8/2/2009

|  |
| --- |
| Package 0.11 | David Gitz and Company |



|  |  |
| --- | --- |
| David Gitz | Remote Control Unit |

Contents

[Package Comparison: 2](#_Toc238567316)

[Important Notes: 3](#_Toc238567317)

[Introduction: 3](#_Toc238567318)

[Summary: 4](#_Toc238567319)

[Specifications: 4](#_Toc238567320)

[Revision Specific Notes: 4](#_Toc238567321)

[Subsystem Specific Documentation 5](#_Toc238567322)

[System Documentation: 8](#_Toc238567323)

**Figure Listings:**

Figure 1. Functional Diagram  
Figure 2. Physical Diagram  
Figure 3. Propeller Cog Usage

**Table Listings:**

Table 1. Package Comparison  
Table 2. RCU Specifications  
Table 2. Network Test Fail Conditions.

# Package Comparison:

Refer to Table 1.  
Package .1

* Radio’s work in AT Mode, i.e. network only effectively supports 2 Nodes. Program on Micro-Controller is as simple as possible.
* System Communications Protocol is designed by Author.
* Radio and LCD operate on same Processor. This provides highest expandability.

Package .11

* Includes all Elements of Package .1, Plus:
* GPS Sensor, functionality to add a “Home” Waypoint to Vehicle.

Package .2

* Radio’s work in API Mode. This allows many nodes to exist on network. This complicates the Micro-Controller program slightly.
* System Communications Protocol is designed by Author.
* Radio and LCD operate on different Processors. This reduces expandability.

Package .3

* Radio’s work in API Mode. This allows many nodes to exist on network. This complicates the Micro-Controller program slightly.
* System supports JAUS Protocol. This greatly complicates Micro-Controller programming.
* Radio and LCD operate on different Processors. This reduces expandability.

**Table 1. Package Comparison**

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | Radio Mode | Radio/LCD Processor Usage | Communications Protocol |
| .1 | AT | 1 | Author-derived |
| .2 | API | 2 | Author-derived |
| .3 | API | 2 | JAUS |

# Important Notes:

As this is an unfinished work, an attempt has been made to provide subsequent releases of fairly compatible and more advanced capabilities.

# Introduction:

This project is an attempt at producing a open source Remote Control Unit (RCU) that is capable of controlling robots, and easily modified to fit a user’s platform requirements. This project is based entirely on open source code, schematics, protocols and standards.

The RCU is also being developed concurrently using the Joint Architecture for Unmanned Systems (JAUS) Protocol that was developed by the Department of Defense for the purposes of Command and Control (C2) of differing, advanced and distributed robotic platforms. Due to the complexity of this Protocol, RCU Version .1 does not implement the JAUS Protocol and instead a custom Protocol as was developed previously by the Author has been implemented instead, with some degraded mission features.

RCU Version .1 was programmed specifically for the control of a Quad-Rotor Unmanned Aircraft System’s (UAS) and specific modes the Vehicle may enter. However, as has been stated previously, the hardware on the RCU is not dependent on the UAS selection and the software may easily be adapted to fit the needs of the selected UAS, if different from the original design.

The RCU incorporates a Propeller Micro-Controller, a device that incorporates 8 Processors that are capable of performing multiple tasks concurrently. This allows the RCU to perform many tasks dependent of each other that would normally hinder a Remote Control’s usefulness, such as communicating with a Vehicle in Real-Time and displaying User generated messages.

# Summary:

The RCU features a large quantity of User available input methods, including 4 Analog Inputs and 16 Digital Inputs (via the Xbox 360 Controller) and a LCD display and indicator LED’s, all of which may be customized and implemented as the System Engineer deems necessary.

With the RCU Version .1, the Controller joysticks function as controlling the UAS’s motor outputs, the buttons control different modes of flight the UAS may enter, and the LCD displays those modes and any error’s that are encountered that have been specifically programmed into the micro-controller.

# Specifications:

**Table 1. RCU Specifications**

|  |  |
| --- | --- |
| Communications | UART, 115200 kbps, 8 Data Bits, No Parity, 1 Stop Bit |
| Range |  |
| Life Time |  |
| User Available Memory |  |
| User Available Processor’s |  |
|  |  |
|  |  |

# Revision Specific Notes:

This documentation consists of material from Google Code SVN 35.   
  
The communication architecture is based on the XBee Transparent (Non-API) and therefore employs only a 2 node communications network. Also, the specifically engineered Communications Protocol is in use. In later work, this will be replaced (if deemed feasible) by the JAUS Communications Protocol.

Because of the XBee Transparent communications architecture employed, the Remote’s LCD and Radio are run from a multi-uart SPIN object. In later versions, they will be separated due to the API implementation strategy.­­­­

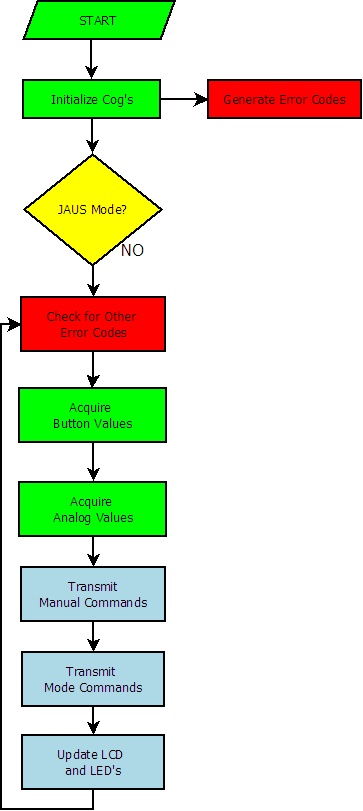
# Subsystem Specific Documentation

*Remote Control Operation*

Left Thumbstick Horizontal - Adjust Vehicle Yaw  
Left Thumbstick Vertical  - Adjust Vehicle Throttle  
Right Thumbstick Horizontal - Adjust Vehicle Roll  
Right Thumbstick Vertical - Adjust Vehicle Pitch

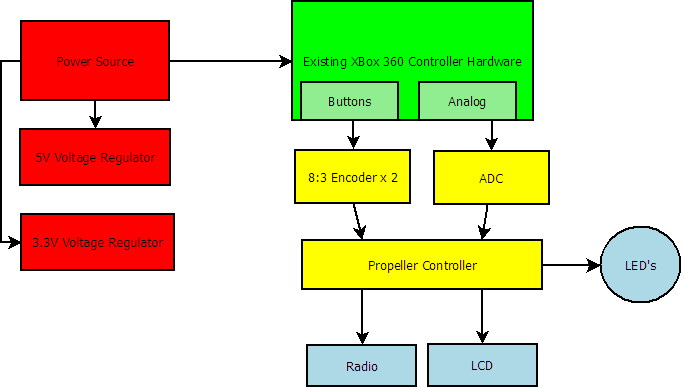
D-Pad Hat:  
UP:  Enter TAKEOFF Mode  
RIGHT:  Enter HOVER Mode  
DOWN:  Enter LAND Mode

Buttons:  
A:  Switches between Manual/Auto Mode  
B:  Kill Vehicle  
X: Go Home Mode.

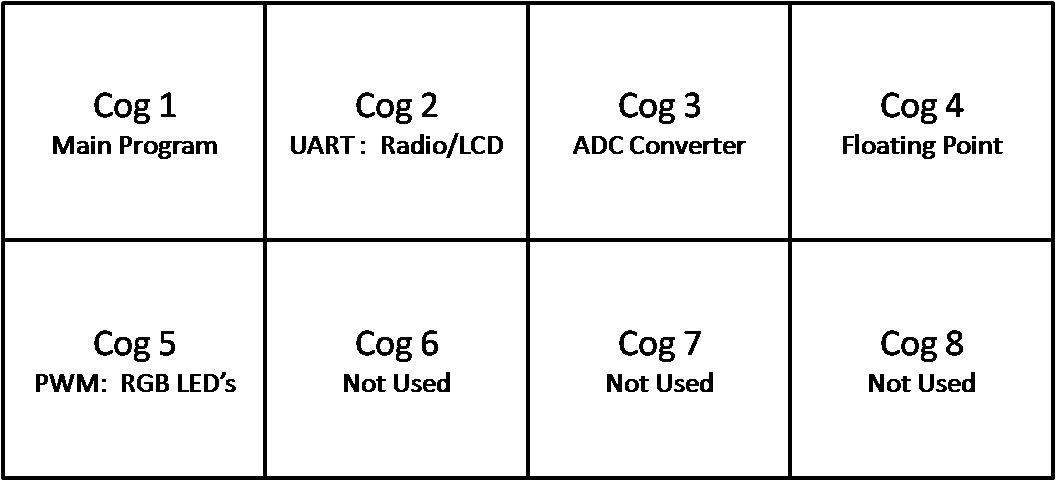


**Fig 1. Functional Diagram**

**Fig 2. Physical Diagram**



**Fig 3. Propeller Cog Usage**



# System Documentation:

Modes:  
TAKE-OFF  
HOVER  
LAND  
GO-TO-WPT  
MANUAL  
  
Definitions:  
BEARING:  The absolute angle between the Vehicle and a waypoint, where 0 Degrees is   
True North.  
HEADING:  The relative angle the Vehicle is on compared to its desired BEARING.  During  
GO-TO-WPT Mode, the HEADING (BEARING-COMPASS\_BEARING) should be optimized to be as  
close to zero as possible.  
COMPASS\_BEARING:  The absolute angle between the Vehicle and True North, assumed to be   
derived from the Compass Sensor.  
DISTANCE:  The relative distance between the Vehicle and a waypoint, in feet.  
THROTTLE:  Combined Engine Speed.  100% is defined as minimum takeoff speed.  Less than 100%   
would result in Vehicle descent.  More than 100% would result in Vehicle ascent.  THROTTLE   
mapping to actual Engine Speed can vary, based on TAKEOFF-SELFCALIBRATION Mode.  
DELTA\_ALTITUDE:  The change in altitude from the start of an operation and each iteration of it.  
Assumed to be derived from altitude sensor, although can be derived from ultrasonic sensor(s).  
PITCH:  The motion about the Vehicle's x-axis resulting in a forward or backward tilting motion.  
ROLL:  The motion about the Vehicle's y-axis resulting in a left or right tilting motion.  
YAW:  The motion about the Vehicle's z-axis resulting in the Vehicle's horizontal rotation.  
REMOVE CONTROL:  
  
Inertial Navigation Unit:  Incorporates sensors to measure PITCH, ROLL, and YAW.  Sensors used   
are gyroscopes and accelerometers.  Angles are calculated by integrating Gyroscope values and   
passing these values through a digital LPF, and by taking the inverse cosine of 2 perpendicular   
accelerometer measurements and passing through a digital HPF, then combining these 2 measurements  
for each axis in a complementary filter.  
  
Specifications:  
Distances between Waypoints (To limit the number of times of calculating parameters):  
LARGE:  
-When a waypoint is wished to be reached, the Vehicle will calculate the bearing and distance  
between itself and the waypoint.  It is assumed that the distance will be LARGE.  
-If the distance is within a smaller distance, it will be classified as MEDIUM or SHORT.    
-If the waypoint is LARGE, after this initial distance and bearing is calculated it will not be  
calculated again until a sufficient time (LARGE\_DELAY) is reached.  
MEDIUM:  
If the waypoint is a MEDIUM distance, the Vehicle will calculate the distance and bearing at   
regular intervals (MEDIUM\_DELAY) until it reaches a SHORT distance.  
SHORT:  
If the waypoint is a SHORT distance, the Vehicle will stop and either enter HOVER mode or  
enter GO-TO-WPT mode with a new waypoint.  
  
Pseudo-Code  
GO-TO-WPT Mode (Waypoint Navigation)   
A.  Recieve Waypoint (Latitude, Longitude)  
B.  Calculate DISTANCE and BEARING.  
C1. If DISTANCE is LARGE Then   
       Set PITCH to 45 deg.  
       Set YAW to 0 deg.  
       Set ROLL to 0 deg.  
       Set THROTTLE to 150 percent.  
       Wait LARGE\_DELAY while maintaining these parameters.  
       Go to B.  
C2. If DISTANCE is MEDIUM Then  
       Set PITCH to 30 deg.  
       Set YAW to 0 deg.  
       Set ROLL to 0 deg.  
       Set THROTTLE to 125 percent.  
       Wait MEDIUM\_DELAY while maintaining these parameters.  
       Go to B.  
C3. If DISTANCE is SHORT Then  
       Set PITCH to 0 deg.  
       Set YAW to 0 deg.  
       Set ROLL to 0 deg.  
       Set THROTTLE to 100 percent.  
       Set MODE to HOVER.  
       Finish.  
         
TAKEOFF Mode  
-2 Sub-Modes  
TAKEOFF-SELFCALIBRATION  
A.  Increase THROTTLE until DEL\_ALTITUDE is 5 Feet.  
      While PITCH, YAW, ROLL are Set to 0 deg.  
B.  Set 100% THROTTLE to current THROTTLE.  
C.  Enter HOVER Mode.  
TAKEOFF  
A.  Set Throttle to 100% until DEL\_ALTITUDE is 5 Feet  
      While PITCH, YAW, ROLL are Set to 0 deg.  
B.  Enter HOVER Mode.  
  
HOVER Mode  
A.  Set THROTTLE to 100%.  
B.  Maintain PITCH, YAW, ROLL at 0 deg.  
C1. If Altitude is less than 5 feet Then  
      Enter TAKEOFF Mode.  
        
LAND Mode (Now piecewise control, consider Landing Throttle a function of Altitude).  
A.  Set THROTTLE to 75%.  
B.  Maintain PITCH, YAW, ROLL at 0 deg.  
C1. If 2 feet < Altitude < 5 feet Then  
      Set THROTTLE to 50%.  
C2. If 1 foot < Altitude < 2 feet Then  
      Set THROTTLE to 25%.  
C3. If Altitude < 1 foot Then  
      Set THROTTLE to 0%.   
  
MANUAL:  
A.  Set THROTTLE, PITCH, YAW, ROLL to REMOTE CONTROL Commands.  
  
ERROR Codes  
0:          No Error.  
1 - 2000:     Vehicle SOM Error   
2001 - 4000:  Vehicle Propellor Error  
  2001:  Communications Processor started unsuccessfully.  
  2002:  PWM Processor started unsuccessfully.  
  2003:  Encoder Processor started unsuccessfully.  
  2004:  Floating Point Processor started unsuccessfully.  
  2008:  Propellor started unsuccessfully (General Error).  
4001 - 6000:  Vehicle Sensor Error  
6001 - 8000:  Vehicle Navigation Error  
8001 - 10000:  Vehicle General Error  
  8020:  Network Error (General Error).  
  
20001 - 22000:  Remote Control Error  
  20001:  Radio Processor started unsuccessfully.  
  20002:  LCD Processor started unsuccessfully.  
  20003:  Analog-To-Digital Processor started unsuccessfully.  
  20004:  Floating Point Processor started unsuccessfully.  
  20008:  Propeller started unsuccessfully (General Error).  
  20011:  Interface Not Found on Network.  
  20012:  Vehicle Not Found on Network.  
  20020:  Network Error (General Error).  
  
22001 - 24000:  Interface Error  
  22020:  Network Error (General Error).    
  
Network Test Mode  
  
Can be initiated (and should be upon powerup) by any node, and repeated for both nodes.  
Limit retry to 3 times.  
  
If any of the following conditions appear during a Network Test:

**Table 2. Network Test Fail Conditions**

|  |  |
| --- | --- |
| **Originator** | **Receiver** |
| Remote Control | No Vehicle |
| Interface | No Vehicle |
| Vehicle | No Interface or No Remote Control |

Issue a Network Error on that Originator Node.