

DIY Autohelm ST4000+ Remote Control

PURPOSE:

I basically wanted to create a remote control for my Raymarine ST4000+ for under \$100.00.

This remote is to be open source, use inexpensive hardware, have free software development, and to be used for non-commercial purposes!

To build this remote you will need to have electronic skills, be able to use a soldering iron, be able to diagnose serial communication, Have some Arduino Knowledge, and be able to use a scope or logic analyzer. If you don't have any of these skills, I highly suggest you buy a finished product.

I will offer no help other than what is enclosed. I will try to give you all the information here to get started and pointed in the right direction.

All the software and information is freely available on the web.

DISCLAIMER:

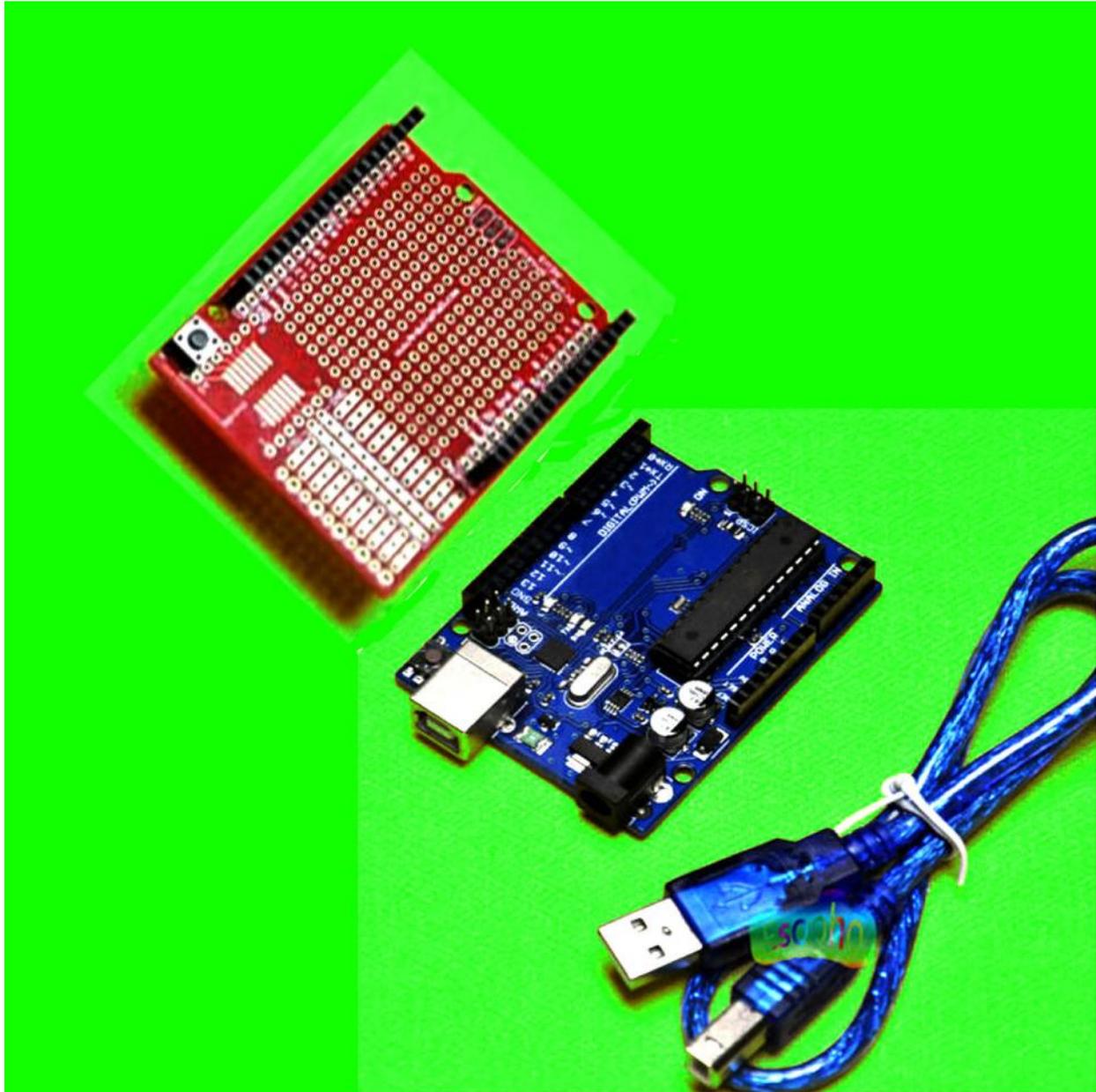
THE SOFTWARE/HARDWARE IS PROVIDED IN AN "AS IS" CONDITION. NO WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE/HARDWARE. I SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES CAUSED BY YOUR CONTRUCTED REMOTE CONTROL, FOR ANY REASON WHATSOEVER.

Hardware Costs:

Arduino Uno + ProtoType Shield (Free shipping)

\$18.00

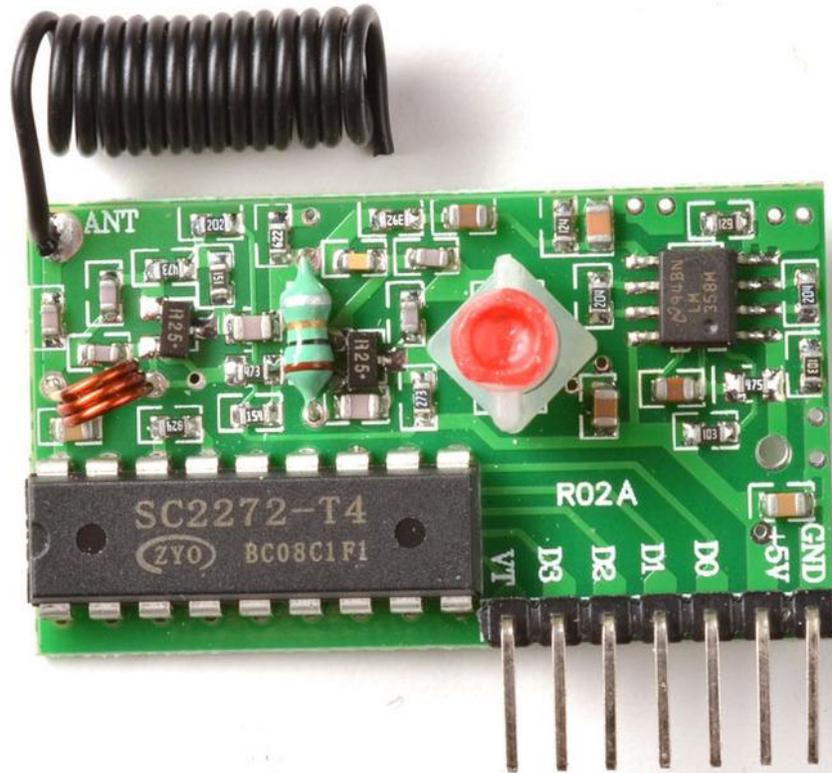
http://www.ebay.com/itm/NEW-UNO-R3-ATMEGA16U2-Board-Proto-Type-Shield-For-Arduino-Mega-1280-2560-328P-/251604457887?pt=LH_DefaultDomain_0&hash=item3a94cb599f



315 MHz Receiver

\$5.00

<http://www.adafruit.com/products/1098>



315 4 Button Transmitter

\$7.00

<http://www.adafruit.com/products/1095>



Enclosure

\$9.00

http://www.ebay.com/itm/Arduino-Project-Box-Official-Arduino-Enclosure-Case-/321421439488?pt=Educational_Toys_US&hash=item4ad635f200



Raymarine Flat Plug (Optional)

\$30.00

<http://www.defender.com/product3.jsp?path=-1|344|2028705|2028871&id=98747>



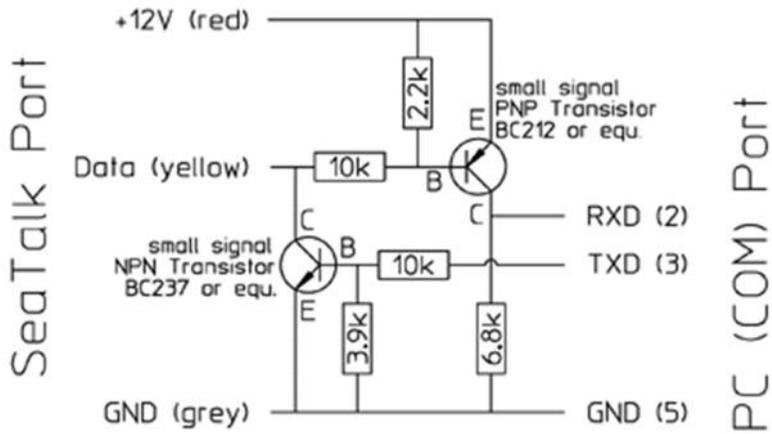
You can find used on Ebay

http://www.ebay.com/itm/8-Raymarine-3-pin-Flat-Seataik-Pigtail-Cable-Raytheon-Autohelm-Autopilot-/191277344801?pt=Boat_Parts_Accessories_Gear&hash=item2c8904a821&vxp=mtr for \$9.00

Or use 3 x 1/8" female spade connectors

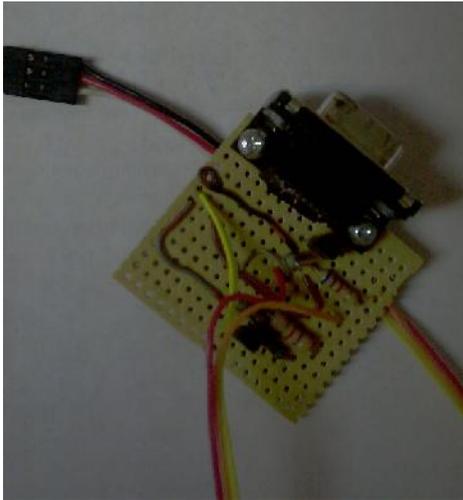
The links shown will probably expire in time and you will need to do some research to find the parts.

The adapter shown will help you with diagnosing problems. The circuit is shown is from the Thomas Knauf page <http://www.thomasknauf.de/rap/seatalk3.htm>.



The NPN transistor can be a 2N4400 and the PNP transistor can be a 2N4402.

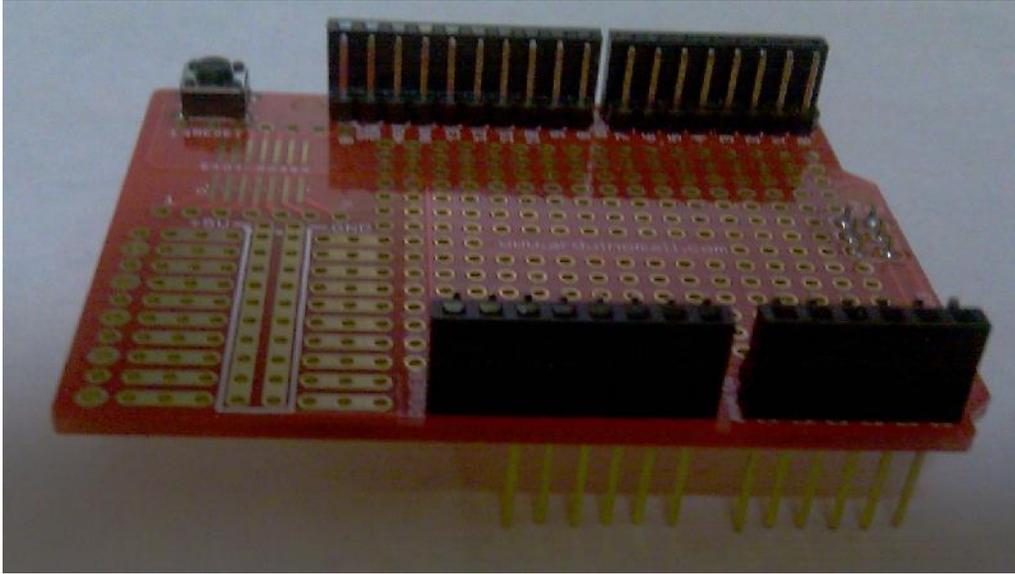
I just bread boarded using parts on hand.



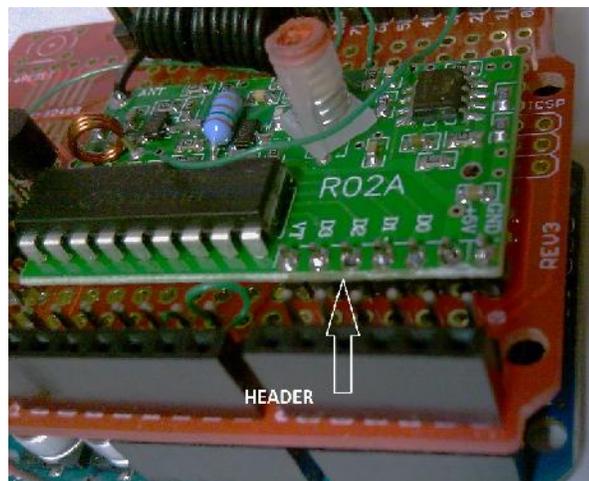
The MSDOS program SEAMON1.exe is available from <http://www.thomasknauf.de/rap/seatalk3.htm>.

Construction:

- 1.) Assemble the prototype shield

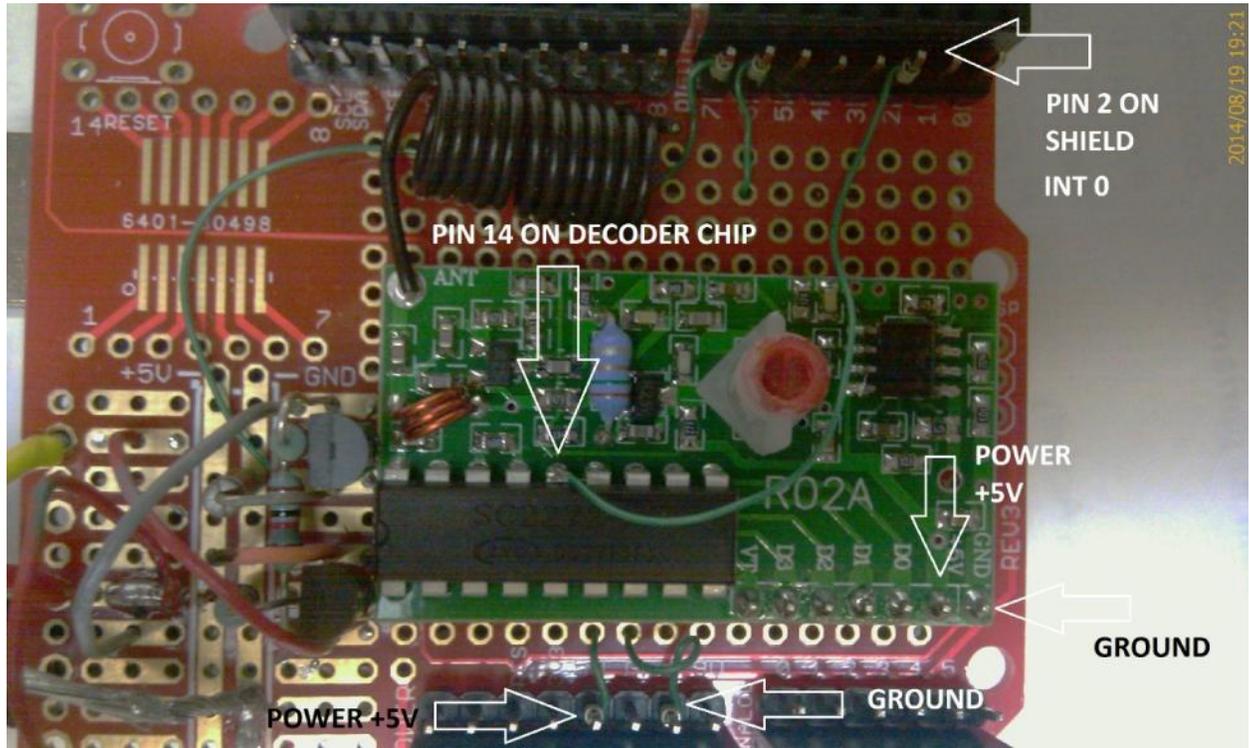


- 2.) Unsolder the existing Header pins from the Receiver (optional). If you don't, the receiver will sit at 90 degrees to the Shield which is ok but you'll need a larger case. Now solder a new straight header as shown to the receiver board.



- 3.) Insert the receiver board into the prototype board and solder in place.
- 4.) I use a wire wrap device to make connections. You can also solder the connections.

Connect the power and ground of the receiver to the power and ground connections on the Shield. Also solder one end of a wire to pin 14 on the receiver decoder chip and the other end goes to pin 2 on the Shield header.



- 5.) Now with this done you can test the receiver.
- 6.) You need to download the Arduino Development Software at <http://arduino.cc/en/Main/Software>.
- 7.) I'm not going to show you how to use this. It is well documented at that site.
- 8.) Next you need to download the RC-SWITCH Library <http://code.google.com/p/rc-switch/>
- 9.) Put the unzipped file into your Arduino Library, Again Read and read more.
- 10.) Load the following code (it is attached as a zip file "Receiver_test.zip")

```

#include <RCSwitch.h>

int ledPin = 13;          // LED connected to digital pin 13
int NO = 0;
int YES = 1;
int HIGHH = 1;
int LOWL = 0;
RCSwitch mySwitch = RCSwitch();
void setup()
{
  Serial.begin(9600);
  mySwitch.enableReceive(0);    // Receiver on interrupt 0 => that is pin #
    int cX;
    for ( cX = 0; cX < 10; cX++ )    // power-on flash LED 10 times
    {
      digitalWrite(ledPin, HIGHH); // turn the LED on (HIGH is the voltage level)
      delay(50);
      digitalWrite(ledPin, LOWL); // turn the LED off (LOW is the voltage level)
      delay(50);
    }
}

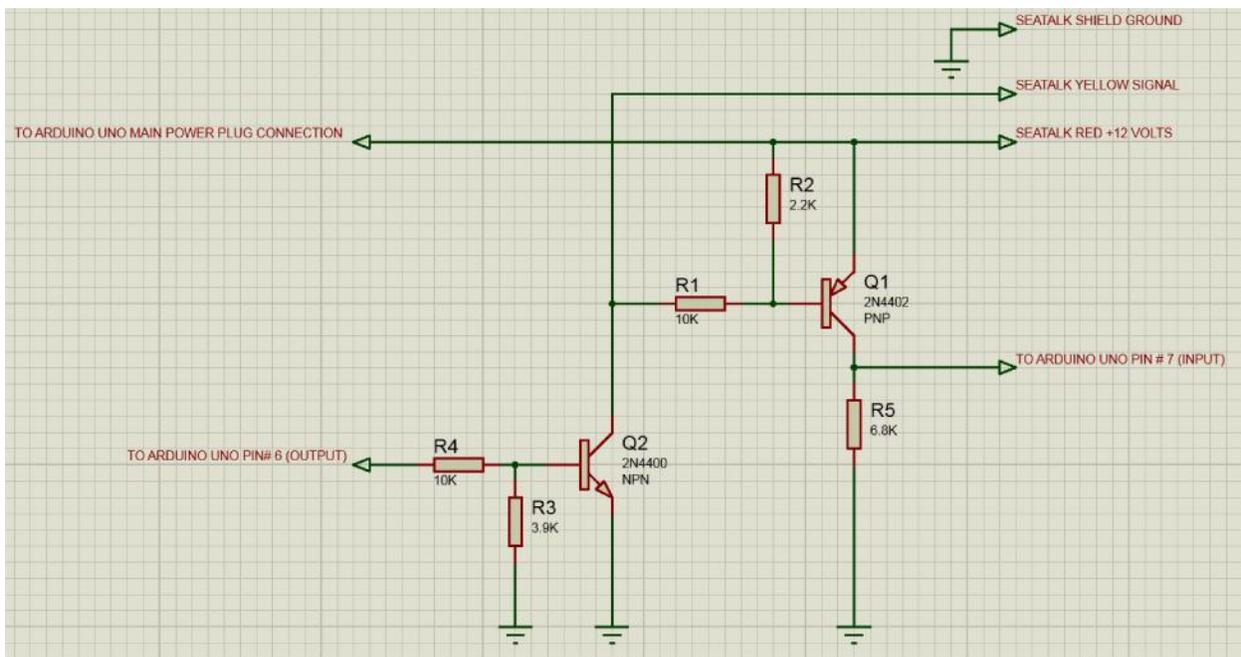
void loop ()
{
  int cX;

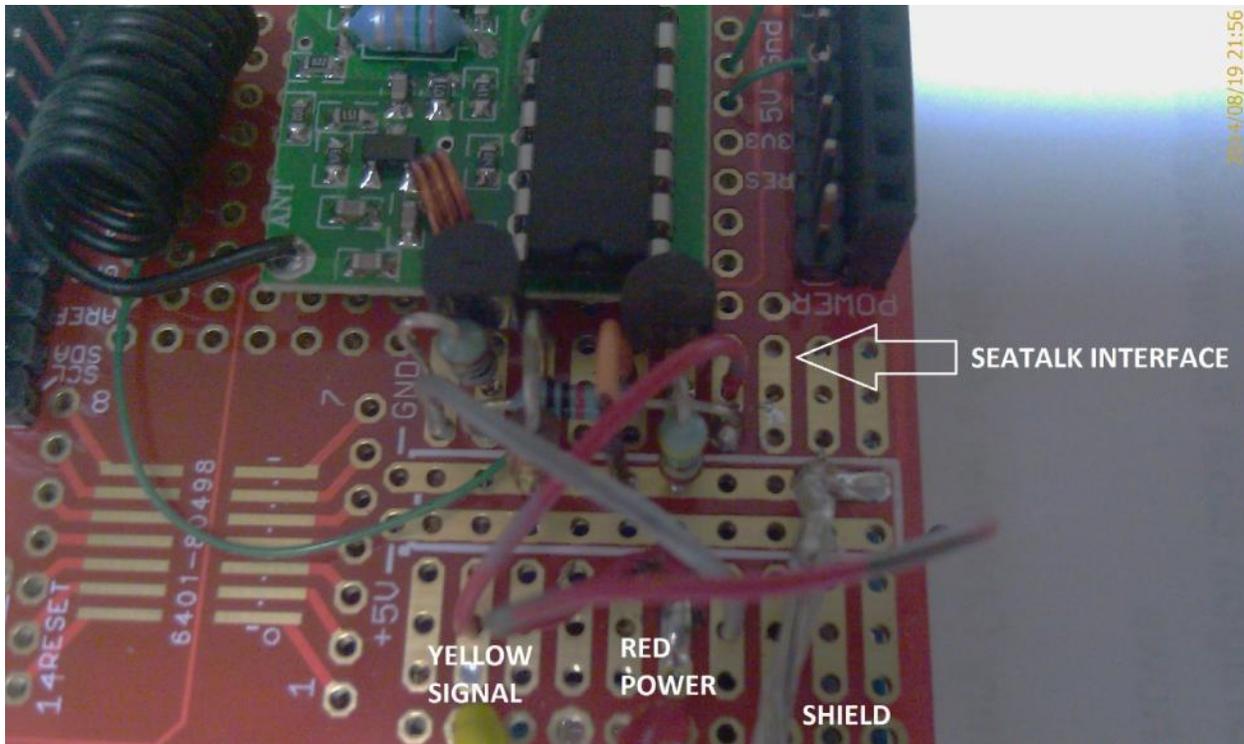
```

```
delay(100);           // programming delay
```

```
if (mySwitch.available())  
{  
  int value = mySwitch.getReceivedValue(); // get key fob value when key pressed  
  Serial.print( value ); // print value on serial monitor  
  mySwitch.resetAvailable();  
}  
  
}
```

- 11.) I left out how to get the program into the Arduino as you will need to read up on that website given earlier, if you're not familiar with using an Arduino.
- 12.) Run the serial monitor in the Arduino Development Software and Press the Remote FOB. Note the values that are displayed as you press the buttons. These should be the same values that I had in my final program. If not, you'll have to change the values in the final program to match your button values.
- 13.) If that works, the next step is to wire up the Seataalk interface on the prototype board. This circuit is available from Yapp Electronics <http://yappelectronics.co.uk/ybw/>





- 14.) Make sure you check all your wiring and the power connections are of the right polarity otherwise the magic smoke may escape.
- 15.) Next download the final program into the Arduino Uno.

/* Seataalk AutoPilot Remote Control

This Project was created for the DIY Sailor Electronics Geek.
 After searching the web, I found that several DIY AP remote projects required quite a few discrete components. I like to use as many off the shelf assemblies as possible.

The code for the RCSSwitch is from <http://dzrmo.wordpress.com/2012/07/08/remote-control-pt2272-for-android/>

The code for the Main Program is from <http://www.vermontficks.org/pic.htm>

The authoritative Website is located here <http://www.thomasknauf.de/seataalk.htm>

All code has been modified/porting to work in this project.

For Non-Commercial Purposes

REMEMBER, USE AT YOUR OWN RISK!

*/

#include <RCSSwitch.h>

```
#define LAMP_OFF    0x00
#define LAMP_ON    0x0C
#define PLUS_ONE   0x07
#define MINUS_ONE  0x05
#define PLUS_TEN   0x08
#define MINUS_TEN  0x06
#define STANDBY    0x02
#define AUTO       0x01
#define TRACK      0x03
#define DISP       0x04
#define TACK_MINUS 0x21
#define TACK_PLUS  0x22
```

```
int ledPin = 13;          // LED connected to digital pin 13
int SEATALK_TX_OUT = 6;
int SEATALK_RX_IN = 7;
int NO = 0;
int YES = 1;
int cLampState;
int HIGHH = 1;
int LOWL = 0;
RCSwitch mySwitch = RCSwitch();
```

```
void setup()
{
  Serial.begin(9600);
  pinMode(SEATALK_TX_OUT, OUTPUT);          // sets the digital pin as output
  pinMode(SEATALK_RX_IN, INPUT);           // sets the digital pin as input
  pinMode(ledPin, OUTPUT);                 // sets the digital pin as output
  mySwitch.enableReceive(0);               // Receiver on interrupt 0 => that is pin #
  int cX;
  for ( cX = 0; cX < 10; cX++ )            // power-on flash LED 10 times
  {
    digitalWrite(ledPin, HIGHH);           // turn the LED on (HIGH is the voltage level)
    delay(50);
    digitalWrite(ledPin, LOWL);           // turn the LED off (LOW is the voltage level)
    delay(50);
  }
}
```

```
void loop ()
{
  int cX;
  delay(100);                               // programming delay
}
```

```

    digitalWrite(SEATALK_TX_OUT, LOWL);          // allow output to float

if (mySwitch.available())
{
    int value = mySwitch.getReceivedValue();    // get key fob value when key pressed
    switch (value)
    {
    case 21964:                                  // FOB Buttons B&D Pressed together to give "Standby"
        digitalWrite(ledPin, HIGHH);
        Serial.print( value );                  // Print out Value of FOB, used in testing
        SendKeystrokeMsg ( STANDBY );
        digitalWrite(ledPin, LOWL);
        break;
    case 21811:                                  // FOB Buttons A&C Pressed together to give "Auto"
        digitalWrite(ledPin, HIGHH);
        Serial.print( value );
        SendKeystrokeMsg ( AUTO );
        digitalWrite(ledPin, LOWL);
        break;
    case 21763:                                  // FOB Button A Pressed to give "-10 DEG"
        digitalWrite(ledPin, HIGHH);
        Serial.print( value );
        SendKeystrokeMsg ( MINUS_TEN );
        digitalWrite(ledPin, LOWL);
        break;
    case 21772:                                  // FOB Button B Pressed to give "+10 DEG"
        digitalWrite(ledPin, HIGHH);
        Serial.print( value );
        SendKeystrokeMsg ( PLUS_TEN );
        digitalWrite(ledPin, LOWL);
        break;
    case 21952:                                  // FOB Button D Pressed to give "Tack Right"
        digitalWrite(ledPin, HIGHH);
        Serial.print( value );
        SendKeystrokeMsg ( TACK_PLUS );
        digitalWrite(ledPin, LOWL);
        break;
    case 21808:                                  // FOB Button C Pressed to give "Tack Left"
        digitalWrite(ledPin, HIGHH);
        Serial.print( value );
        SendKeystrokeMsg ( TACK_MINUS );
        digitalWrite(ledPin, LOWL);
        break;
    }
}

```

```

    }

    mySwitch.resetAvailable();
}

}

void SendKeystrokeMsg ( int cData )
{
    int cError;

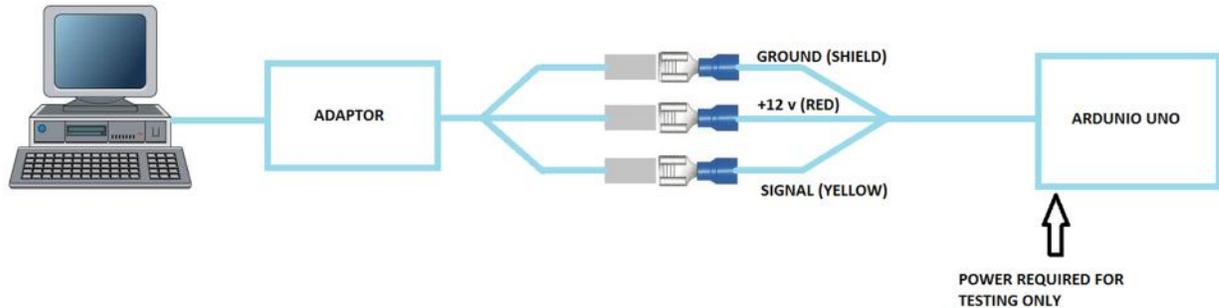
    do {

        CheckBus();                // wait for bus to be idle
        digitalWrite(ledPin, HIGH); // turn the LED on (HIGH is the voltage level)
        cError = SendByte ( NO, YES, 0x86 ); // command: keystroke
        cError = SendByte ( cError, NO, 0x11 ); // data: remote control, 1 extra byte (4 total)
        cError = SendByte ( cError, NO, cData ); // data:
        cError = SendByte ( cError, NO, ~cData ); // data: inverted data
        delay(10); // LED visible delay
        digitalWrite(ledPin, LOW); // turn the LED off (LOW is the voltage level)
    } while ( cError == YES ); // repeat if message was corrupted
}

int SendByte ( int cError, int cCommand, int cData )
{
    int cX;
    if ( cError != YES ) // if no error from previous
    {
        cError = SendBit ( cError, HIGH ); // start bit (0V)
        for ( cX = 0; cX < 8; cX++ )
        {
            cError = SendBit ( cError, ~cData & 0x01 ); // LSB data bit
            cData >>= 1; // shift right
        }
        cError = SendBit ( cError, cCommand ? LOW : HIGH ); // set if command byte, clear if data
    }
    byte
    cError = SendBit ( cError, LOW ); // stop bit (+12V)
}
return ( cError );
}

```


- 16.) Hook up the adaptor as described at the beginning of this project and run SEAMON1.exe.
17.) Hook up your Arduino to the adaptor.



- 18.) Next using the remote, press the various buttons and observe the sentences sent. If they correspond with the ones high lighted below, you're ready to try it on the real thing.
(These are from Thomas Knauf Webpage <http://www.thomasknauf.de/rap/seatalk2.htm>)

86 21 01 FE Auto

86 21 02 FD Standby

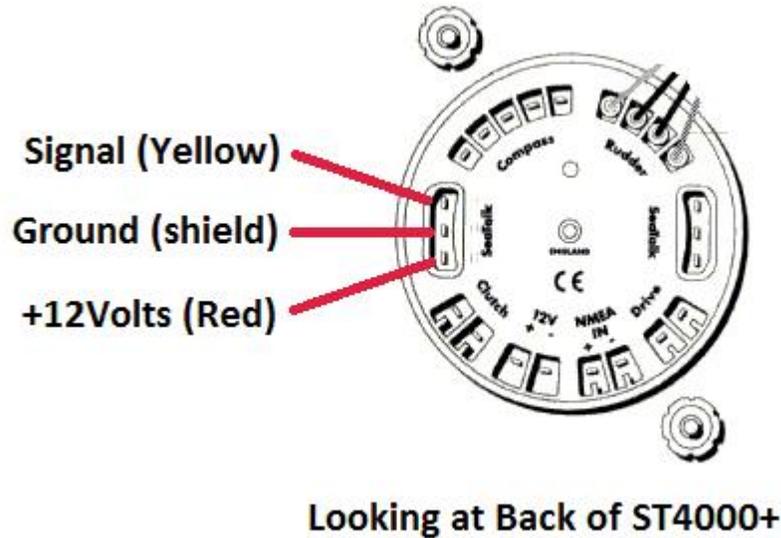
86 21 06 F9 -10 (in auto mode)

86 21 08 F7 +10 (in auto mode)

86 21 21 DE -1 & -10 (Port tack)

86 21 22 DD +1 & +10 (Starboard tack)

- 19.) Next plug into your autopilot (Raymarine st4000+) at you own risk!
- 20.) Press the remote buttons and see the results. I get about 25 feet reliably without an external antenna and the unit mounted in the Pedestal.



Remember this is still a work in progress and there are additional software modifications that can be done. This brief will get you going if you want to try it. You can probably port this to other microcontrollers but I chose the Arduino for sake of convenience.

And REMEMBER NOT TO LET THE MAGIC SMOKE ESCAPE or you may be looking for new hardware.

Always check and double check connections for correct polarity.

GOOD LUCK!

Revision Aug 29, 2014

Added a different Enclosure and finished picture.

Enclosure is a Hammond 1591CSFLBK @ \$5.95

