GauchoHawk
End of Quarter Report

Vikram Sastry, Yesh Ramesh, Kurt Madland, Jack Zang, Richard Young, and Shawn Zhang
INTRODUCTION

What is the Pixhawk?
Pixhawk is an independent, open-hardware project aiming at providing high-end autopilot hardware and open-source flight control software to the academic, hobby, and industrial communities at low costs and high availability.

Pixhawk Pitfalls
1. No high quality inertial measurement unit (IMU) for precision flight control.
2. No magnetometer for extended periods of inertial guidance and state estimation.
3. No Ethernet interface for high speed connection to Linux Computers or IP data links.
5. No precise time source for time-stamping sensor data in absence of GPS.
Pixhawk + Improved Hardware + a few Gauchos (and a little help from Aerovironment) = GauchoHawk
MEET THE TEAM

Vikram Sastry  Team Leader, System Design
Yesh Ramesh  Software Porting, Digital Design
Kurt Madland  Analog & Digital Design, System Design
Jack Zang  Analog & Digital Design, System Design
Shawn Zhang  Software Porting, Interface Design
Richard Young  Software Porting, Interface Design
PROJECT DESCRIPTION

Shield Hardware Development

We propose to use the STMicroelectronics Nucleo-STM32F767ZI board in conjunction with a “Shield” daughterboard to introduce a high quality IMU, Barometer/Altimeter, Magnetometer, Airspeed Sensor, RTK GPS, XBEE Radio, SBUS, CAN, Servo/Motor PWM, and Interrupt/GPIO capability.

PX4 Software Implementation

We propose to port the open source PX4 middleware, flight control stack, and NuttX OS to the Nucleo board and develop a new sensor driver suite to support the full expanded functionality of the sensor shield daughterboard.
PROJECT APPLICATIONS

1. **Agriculture**
   - Identifying irrigation deficiencies, inspect crops for stress, etc.

2. **Military**
   - Training, tactical UAV missions, missile guidance, etc.

3. **Online Retail**
   - Reduce delivery overhead, etc.

4. **Many more...**
System Design Overview
Components

- **List of Components**
  - **Microcontroller Unit** - STM32F767ZI
  - **Wireless Network** - XBEE 900HP
  - **Pressure** - LPS22HB
  - **Pressure & Temperature** - MS4515DO
  - **Navigation** - Neo-M8P
  - **Magnetometer** - QMC5883L
  - **Motion Tracking** - MPU-9250 IMU
PCB Schematic
PCB Layout
PCB
Power Distribution

Supplying 24V to our PCB
- 5.5 V analog (servos), 5V digital, and 3.3V digital
- Servos require 5.5V to VDD

<table>
<thead>
<tr>
<th>Component</th>
<th>Supply Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Servos (8)</td>
<td>5.5 V</td>
</tr>
<tr>
<td>Peripherals (7)</td>
<td>3.3 V</td>
</tr>
<tr>
<td>Servos Signal (8)</td>
<td>5 V</td>
</tr>
</tbody>
</table>
Software
Port PX4 to Nucleo F7

- Successfully able to flash and run the PX4 firmware on our board
- Use STM32CubeProgrammer application to load "elf" file
- Use mavproxy via python’s pip installer to connect to appropriate serial port
- Run:
  - "module load nsh"
  - "nsh start"
- Above commands will run NuttX Shell
  - Allows us to interface with the board and pass commands/read data

Commands

- PWM currently working
- Some driver specific commands are still needed
**Software**

- **Drivers**
  - We have code for most of the drivers, but most are untested
  - Currently working on setting up tests for each component and driver pair

- **Prototype Skeleton Code**
  - Our sponsor sent us some skeleton code fleshing out some of the drivers we don’t have configured yet
  - Caveat: They’re not formatted for PX4 usage, so some adaptation is needed
  - Shouts out to the Phil Dawg
- **PCB Assembly may be faulty**
  - Requires us to re-spin the PCBs, which is a time consuming process
  - Could be problematic to debug given state of drivers
- **Drivers not all functioning, or even tested**
  - We have an assortment of driver code from various sources:
    - Some from the official PX4 Repo
    - Some from our sponsor’s skeleton code
  - Skeleton code requires adaptation to function with the PX4 flight stack
  - Above could lead to consistency issues
Conclusion

- **Goals for Spring 2018:**
  - All modules have a corresponding driver
  - All drivers to be integrated and communicating with NuttX
    - Including driver specific commands accessed via serial port
  - Work through test cases for each module on the board
  - Ensure that Auto Pilot is correctly functioning
  - Integrate servos and Ground Control communication
  - Put it on an airframe
Thank you to Dr. Isukapalli, Caio, and Phil Tokumaru for their patience and guidance.