation solutions, states and status computed by the flight computer. Furthermore, it can be used as a serial terminal to access the embedded system (flight computer).

## **Development Software:**

The following software has been installed in the computers located in the UAV Laboratory:

**TFTP server:** Tftp-32 software is used to transfer data between the PC and the flight computer through the network by using TFTP protocol

**Terminal:** Either Hyperterminal or any other serial terminal program can be used to communicate with the embedded system. To avoid the use of several programs during the simulation and software development, the OpenUGS ground station enables serial communication.

**Cygwin:** A GNU/Linux emulator used to compile the flight software for the MPC5200 flight computer.

**eCos (Embedded configurable operating system) version 3.0:** The eCos development tools for power-pc as well as the libraries and utilities for the MPC5200 single board computer must be installed on the development computer in order to compile the flight software.

## **Procedure for Running PIL Simulation**

Follow the steps below to run the processor in the loop simulation:

### **Running the simulation models:**

Go to the folder *Simulation/PIL\_Sim* and execute the Matlab script “setup.m” to start the simulation and external programs (Ground Station and FlightGear). This script sets the aircraft configuration and trim condition by loading two files from the *Libraries* directory, ‘UAV\_modelconfig.mat’ and ‘UAV\_trimcondition.mat’. The sample time of the simulation and the flight software is set in this script, as is the value of the time delay in the flight control loop. The RTW target is built and connected. At this point, the simulation is ready to run.

### **Loading the embedded program:**

#### **1. Compile the flight code:**

- Edit the file *Software/FlightCode/Makefile*. Two changes are needed: first, modify the “SETUP\_MODE” variable (line 28 ) to use “PIL\_SIM”.

```
##### OPERATING MODE SELECTION #####
# Uncomment the appropriate line: "PIL_SIM" for Processor in
# the Loop Simulation, "FLIGHT_BUILD" for flight testing

SETUP_MODE=PIL_SIM
#SETUP_MODE=FLIGHT_BUILD
```

Second, change the selected flight controller:. To specify which controller will be compiled modify the “Flight Control Law” section, which is as follows:

```
##### FLIGHT CONTROL LAW #####
# Point to the desired flight control law code here
CONTROL = "control/student_control.c"
#CONTROL = "control/baseline_control.c"
#CONTROL = "control/lqr_control.c"
#CONTROL = "control/empty_control.c"
#CONTROL = "control/mimo_control4.c"
```

With this code, the controller in “Software/FlightCode/control/student\_control.c” will be compiled. This code is the same code used in the Software in the Loop simulation (SIL).

- Start Cygwin, and change to the directory where the source code is located. It can be done by using the “cd” command. For example “cd <base\_dir>/Software/FlightCode“
- Type “make” in the Cygwin terminal. If no compilation errors are displayed, the binary file called “output” will be generated. This file is an executable which will be loaded onto the MPC5200b-tiny - flight computer. Recall that this file must be placed in the folder selected for the TFTP server operation. To remove all the files generated from the compilation type: *make clean*.

```
~/FlightCode
/opt/ecos/ecos_libs/include/cyg/libc/time/time.inl:441: warning: `cyg_libc_time_year_is_leap' is static but used in inline function `gmtime_r' which is not static
/opt/ecos/ecos_libs/include/cyg/libc/time/time.inl: In function `mktime':
/opt/ecos/ecos_libs/include/cyg/libc/time/time.inl:649: warning: `cyg_libc_time_year_is_leap' is static but used in inline function `mktime' which is not static
/opt/ecos/ecos_libs/include/cyg/libc/time/time.inl:669: warning: `cyg_libc_time_year_is_leap' is static but used in inline function `mktime' which is not static
/opt/ecos/ecos_libs/include/cyg/libc/time/time.inl:673: warning: `cyg_libc_time_year_is_leap' is static but used in inline function `mktime' which is not static

In file included from L_avionics.c:45:
control/control_manager.c: In function `control_uav':
control/control_manager.c:82: warning: unused variable `dr_last'
control/control_manager.c:81: warning: unused variable `da_last'
control/control_manager.c:80: warning: unused variable `de_last'
L_avionics.c:274:1: warning: multi-line comment
##### UMN UAV software #####
Aerospace Engineering and Mechanics, University of Minnesota
Contact E -> descobar@aem.umn.edu

mpc5200@ultrastick ~/FlightCode
$
```

Figure 6. Cygwin terminal

## 2. Load executable file onto flight computer:

- Start the TFTP server tftpd-32 from the desktop icon or the programs menu. Then, select the folder where the MPC5200 binary files will be located. Usually this is the same source code directory (*Software/FlightCode*).

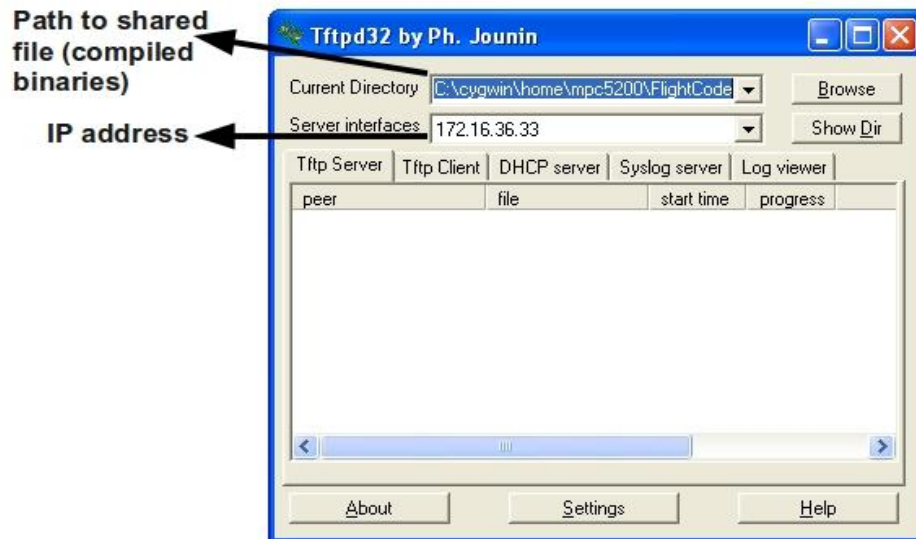


Figure 5. Tftpd32 window

- If not started, run the OpenUGS ground station software from the desktop icon or the executable file.
- Click on the “Settings” tab, then make sure the following parameters are in place
  - Port name: COM2
  - Bit rate: 115200
  - Data bits: 8
  - Stop bits: 1
  - Parity: none
  - Flow control: none
  - Interface: Terminal
  - Mode: ASCII
- Click on *Options – Input – Serial port*. The console at the bottom should display the message “Serial communication initialized !”. A screen shot of the Ground Station program is shown in Figure 7.
- Turn on all of MPC5200B electronics by following the steps below:
  - Turn on the power supply
  - Turn on the switch labeled as “MPC5200 PWR”
- Once the flight computer is on, the serial terminal should display *RedBoot>*. For loading the binary file located in the host computer type the command as follows:

*load -m tftp -h <pc\_ip\_address> output*

*<pc\_ip\_address>* is the IP address of the computer where the binary file is located and the TFTP server is running. The ip address can be found out in the Tftpd32 window as shown in Figure 5. After executing the previous line the terminal should display something similar



to:

*\*\*\*Loadable segment: 0, Len=479368*

*Entry point: 0x00100100, address range: 0x00100000-0x00175088*

*RedBoot>*

Once the binary file is loaded in RAM memory, you are be able to run the embedded program by typing *go*.

If you want to save the binary file in a non volatile memory (flash memory) type the following command.

*fis create output*

To load this program from flash memory type:

*fis load output*

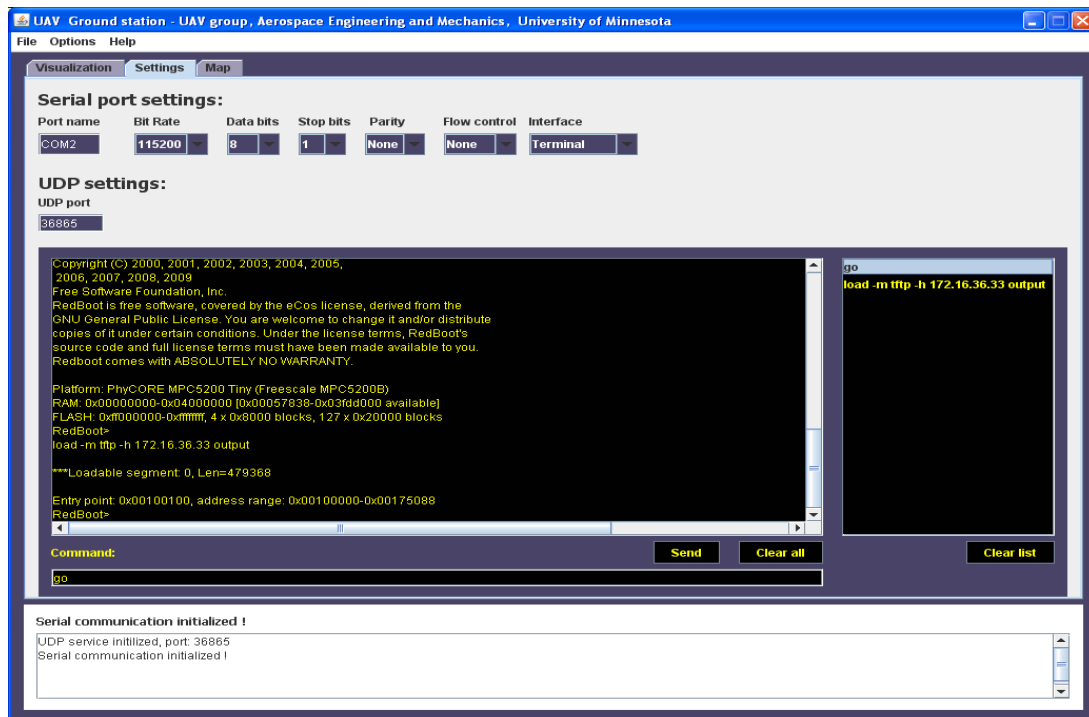


Figure 7. OpenUGS Ground station terminal

- Before running the flight computer program, make sure you have started the simulation. While in 'manual mode', trim inputs will be used in the simulation.
- To start the flight computer program type the command *go* in the serial terminal. To

visualize the flight computer program output in the ground station change the interface box from Terminal to visualization, and click on the Visualization tab.

- Verify that the attitude solution given by the ground station display is close to the solution given by the Simulation that is shown through FlightGear.
- Change to autopilot mode by clicking the switch in the top level of the UAV\_PIL sim.
- At this point, the autopilot should take control of the UAV model, and you can see the behavior of the system by looking at the Ground station and FlightGear. The Ground station displays the state solutions from the flight computer while FlightGear displays the simulation solutions. Figure 8 shows the Ground station - visualization tab. Every time the autopilot mode is switched, the Flight computer performs the control sequence programmed in the flight computer.
- Stop the simulation by doing click on the stop button of the Simulink model. Data from the executed simulation can be saved by running the Matlab script *save\_pil\_data.m*. To plot this data use the script *plot\_pil.m* function which receives file names of data saved after simulations.

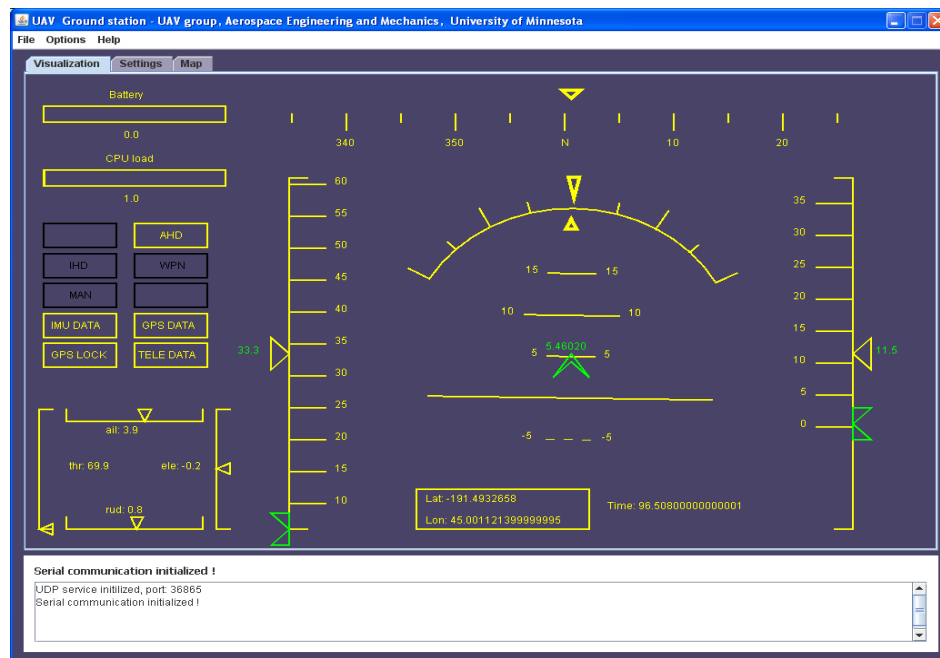


Figure 8. OpenUGS Ground station – Visualization tab