Protocol for Nunchuk Controller (racer/robot)

Messages between Controller and Robot are sent over ESP-NOW protocol, using short binary structures defined in the Arduino code as indicated in the following sections. The MAC address of the responding robot or racer is hard encoded into the “communic.cpp” module of the software. Note that in the current version of the software, there are 3 values indicated, though only the first one (index 0) is actually used at this time.

Colorcodes

For those incoming messages that receive colorcodes, the code is a single char value, as follows:

- ‘G’ Green
- ‘Y’ Yellow
- ‘O’ Orange
- ‘R’ Red
- ‘W’ White
- ‘X’ Black
- ‘B’ Blue
- ‘C’ Cyan
- ‘P’ Purple
- ‘Z’ Medium Gray
- ‘H’ (header color; cyan)

Controller to Robot Messages

```
typedef struct {
  char msgtype;   // 'C', 'Z', 'X', 'Y', 'V', 'H', '0', '1', '2'
  int  intdata;   // note this is a 2-byte (16 bit) integer
} MessageCtoR;    // this type (structure) holds a message that goes from
                     Controller to Robot
```

All messages are 3 bytes long, with the 1st byte indicating the message type, and the following 2 bytes (Arduino 2 byte integer) holding the value associated with the message. All reported intdata are LSB first.

Message type ‘C’ :

This indicates the status of the Nunchuk ‘C’ button (the “upper” one on the Nunchuk device). The intdata value is integer 1 when button PRESS is detected, and 0 when button RELEASE is detected.

Button 2

Message type ‘Z’ :

This indicates the status of the Nunchuk ‘Z’ button (the “lower” one on the Nunchuk device). The intdata value is integer 1 when button PRESS is detected, and 0 when button RELEASE is detected.

Message type ‘0’ :

This indicates the status of the Feather ‘0’ button. The intdata value is integer 1 when button PRESS is detected, and 0 when button RELEASE is detected.

Message type ‘1’ :

This indicates the status of the Feather ‘1’ button. The intdata value is integer 1 when button PRESS is detected, and 0 when button RELEASE is detected.

Message type ‘2’ :

This indicates the status of the Feather ‘2’ button. The intdata value is integer 1 when button PRESS is detected, and 0 when button RELEASE is detected.

Message type ‘X’:

This message sends the X value of the joystick (left and right). The value in intdata is scaled such that FULL LEFT reports -255, and FULL RIGHT reports +255. The value is reported whenever it changes by 2 or more, and has a small deadband at center to reduce jitter.

Message type ‘Y’:

This message sends the Y value of the joystick (forward and back). The value in intdata is scaled such that FULL BACK reports -255, and FULL FORWARD reports +255. The value is reported whenever it changes by 2 or more, and has a small deadband at center to reduce jitter.

Message type ‘H’:

This message sends a “heartbeat’ indication to the racer / robot for the purpose of indicating that it is alive and connected. The message is sent once every 1000 mS. The intvalue is irrelevant in this message.

Message type ‘V’:

This message is currently not used, but is intended to send the binary value representing the battery voltage of the LiPo in the control box.

Robot to Controller Messages - Battery Status

```
typedef struct {
  uint8_t messagetype;
  char    colorcode;
  float   battvolts;
  float   cellvolts;
} MessageRtoC_batt;    // this type (structure) holds a message that goes from
                          Robot to Controller
```

This message is sent from the racer / robot to the controller and indicates the robot's battery voltage (and corresponding cell voltage for multi-cell batteries) for purposes of display.

- messagetype: 0 (MSG_VOLTS_E) indicates voltage of the PRIMARY battery (the one that feeds the processor and displays), 1 (MSG_VOLTS_M_ indicates the voltage of the MOTOR battery
- colorcode: character indicating color that should be used to display the passed voltages, per the table above. Generally the colorcode reflects the level of charge (ie, G/reen, Y/ellow, O/range, R/ed)
- battvolts: floating point value representing the battery voltage (ie 8.2). This is presented LSB first
- Cellvolts: floating point value representing the cell voltage (ie 3.7) for multicell batteries. Will be the same as battvolts for single cell batteries. This is presented LSB first

Robot to Controller Messages - Status Text

```
typedef struct {
  uint8_t messagetype;
  char    colorcode;
  char    text[22];
} MessageRtoC_text;    // this type (structure) holds a message that goes from
                          Robot to Controller
```

This message type is sent from racer / robot to the controller and contains a text string for display on the controller's TFT screen. There are 3 lines available for status messages. All messages are centered on their display line and are automatically cleared (erased) after 30 seconds.

- messagetype: 2 (MSG_STAT1) for first status line, 3 (MSG_STAT2) for second status line, 4 (MSG_STAT3) for third status line
- colorcode: character indicating color that should be used to display the passed text, per the table above.
- Text: the text to display as an Arduino character array; messages are null (\0) terminated

Robot to Controller Messages - ModeMenu Text

```
typedef struct {
  uint8_t messagetype;
  char    colorcode;
```

```
  char    text[22];
} MessageRtoC_text;    // this type (structure) holds a message that goes from
                          Robot to Controller
```

This message type is sent from racer / robot to the controller and contains a text string for display on the bottom line of the controller's TFT screen. The bottom line is typically used to indicate the "Mode" that the robot / racer is operating in, or about to operate in if is reflecting a menu possibility.

- messagetype: 5 (MSG_MENUITEM)
- colorcode: character indicating color that should be used to display the passed text, per the table above.
- Text: the text to display as an Arduino character array; messages are null (\0) terminated
-

Robot to Controller Messages - Simple Message

```
typedef struct {
  uint8_t messagetype;
  char    colorcode;
} MessageRtoC_simple;    // this type (structure) holds a message that goes from
                          Robot to Controller
```

This message type is sent from racer / robot to the controller for simple commands or messages, such as "clear display" or "beep". Currently neither of these messages are actually implemented for the controller

Joystick Values Display

The second line of the controller's TFT is used to display the joystick values, X and Y. This display is internally driven and does not require any messages from the robot / racer.