Hackflight: A Simple Software Ecosystem for Miniature Aerial Vehicles

Simon D. Levy
Julio Hidalgo Lopez
Matt Lubas
Alfredo Rwagaju

Washington & Lee University

Intercollegiate Dronefest
18 August 2016
Motivation #1:
What I Cannot Build, I Do Not Understand

R. Feynman (1918-1988)
Keep It Simple, Stupid!

- PX4 Firmware: > 100K lines of code
- Cleanflight: > 50K
Motivation #2:
I’m not gonna pay a lot for this muffler flight controller!

$200  $15
Hackflight Genealogy

MultiWii

Baseflight (~14K l.o.c.)

Hackflight

| Cleanflight

| Raceflight

| Betaflight
Hackflight Genealogy

Baseflight

Hackflight

Platform-Independent Firmware

BreezySTM32

(~4200 lines of C)

(~1500 lines of C++)
<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
<th>Days Ago</th>
</tr>
</thead>
<tbody>
<tr>
<td>baro.cpp</td>
<td>Made all Board methods static</td>
<td>7</td>
</tr>
<tr>
<td>baro.hpp</td>
<td>Made all Board methods static</td>
<td>7</td>
</tr>
<tr>
<td>board.hpp</td>
<td>Add support for displaying aux-switch status</td>
<td>3</td>
</tr>
<tr>
<td>crossplatform.h</td>
<td>Debugging console working with Windows</td>
<td>30</td>
</tr>
<tr>
<td>filters.cpp</td>
<td>Add Hover class, module for filters</td>
<td>30</td>
</tr>
<tr>
<td>filters.hpp</td>
<td>Add Hover class, module for filters</td>
<td>30</td>
</tr>
<tr>
<td>hover.cpp</td>
<td>Fixed typo in comment</td>
<td>13</td>
</tr>
<tr>
<td>hover.hpp</td>
<td>Built Windows release</td>
<td>13</td>
</tr>
<tr>
<td>imu.cpp</td>
<td>Computing vario, alt-hold PID Dust</td>
<td>15</td>
</tr>
<tr>
<td>imu.hpp</td>
<td>Computing vario, alt-hold PID Dust</td>
<td>15</td>
</tr>
<tr>
<td>mixer.cpp</td>
<td>Made all Board methods static</td>
<td>7</td>
</tr>
<tr>
<td>mixer.hpp</td>
<td>Made all Board methods static</td>
<td>7</td>
</tr>
<tr>
<td>msp.cpp</td>
<td>Renamed Navigate to Hover; removed alt-hold implementation</td>
<td>19</td>
</tr>
<tr>
<td>msp.hpp</td>
<td>Renamed Navigate to Hover; removed alt-hold implementation</td>
<td>19</td>
</tr>
<tr>
<td>mw.cpp</td>
<td>Renamed Navigate to Hover; removed alt-hold implementation</td>
<td>19</td>
</tr>
<tr>
<td>mw.hpp</td>
<td>Renamed Navigate to Hover; removed alt-hold implementation</td>
<td>19</td>
</tr>
<tr>
<td>rc.cpp</td>
<td>Made all Board methods static</td>
<td>7</td>
</tr>
<tr>
<td>rc.hpp</td>
<td>Made all Board methods static</td>
<td>7</td>
</tr>
<tr>
<td>sonars.cpp</td>
<td>Better variable names</td>
<td>6</td>
</tr>
<tr>
<td>sonars.hpp</td>
<td>Begin supporting four horizontal sonars</td>
<td>6</td>
</tr>
<tr>
<td>stabilize.cpp</td>
<td>More cleanup</td>
<td>22</td>
</tr>
<tr>
<td>stabilize.hpp</td>
<td>More cleanup</td>
<td>22</td>
</tr>
</tbody>
</table>
BreezySTM32: An Arduino-like API for STM32 flight controllers

```c
#include <breezystm32.h>

static bool sonar_present;
mb1242_t mb1242;

void setup(void)
{
    i2cInit(I2CDEV_2);
    delay(500);
    sonar_present = mb1242_init(&mb1242, 0);  // Use default address
}

void loop(void)
{
    if(sonar_present)
        printf("%d\n", mb1242_poll(&mb1242));
    else
        printf("no sonar\n");
    delay(100);
}
```
/*
 * ms4925.c : Airspeed Measurement Values
 *
 * Copyright (C) 2016 James Jackson
 *
 * This file is part of BreezySTM32.
 *
 * BreezySTM32 is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * BreezySTM32 is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with BreezySTM32. If not, see <http://www.gnu.org/licenses/>.
 */

#include "breezystm32.h"

void setup(void)
{
    delay(500);
    i2cInit(I2CDEV_2);
}

int16_t velocity;
int16_t temp;

void loop(void)
{
    if( ms4925_detect() )
    {
        ms4925_read(&velocity, &temp);
        printf("velocity = %d, temp = %d\n", velocity, temp);
    }
    else
    {
        printf("no airspeed\n");
    }
    delay(10);
}
Python Ground Control Station

![Python Ground Control Station Image]
MSPPG: Parser (Code) Generator for MultiWii Serial Protocol

```json
{
  "RC": {
    "ID": 100,
    "c0": "short",
    "c1": "short",
    "c2": "short",
    "c3": "short",
    "c4": "short",
    "c5": "short",
    "c6": "short",
    "c7": "short",
    "c8": "short"
  },
  "ATTITUDE": {
    "ID": 100,
    "comment": "angles should be divided by 10",
    "roll": "short",
    "pitch": "short",
    "yaw": "short"
  },
  "ALTITUDE": {
    "ID": 100,
    "altitude": "int",
    "vario": "short"
  },
  "SONAR": {
    "ID": 100,
    "comment": "four horizontal-facing sonars",
    "back": "short",
    "front": "short",
    "left": "short",
    "right": "short"
  },
  "SET_RAW_RC": {
    "ID": 200,
    "c0": "short",
    "c1": "short",
    "c2": "short",
    "c3": "short",
    "c4": "short",
    "c5": "short",
    "c6": "short",
    "c7": "short",
    "c8": "short",
    "c9": "short",
    "c10": "short"
  },
  "SET_HEAD": {
    "ID": 205,
    "head": "short"
  },
  "set_motor": {
    "ID": 210,
    "c1": "short",
    "c2": "short",
    "c3": "short",
    "c4": "short"
  }
}
```
HackflightSim
```cpp
class Board {

public:

    // your implementation should support these methods

    static void    init(uint32_t & imu0optimeUsec, uint32_t & calibratingGyroMsec);
    static bool    baroInit(void);
    static void    baroUpdate(void);
    static int32_t  baroGetPressure(void);
    static void    checkReboot(bool pendReboot);
    static void    delayMILLISECONDS(uint32_t mseconds);
    static uint32_t  getMicros();
    static void    imuInit(uint16_t & acc1G, float & gyroScale);
    static void    imuRead(int16_t accADC[3], int16_t gyroADC[3]);
    static void    ledGreenOFF(void);
    static void    ledGreenON(void);
    static void    ledGreenToggle(void);
    static void    ledRedOFF(void);
    static void    ledRedON(void);
    static void    ledRedToggle(void);
    static uint16_t  readPWM(uint8_t chan);
    static void    reboot(void);
    static uint8_t  serialAvailableBytes(void);
    static uint8_t  serialReadByte(void);
    static void    serialWriteByte(uint8_t c);
    static void    showArmedStatus(bool armed);
    static void    showAuxStatus(uint8_t status);
    static bool    sonarInit(uint8_t index);
    static void    sonarUpdate(uint8_t index);
    static uint16_t    sonarGetDistance(uint8_t index);
    static void    writeMotor(uint8_t index, uint16_t value);

}; // class Board
```
Missions ("Scenes")

Python Computer Vision via "companion board"
Sonar-Based SLAM
(Simultaneous Localization And Mapping)
Personnel & Acknowledgements

Prof. Simon D. Levy

Bipeen Acharya ‘15  Shannon Nollett ‘15  Fred Gisa ‘16

Commonwealth Research Commercialization Fund (2014)

Summer Research Scholars Program
Gerry Lenfest Summer Research Grant
Advanced Research Cohort Program

Bert Wagner
Advanced Aerials, Inc.