**Build an Arduino Repeater Control System for the RS-UV3 Repeater**

By

Marty Boroff, WD9GYM

I obtained my Novice Class license in October 1977. A month later I earned my Technician class license. I was very interested in making use of the local repeater’s autopatch. Cell phones were not available for the average person. I operated on 2 meters for the next two years when I earned my General ticket. Not much later I obtained an Advanced class license and operated HF and VHF until the personal computer invaded my home in the late 1980s.

For close to forty years I was a programmer/system consultant for a major corporation in the northern suburbs of Chicago. When I retired in 2006 I stopped programing any computers until December of 2013 when I stumbled into the Arduino world. Since then I have constructed numerous DIY projects to use around the house. In early 2015 the book “Arduino Projects for Amateur Radio” by Dr. Jack Purdum, W8TEE and Dennis Kidder, W6DQ got me back in the game by sparking my interest by combining both hobbies. I purchased and programmed an open source Ten Tec Patriot.

The September, 2015, issue of QST had an ad for the RS-UV3 which indicated that two of these boards could be used as a repeater. Several email exchanges with Jim Veatch at Hobbypcb.com indicated the following information about this project:

**RS-UV3 Functions**:

1. Configurable hang-time that keeps the PTT up for a given length of time after the PTT  
    signal goes away.

2. Text-to-Morse code with programmable speed and tone frequency.

3. ID timer to send a stored ID at a specific interval, waits for the PTT to go away then

ID's and drops.

4. Time out timer to limit the length of time the TX can be keyed, can send a CW message when time-out occurs.

5. CTCSS (tone squelch) encode and decode.

6. DTMF encode and decode (right now it needs an external processor to initiate or process the tones).

The repeater has the COR out of the receiver wired to the PTT of the transmitter and RX audio sent to TX audio. The system has a local speaker that can be turned on or off and a place to plug  in the USB-TTL serial adapter to either the TX or RX for configuration. The TX feeds the 2W PA and the RX is directly connected to the duplexer.

One can easily change the mounting configuration and by using two soft-serial ports (or a version of the Arduino that has more 'real' serial ports), the Arduino would communicate with both the transmitter and receiver. Since the TX produces +23 dBm (200 mW) and the RX can hear -125 dBm signals you need on the order of 150 dB isolation between them for full duplex operation. So mounting them in separate shielded enclosures is a requirement.

**Functions the Arduino could perform:**

Basic Control functions:

1. Repeater ON/OFF
2. Tone Squelch ON/OFF
3. Fan ON/OFF (the RS-UV3's have temp sensors)

Remote Base:

Add a third RS-UV3 and use on 2M/220 to link to the 440 repeater. Talk on simplex or other repeater or tune in the NOAA broadcast but check to see if it's legal to rebroadcast NOAA it might be.

Voter:

RS-UV3's can measure the signal strength and S/N ratio of the incoming signal. The Arduino could use this info to select the best signal from remote receivers.

Logging repeater usage:

Power Management:

1. Solar, Battery
2. Mains and back-up power.

Paging:

   Use DTMF signaling to alert users

**Project Scope:**

I learned many years ago about the KISS (Keep It Simple Stupid) principle. I prefer to develop my code in functional blocks. Although this can lead to spaghetti code, I make every attempt to keep code modularized as functional “Black Box” routines. Construction begins with a bread board model.

In keeping with the KISS principle, I have decided to break this project up into multiple articles. This project develops a user interface to the RS-UV3 via the Arduino which looks and feels very similar to what is found on modern handheld (HT) radios. In addition to providing operational control, I built in the first three functions listed above. The next project may contain logging some form of power management and additional functions.



*Figure 1. The Completed Repeater Control System.*

# The Components

There major components in this project are presented in Table 1. The Arduino ATMEGA2560 was chosen because it has considerable memory, and multiple serial ports to communicate with multiple RS-UV3 units. The relay board is used to control power for two UV3 boards and one of more fans. The LCD display provides an inexpensive output with enough real estate to provide menu processing. The Matrix Keypad is about a simple an input device as I could find. I have been using micro UBECs for a while and find they very nicely allow me to use my shack’s 13.8 V 15A power supply for Arduino projects and my radios. For the most part, I have found that Schmartboard jumper wires secure to header pins much better than Dupont jumper wires but this is a personal preference as they are a bit more expensive.

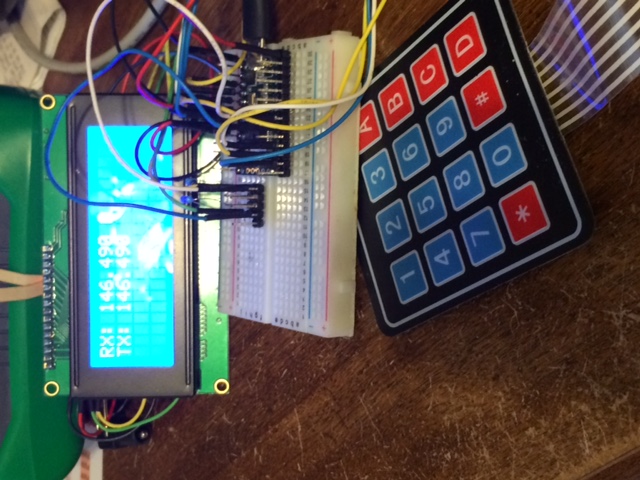
*Table 1. Major Construction Components.*

|  |  |  |
| --- | --- | --- |
| **Component** | **Cost** | **Description/Vendor** |
| ATMEGA2560 | $9.95 | Inland Arduino Mega 2560 – Micro Center <http://www.microcenter.com/product/431995/Arduino_Mega_2560> |
| GHEO ATMEGA2560 Prototype board | $3.99 | **Micro Center** <http://www.microcenter.com/product/447055/Mega_Proto_PCB_Shield_Rev_3_for_Arduino> |
| 2 Channel Opto-Isolated Relay Board | $3.00 | Your Duino.com  <http://yourduino.com/sunshop2/index.php?l=product_detail&p=218> |
| LCD Display Blue 2004 (20x4) IIC, I2C, TWI | $10.00 | Your Duino.com  <http://yourduino.com/sunshop2/index.php?l=product_detail&p=332> |
| 4X4 Matrix Keypad | $2.00 | Your Duino.com  <http://yourduino.com/sunshop2/index.php?l=product_detail&p=393> |
| Micro UBEC 3A/5 V | $3.69 | Hobby King <https://www.hobbyking.com/hobbyking/store/uh_viewItem.asp?idProduct=62873> |
| 12 Vdc fan | $0.00 | Salvage from old equipment or eBay |
| Schmartboard Jumper Wires | $25.00 | Assorted wire bundle of 7” jumpers  <http://schmartboard.com/schmartboard-qty-10-of-each-yellow-blue-red-black-white-7-female-jumpers-and-200-headers-920-0112-50/> |

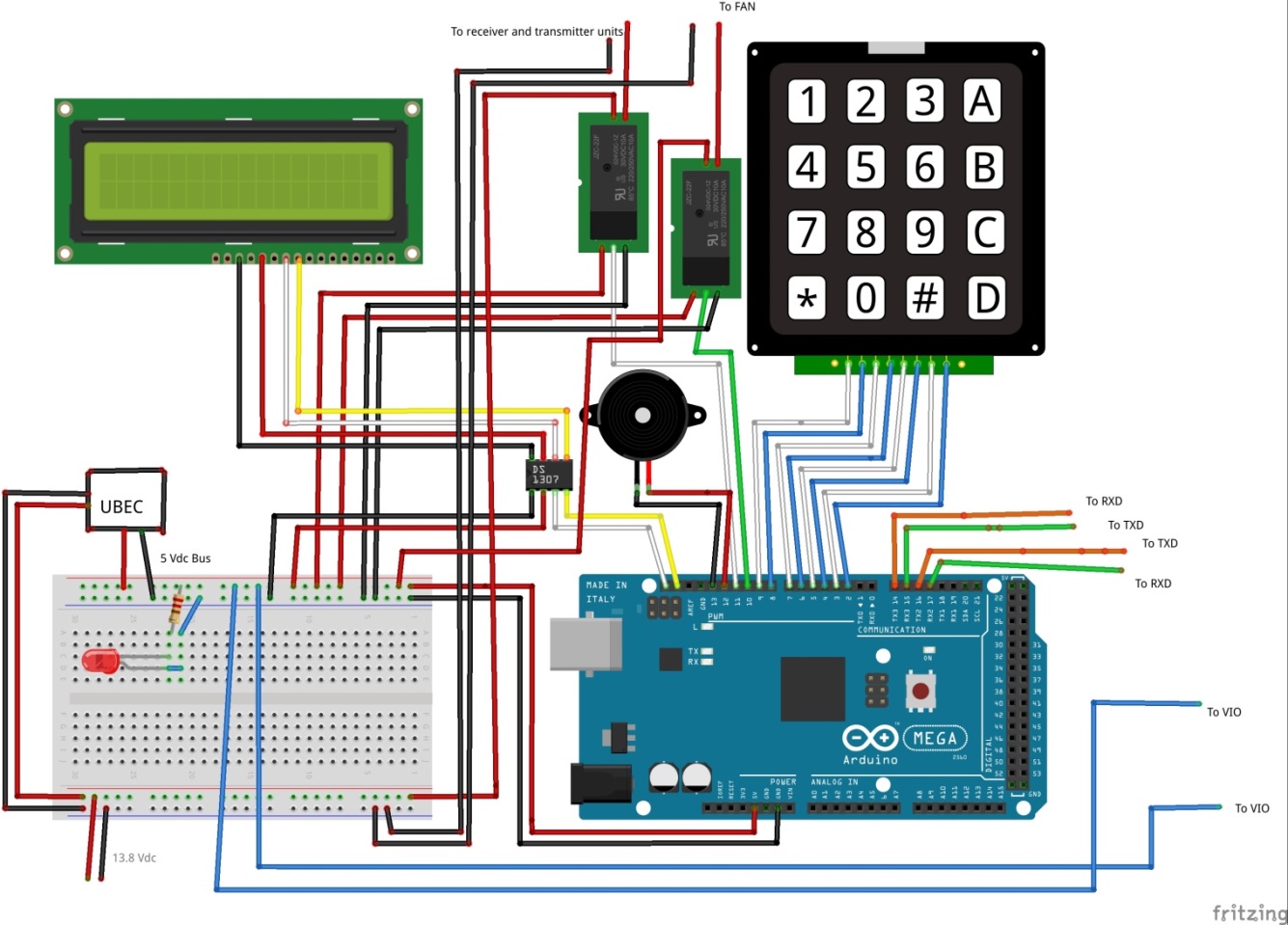
|  |  |  |
| --- | --- | --- |
| 1K Resister | $0.00 | In junk ox |
| Power LED Red or Green | $0.00 | In Junk Box |
| 3.2 Battery Clip | $0.82 | eBay |
| CR2032 | $0.89 | eBay |
| DC Power Socket Panel Mount | $0.24 | Found on eBay 5mm x 2.1 MM |
| Header Pins | $0.00 | Various straight and 90 digress bend header pins. |
| Total Cost | $39.24 | Does not include the Schmartboard jumpers |

**Construction**

I always begin a project with a half or full size breadboard. Depicted here is a breadboard ready to start the user interface programing. Additional components, battery backup for the clock and relays, were added during development.



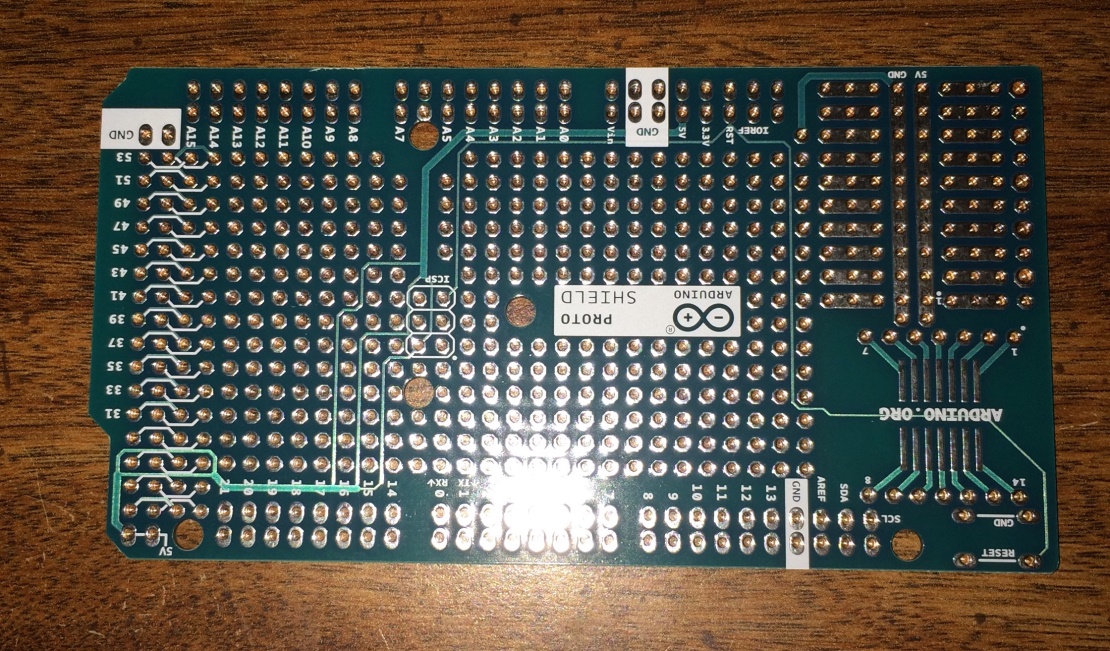
*Figure 2. Photo of Breadboard wiring of the Control System.*



*Figure 3. Breadboard wiring of the Repeater Control System.*

Using the a prototype board allows easy transfer of the breadboard wiring to a permanent

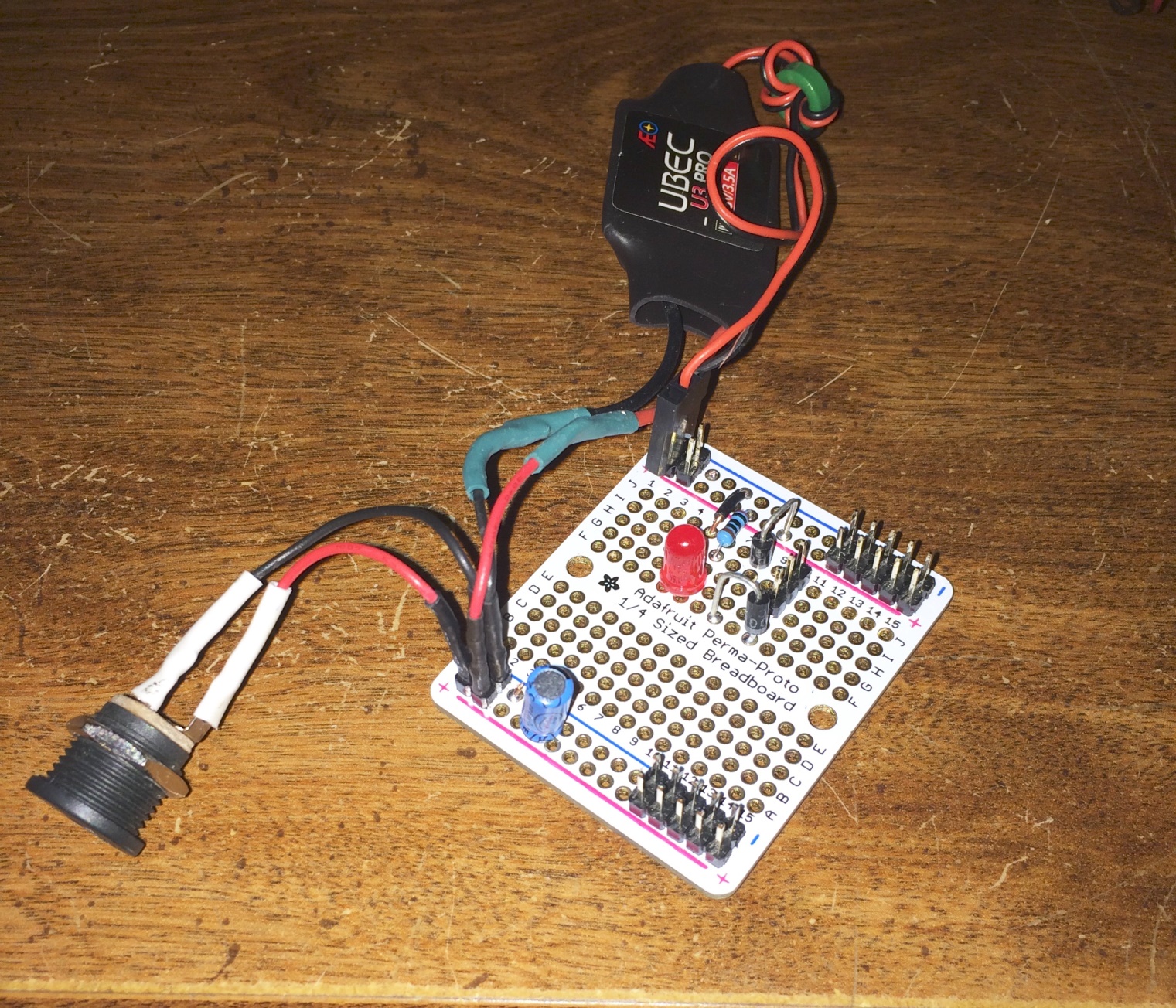
PCB. By soldering 90 degree header pins to these boards you can use Schmartboard female jumpers for the same wiring points. I like to use a prototype board that has a 5V bus to attach peripheral power connections. This board has room for 11 pair of connections.



*Figure 4 GHEO Prototype Boards.*

I tried several different do-it-yourself power supplies, such as a LM7805, and a couple of 10 uF capacitors for line filters, but I found the LM7805 usually runs very hot and requires a heat sink. Using a UBEC rated for 5Vdc @ 3Amps run nice and cool and can power a microcontroller and lots of 5V peripherals.

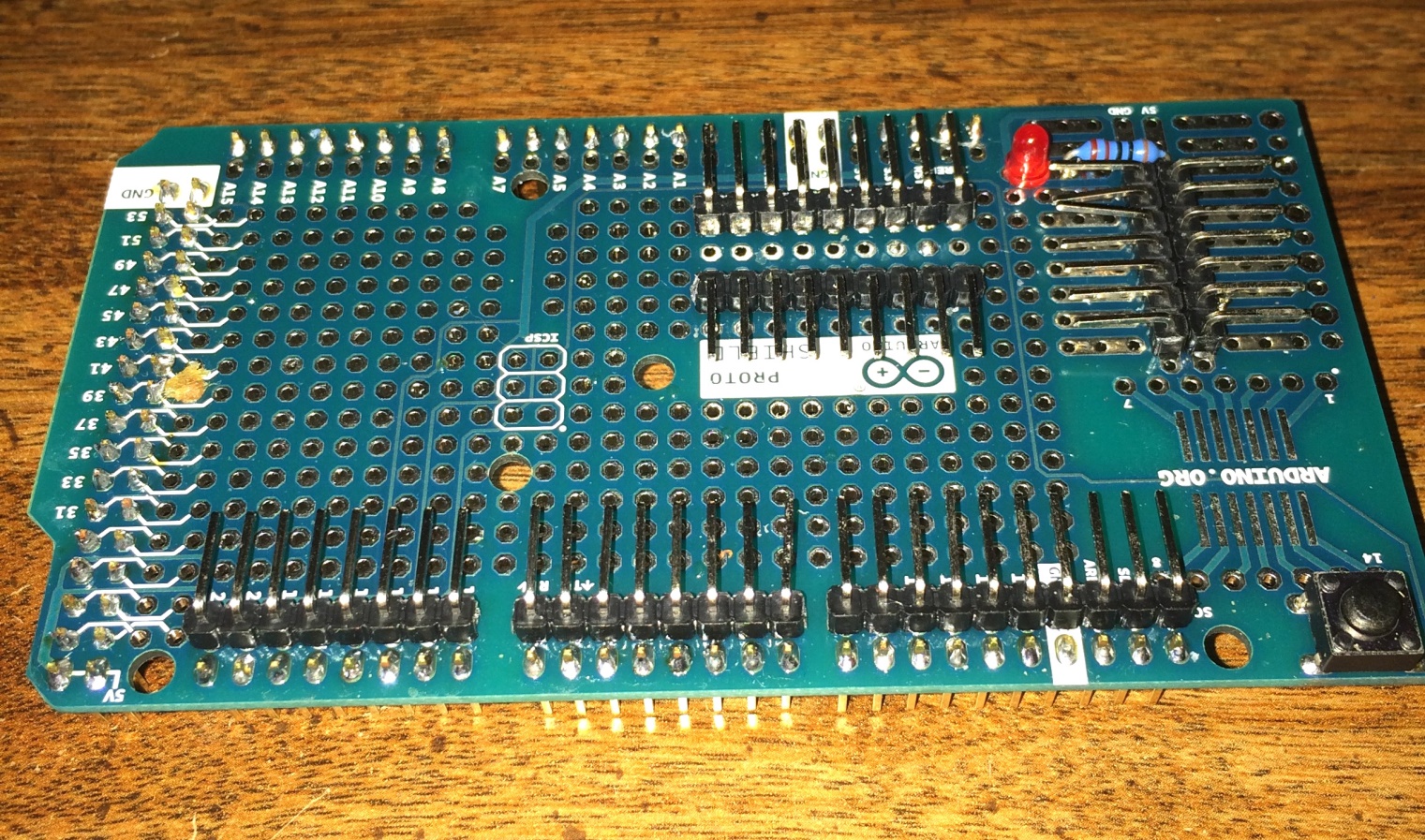
This is an example useing Schmartboard jumpers for all of the connections. If you solder the wire to the Perm Proto board, ensure you use some sort of quick-connect so that you can fish the wires for the power connector and a rocker switch through holes in your project case.



*Figure 5 Typical Power Supply Board.*

The bus on the right is the 5Vdc bus, the bus on the left runs 9Vdc – 15Vdc as required to power the RS-UV3 boards and a 12Vdc cooling fan, and in the center are 3 Volt connections. The Also, connect the negative side of both buses together. There are an infinite number of applications for this power supply. This project puts all of the jumpers on the ATMEGA2560 prototype board.

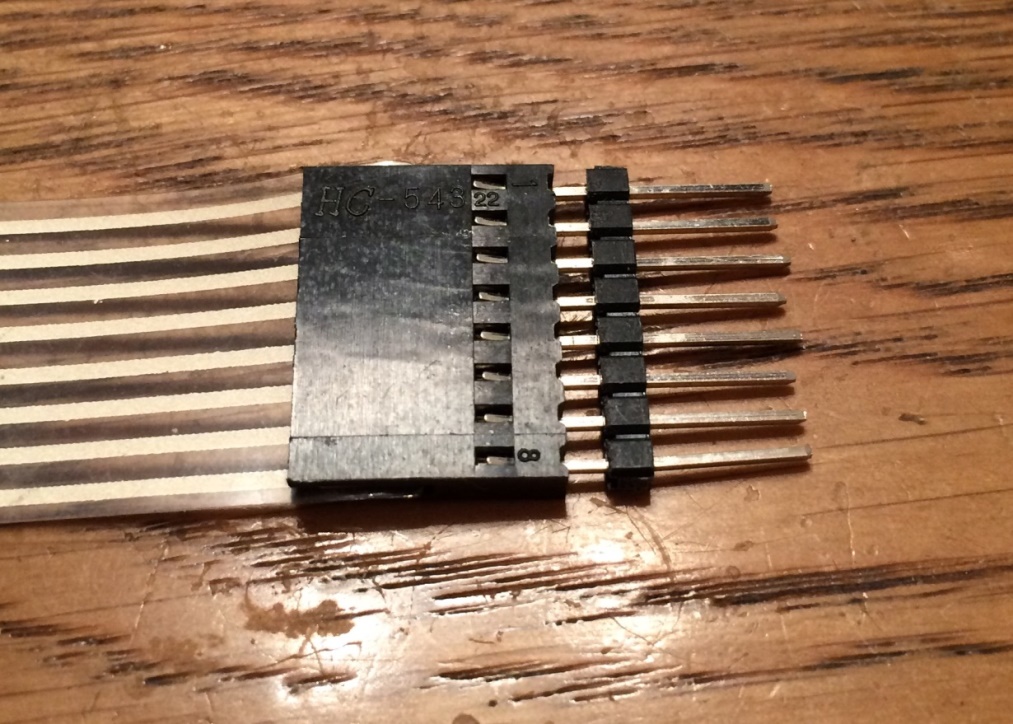
Main board



*Figure 6. Completed ATMEGA2560 Prototype Board.*

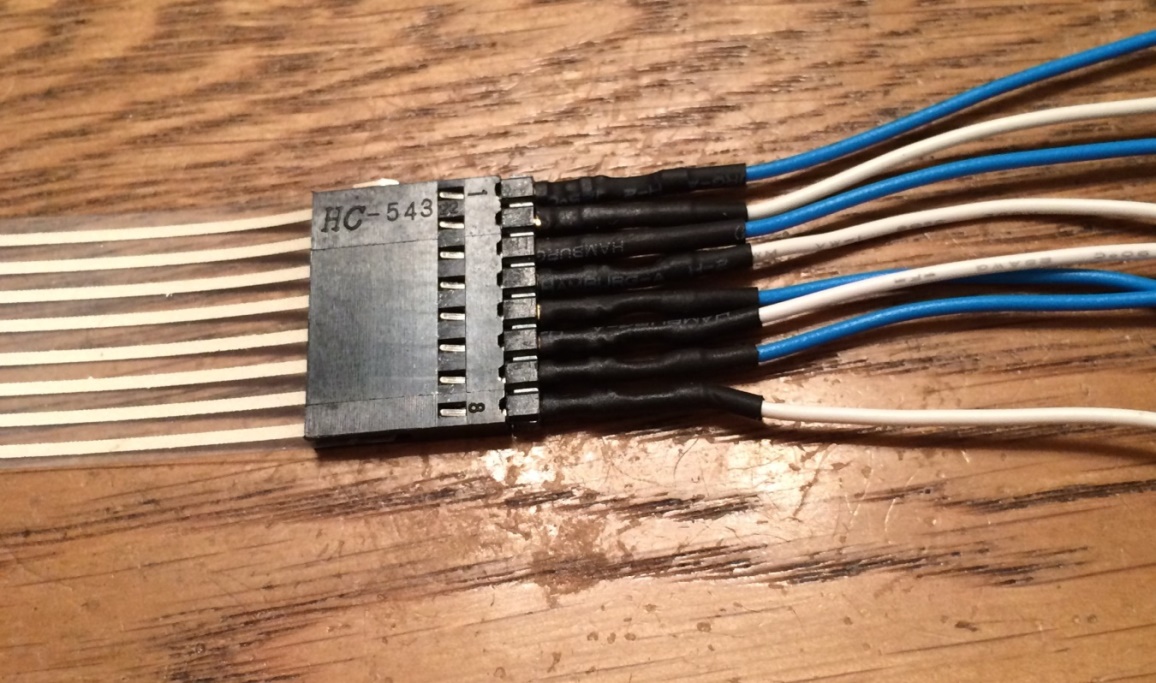
The main board is constructed by using a prototype board for the ATMEGA2560. I put straight header pins in all of the holes along the three edges that align with the ATMEGA2560 when seated on the ATMEGA2560. On the 5 volt bus I put 90 degree header pins. I like to add a 220 ohm resister and a red LED that will light up when 5 volts are on the bus. And finally this project will require a bus for 9 Volts to 12 Volts to supply power for the UV3s and the UBEC power module.

Peripheral Wiring



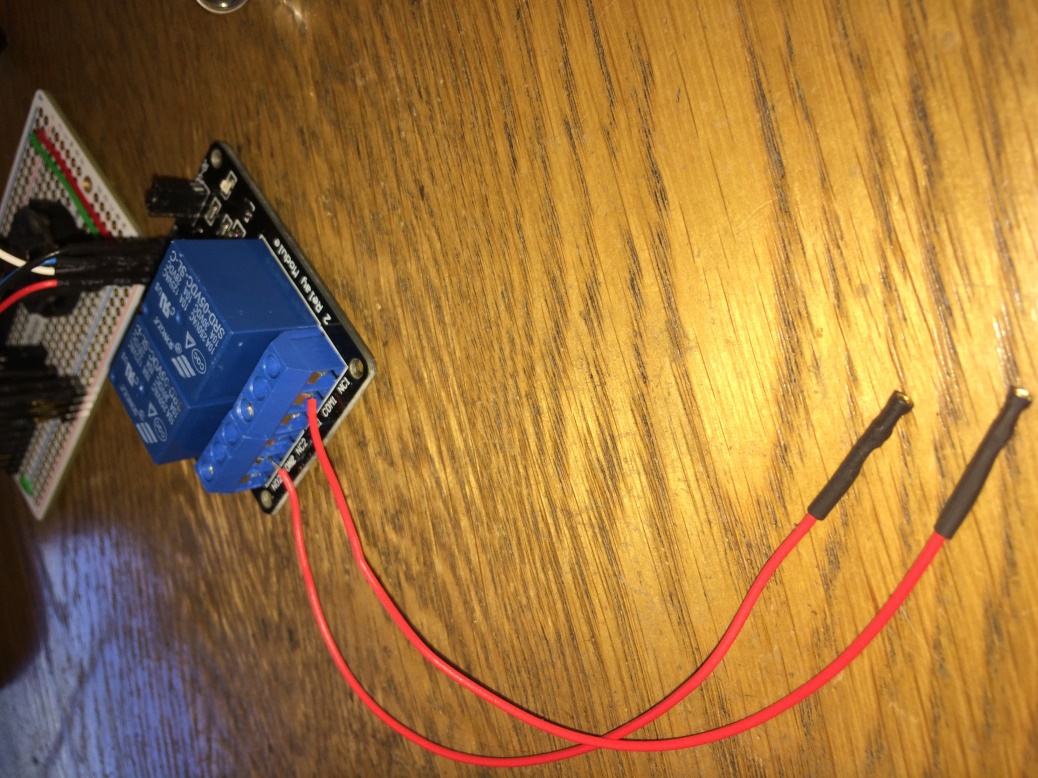
*Figure 7. Inserting double male jumpers into keypad ribbon connector.*

Insert a row of 8 double header pins into the end of the ribbon cable on the keypad.



*Figure 8. Keypad connected to jumper wires.*

Insert alternating colored jumpers into the row of 8 double header pins.



*Figure 9. Relay board with jumpers to power supply.*

Connect two red jumpers to comm of each relay. Connect the other end of relay one to the 13.8 V bus of the power supply. Connect the other to either the 13V bus of the power supply or 5V bus depending on the voltage of your cooling fan. Connect the negative of the UV3s and the fan to the 13V bus negative.

Connect all of the peripheral Vcc and Gnd to the 5 Volt bus. Connect all of the data or signal leads to the appropriate Arduino pins.

|  |  |
| --- | --- |
| Pins 2 – 9 Keypad | Pins 14/15 are Tx and RX leads to UV3 unit A |
| Pin 10 In1 on relay board | Pins 16/17 are Tx and RX leads to UV3 unit B |
| Pin 11 to In2 on Relay board | Pin 20 is SDA on the DS3231 and the 20 X 4 LCD |
| Pin12 to + buzzer, - buzzer to 5V negative on bus | Pin 21 is SCL on the DS3231 and the 20 X 4 LCD |

**Software**

The source code and all documentation maybe downloaded from:  
  
 <https://github.com/mboroff/Repeater_Controller.git>   
  
The files should be installed (unzipped) into Documents/Arduino/Repeater\_Controller. In order to compile the code you must download and install the Arduino IDE version 1.6.5 or newer from:

<https://www.arduino.cc/en/Main/Software>  
  
Download and install DS3231 library from:  
  
 <http://www.rinkydinkelectronics.com/library.php?id=73>  
  
 There are a bunch of real time clocks available. I built a couple clock projects that used DS1307 chips and an external crystal. Over time these boards lost accuracy. DS3231 boards cost about the same but hold their accuracy for my projects.

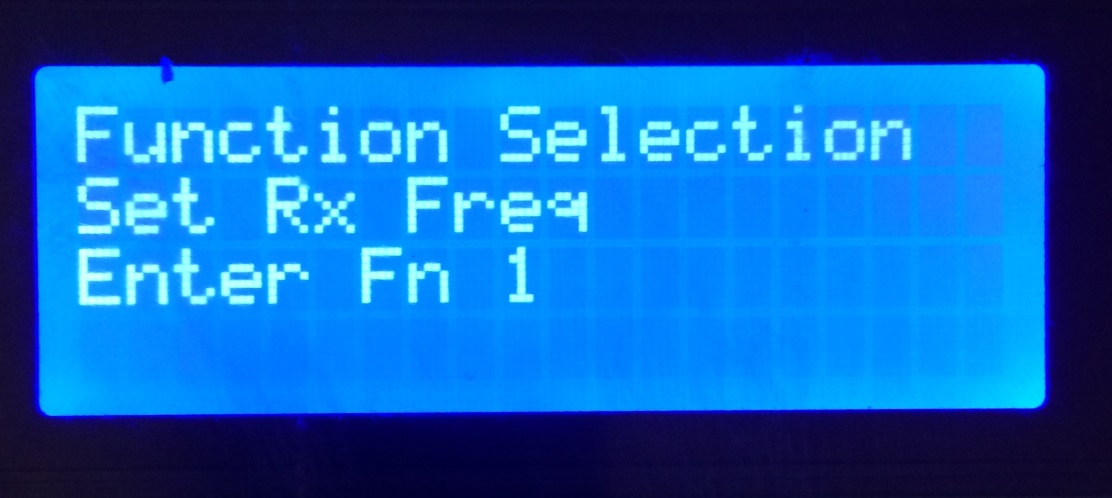
For information on using the Arduino IDE see <https://www.arduino.cc/en/Guide/HomePage>. If you have no experience installing third party libraries, Adafruit has one of the best tutorials found at <https://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use>.

Repeater\_Controller.ino is the main file to be used by the IDE. This file contains all of the global variable, setup() and loop() functions. I have used the Arduino IDE option “***Add a file***” to define places for each major function of the sketch. Some examples are beep\_function.ino, timeRoutines.ino….etc.

|  |  |
| --- | --- |
|  | Current board  Temperature of UV3 |
| *Figure 29. Main Display.* |  |

Operation

The **\*** (asterisk) key switches the display to the first function.



*Figure 30.Function menu.*

Press **‘\*’** to enter the current function. Press **‘A’** or **‘B’** to scroll through functions. Press **‘#’** to cancel and return to Main Display.

Within each function the “**\***” is pressed to confirm and store the new value. The “**#**” can be pressed to cancel the function.

The available functions for this release are:

1. Set Rx Freq – Set the receiving frequency of the selected RS-UV3
2. Set Tx Freq – Set the transmitting frequency of the selected RS-UV3
3. Set Sqlch Level – Set the squelch level of the selected RS-UV3
4. Tone Sqlch Mode – Set the Tone Squelch Mode of the selected RS-UV3
5. Set CTCSS – Sets the CTCSS tone frequency
6. Repeater ON/OFF – Turns the power on or off of the connected RS-UV3 units \*
7. BEEP ON/OFF – Turns the audio acknowledgement of a key press On or Off \*
8. Fan ON/OFF – Turns the cooling fan On or Off \*
9. Set Clock – Sets the time of the clock \*\*
10. Set Volume – Sets the volume of the selected RS-UV3
11. Display System Info
12. Reset RS-UV3 \*\*\*
13. Set Call Sign – Keypad performs like a telephone keypad i.e. ‘2’ = A B C 2
14. Transmit Call Sign

\* The Arduino stores the values of the repeater power state, fan state and beeper state are  
 maintained in non-volatile memory.

\*\* Provided a 3 V battery is connected to the Arduino time is maintained. DST and Time Zone is   
accommodated. Time is set by selecting a time zone and entering UTC.

\*\*\* Provided you wired for a Piezo Buzzer, the buzzer will be unconditionally sounded three times

**Appendix A:**

Jack Purdum, Dennis Kidder, *Arduino Projects for Amateur Radio*, McGraw-Hill Publishing