Ag Irrigation-Pump Remote Control

Using Arduino-UNO and Digi-Xbee RF

Written By: Thomas G (01/2017)

Last Updated: 2017.05.04

Illustration 1: HandRemote

Illustration 2: PumpController

Table of Contents

1. INTRODUCTION

The "***Ag Irrigation Pump Remote Control***" project's goal is to provide RF remote control and monitoring features of an Agriculture-Industry Irrigation Pump; typically used in wheel-line and hand-line irrigation setups. The project uses Xbee RF radio's with up to a 28-mile range (More practically a mile or two at ground level). The project consists of a Hand-Held "***Hand Remote***" and a pole mounted "***Pump Controller***" as well as high voltage control electrical components to be added inside the pumps electrical panel.

- ✔ The project is Open Source and Resides at: https://github.com/tgit23/AgIrrigationRemoteControl
- ✔ The most recent version of this document is at: https://github.com/tgit23/AgIrrigationRemoteControl/docs/AgIrrigationPumpRemoteControl.pdf

1.1 Feature and Limitations

- ✔ Signal Distance (Recommended usage for up to 2-miles; spotty but usable usage beyond 2-miles)
  - ○ Test Scenario
    - ▪ Pump-Controller mounted 5-feet from the ground on a pole
    - ▪ Hand-Controller checked inside a moving vehicle (~ 3-feet off the ground)
  - ○ Results
    - ▪ At 1-3/4 miles away from the Pump-Controller the Hand-Remote had signal 90% of the time of motion at in a rural populated area
    - ▪ Houses were located on square mile blocks and had approximately 6 to 10 houses / buildings per mile
    - ▪ Ground level is fairly flat and actually dips down very slightly in many areas (Slightly lower spots didn't seem to affect range)
  - ○ The Digi XBEE S3B RF radio module is advertised to have a range of up to 26-miles (that range has not tested to be accurate for the common user)
- ✔ Up to (4) Pump-Controllers can be controller on each Hand-Remote
  - ○ Limited by the current firmware and the Arduino UNO EEPROM memory
  - ○ Limit can be extended with the right firmware modification (not yet designed)
- ✔ Any number of Hand-Remotes can be used

1.2 Bill Of Materials (BOM) - $241.26

Pricing for multiple units may be a little less than itemized pricing due to usage of excess material.

Itemized Pricing

- ✔ Pump-Control = **$186.26** / Per-Pump-Controller
  - ○ (1) 480V 3-Phase to 12VDC Power Supply = **$ 66.99** - $59.00 (RHINO) + $7.99 (FULREE)
  - ○ (1) 600VAC Solid State Relay = **$24.50**
  - ○ (1) DIN Rail = $ 8.00
  - ○ (1) Arduino UNO R3 w/Male Headers = $ 3.80
  - ○ (1) Zigbee Shield = $ 4.83
  - ○ (1) Xbee 900hp Pro Module = **$36.11**
  - ○ (1) High Gain Antenna = $ 4.33
  - ○ (15) Screw Terminal Blocks = $ 2.15
  - ○ (4) Female DuPont Pin Headers = $ 0.79
  - ○ (1) 50x70cm Perfboard = $ 1.56
  - ○ (1) Right-Angle 9VDC Power Plug = $ 4.51
  - ○ (1) Computer Power Connector = $ 1.99
  - ○ (25-feet) 600V Wire 16AWG+ = $ 9.18
  - ○ (50-feet) Outdoor data cable = $11.50
  - ○ (Various) Solder-less Crimp Connectors = $ 6.02

- ✔ Hand-Remote = **$55.00** / Per-Hand-Remote
  - ○ (1) Arduino UNO R3 = $ 3.80
  - ○ (2) Keypad / LCD Shield = $ 2.19
  - ○ (1) Zigbee Shield = $ 4.83
  - ○ (1) ON/OFF Rocker Switch = $ 0.86
  - ○ (1) Active Buzzer = $ 0.60
  - ○ (2) Resistors = $ 2.28
  - ○ (1) Xbee 900hp Pro Module = **$36.11**
  - ○ (1) High Gain Antenna = $ 4.33

A. Pump-Panel Components ($99.50)

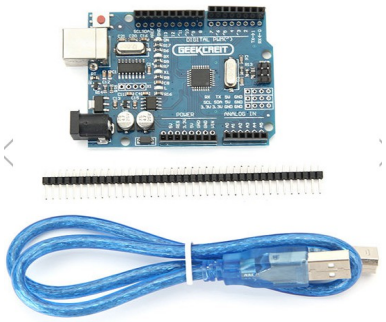
(1) RHINO switching Power supply, 24 VDC output, 2.5A, 60W, 320-600 VAC input, 3-phase, 35mm DIN
AutomationDirect.com ($59.00/ea)
--- OR ---
(1) Delta 3-phase to 24 VDC Power Supply 2.5A
Mouser.com ($65.93/EA)
--- OR ---
TDK Lamda Rail Power Supplies 120W 12V 10A 380-480VAC
Mouser.com ($104.04/EA)

6" of DIN rail, slotted, 35mm, 10 mm height
For mounting Power Supply and Solid State Relay of each Pump
AutomationDirect.com ($8.00/2x 1-Meter Pieces)

(1) Solid state relay, 35mm DIN-rail, 3-32 VDC input,
SPST, N.O., SCR, 10A, 480 or more VAC load voltage, Random type
AutomationDirect.com ($24.50/EA)

(1) Fulree DC 8-35V to 1.5-24V Adjustable Buck Converter Power Supply Voltage Regulator
*** For 24Vdc Power Supply purchases ONLY ***
Banggood.com ($7.99/EA)

B. Electronic Control ($39.10)



(2) UNO R3 Atmega328P Development Board with Male Pin Header and USB Cable

AliExpress.com ($3.80/EA)
(1) for each HandRemote
(1) for each PumpController

(1) Keypad Shield Blue Back-light for Arduino Robot LCD 1602 Board

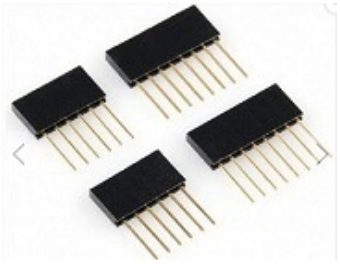
AliExpress.com ($2.19/EA)
(1) Display for each HandRemote
Wiki
Schematic
Layout

(2) Zigbee Shield RF Wireless Module Expansion Board for Arduino Xbee

Banggood.com ($4.83/EA)
(1) for each HandRemote
(1) for each PumpController
Schematic
Tutorial

(15) 2x Pin Plug-in Screw Terminal Block Connector
5.08mm Pitch External I/O connections

Banggood.com ($2.15 / 20pcs)
(15) for each PumpController

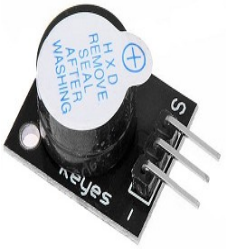


(1) 2-PIN Male – Zigbee → Perfboard POWER
(1) 4-PIN Female – Perfboard → Zigbee POWER
(2) 6-PIN Male – Zigbee I/O → Perfboard
(1) 4-PIN Female - Zigbee->Perfboard RX/TX

Aliexpress.com ($0.79 / 10pcs)
(2) Female, (2) Male for each PumpController
** Male Headers are part of Arduino Purchase **

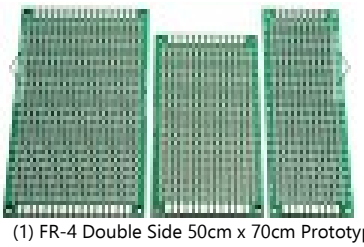
(1) Black Snap-in On/Off Rocker Switch

Aliexpress.com ($0.86 / 5pcs)
(1) for each HandRemote



(1) Active Speaker Buzzer Alarm

Aliexpress.com ($0.60 / ea)
(1) for each HandRemote



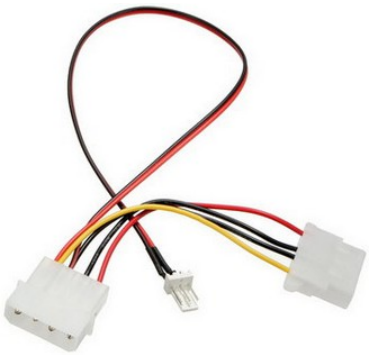
(1) FR-4 Double Side 50cm x 70cm Prototype PCB Printed Circuit Board

Aliexpress.com ($1.56/ 4pcs)
(1) for each PumpController

(2) Resistors
1 valued at 1/3 the other (10K-Ohms or higher)

Banggood.com ($2.28 / 300pcs)
(2) for each HandRemote

(1) Right-Angle 9Vdc Power Plug for Arduino Power Jack
Amazon.com ($4.51 / 10pcs)



(1) Computer power connector
Banggood.com ($1.99 / ea)

(5) #4 Small Screws no longer than 10mm or bigger than 3.5mm for Display and Case covers
https://www.boltdepot.com/Sheet_metal_screws_Phillips_pan_head_Zinc_plated_steel_4.aspx ($0.05/ea)

C. RF Electronics ($80.88)

(2) Xbee-Pro 900Hp S3B Digimesh, 900Mhz, 250Mw, Rpsma Connector, 200Kbps

https://www.arrow.com/en/products/xbp9b-dmst-002/digi-international ($36.11/EA)
(1) for each HandRemote and (1) for each PumpController

(2) Antenna Helical 3dB Gain 900MHz

https://www.arrow.com/en/products/w1063/pulse-electronics-corporation ($4.33/ea)
(1) for each HandRemote and (1) for each PumpController

D. Wire ($26.70)

(~30-feet) 24-AWG outdoor direct burial wire

Amazon.com ($11.50 / 50ft)
Enough to connect all accessory/power devices to the Pump-Controller

(~10-feet) **600V** 12-AWG to 16-AWG, 75-deg-C wire

Amazon.com ($9.18 / 25ft)
Enough to connect RHINO P.S., SSR and Exit Panel
Spec Sheet

(Various) Solder-less Terminal Connectors

Banggood.com ($6.02 / 300pcs)
Banggood.com ($9.69 / 100pcs)
Any easy way to reliably connect wires

E. Tools Required ($NA)

- 3D-Printer (*may be optional if other casing solutions are available*)
- Soldering Iron / Solder / Connecting Wire
- Screw Driver

2. HAND-REMOTE

The HandRemote is the remote controller unit used to control and monitor an irrigation pump remotely.

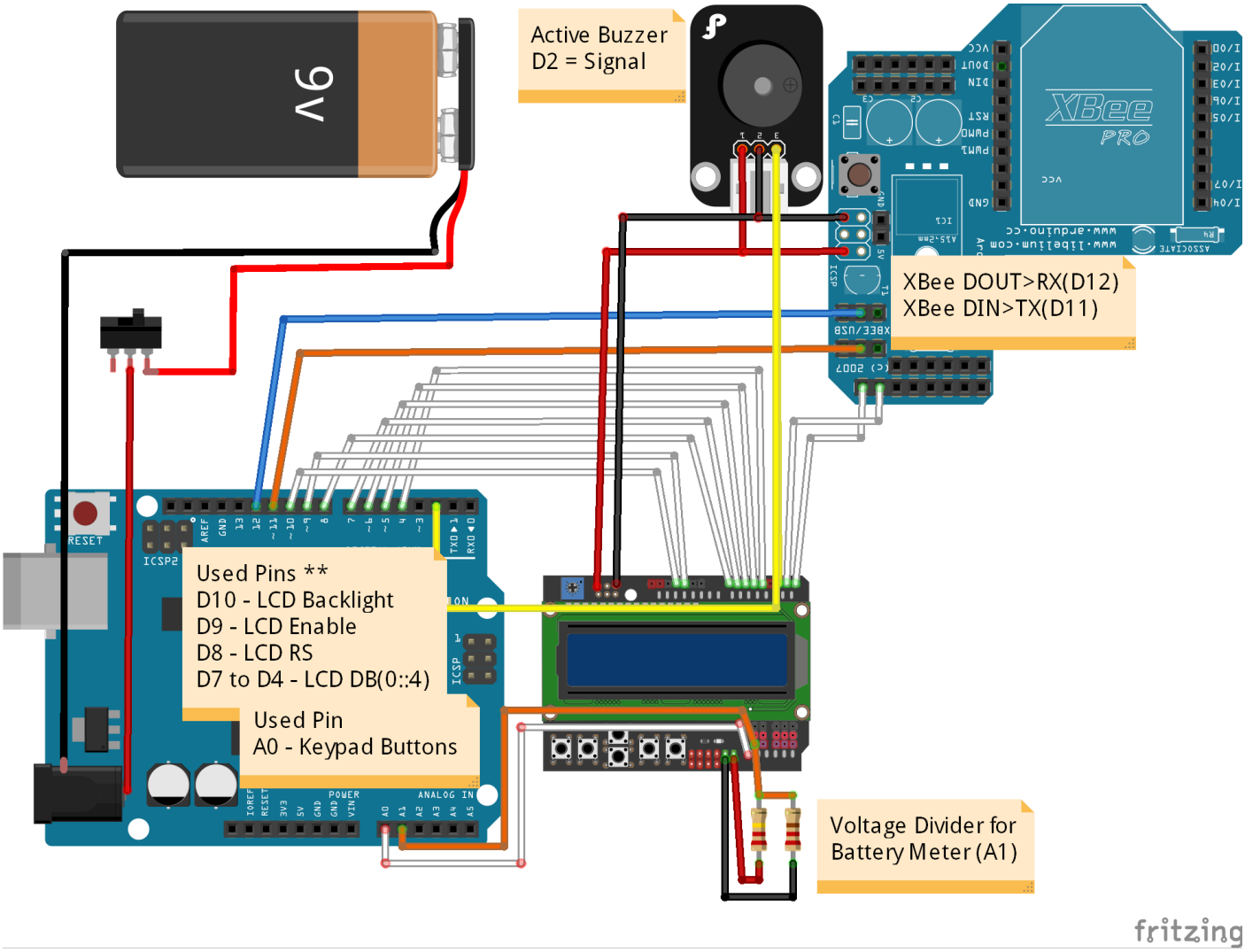
- ✔ Features
  - Buzzer alarm settings for monitored values

- Battery level indicator (9V-Battery Powered)
- Programmable Interface through USB

2.1 Electronics Assembly

✔ The HANDREMOTE uses the following electronic parts

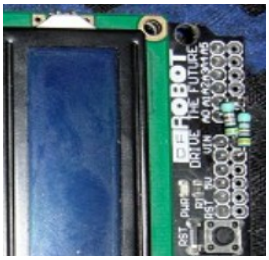
- 1- Arduino UNO R3 Board
- 1- LCD Keypad Shield
- 1- Zigbee Shield
- 1- Xbee-Pro 900Hp S3B Module and High Gain Helical Antenna
- 1- 9Vdc Battery Connector
- 1- Active Buzzer
- 2- Resistors



A. LCD Keypad Shield

1. Wire the Battery-Meter Voltage-Divider
   a) Select (2) Resistors; One-Value being 1/3 the other (10K-ohm up to 1M-ohm is recommended)
   b) Solder First Resistor (Full Value) on the top of the LCD Shield from **VIN** → **A1**
   c) Solder Other Resistor (1/3 Value) from **A1** → **GND** (Solder hole just left of VIN)

d) Snip off the excess back-side leads



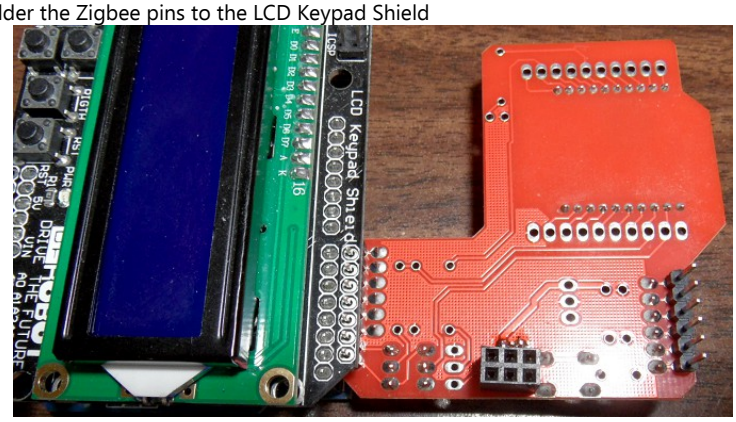
The two resistors setup a voltage divider from the 9Vdc battery (which we will attach to VIN). Analog inputs on the Arduino should never exceed 5Vdc. The voltage divider will split the battery voltage to 1/3 its actual voltage. This will allow the hand-remote to monitor the battery level. The larger the resistor values the less drain on the battery; but too-large a value resistors may affect accuracy.

2. Plug the LCD Keypad Shield onto the Arduino UNO

B. Zigbee/Xbee Shield

1. Remove Header Pin **D7** From Zigbee Shield
   a) Using Needle Nose pliers clamp onto Pin D7 of the back of the Zigbee Shield
   b) Using Soldering Iron – Heat the top of the pins solder pad and pull the pin from the board

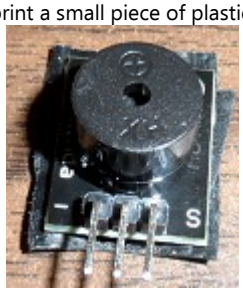
2. Solder Zigbee Shield to the LCD Keypad (Flipped upside down)
   a) Insert pins D0 → D6 (D7 removed in Step #1) into the Top Solder holes of the LCD Keypad Shield
   b) Position the Zigbee Shield flat with LCD Shield and Up against the LCD Shield – Requires bending the header pins a little (**with UNO Attached**)
   c) Solder the Zigbee pins to the LCD Keypad Shield



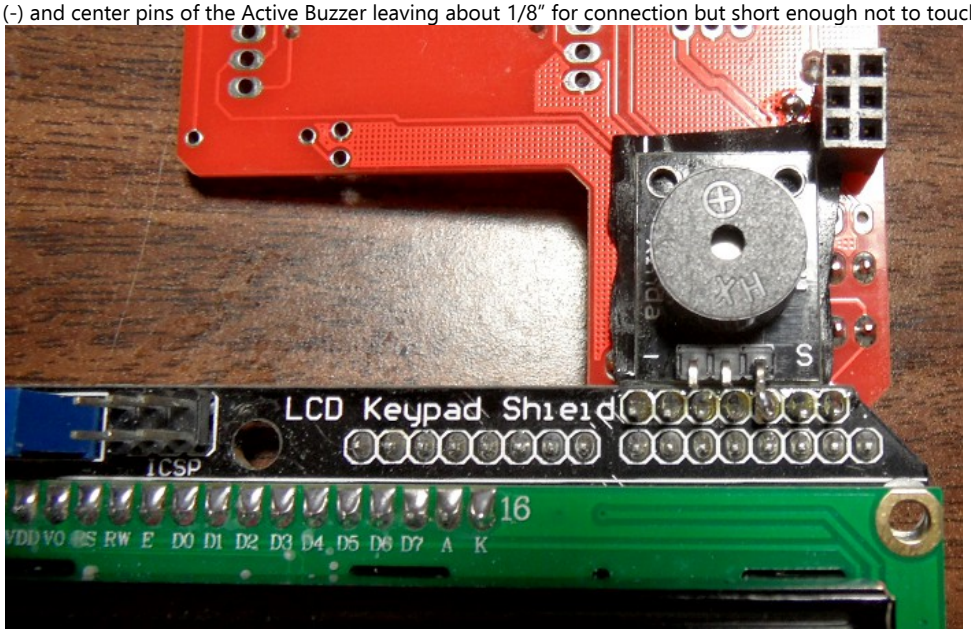
✔ Further Reading for the Curious
- https://www.arduino.cc/en/Main/ArduinoXbeeShield
- https://www.arduino.cc/en/Guide/ArduinoXbeeShield

C. Active Buzzer

1. Solder Active Buzzer to LCD Keypad Shield
   a) Either print a small piece of plastic or add Black tape to the Back-Side of the Active Buzzer to prevent shorting against the Zigbee Shield.



   b) Solder the **'S' pin** on the Active Buzzer to **D2** (3rd Left Pin – LCD Face Up) on the LCD Shield
   c) Cut the (-) and center pins of the Active Buzzer leaving about 1/8" for connection but short enough not to touch D3 & D4.



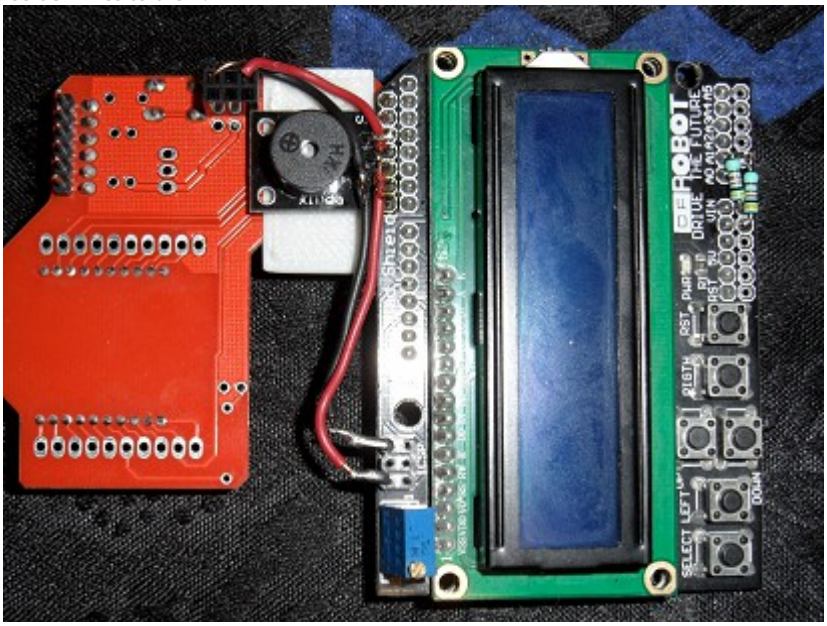
2. Wire Power to the Active Buzzer and Zigbee Shield

a) Bend the first and last pins closest to the edge of the LCD Keypad Shields ICSP pin header

b) Solder one end of a **Red** wire → Left-Most Pin of the LCD Keypad Shields ICSP pin header (**5V+**)
- Pull wire to **Active Buzzer (+)** Pin (Center Pin), strip insulation and solder
- Pull wire to Lower-Right of the **Zigbee ICSP (5V+)** plug, strip insulation, cut to length and insert into plug

c) Solder one end of a **Black** wire → Right-Most Pin of the LCD Keypad Shields ICSP pin header (**5V+**)
- Pull wire to Active **Buzzer Left-Pin (-)**, strip insulation and solder
- Pull wire to Top-Right of the **Zigbee ICSP (GND)** plug, strip insulation, cut to length and insert into plug

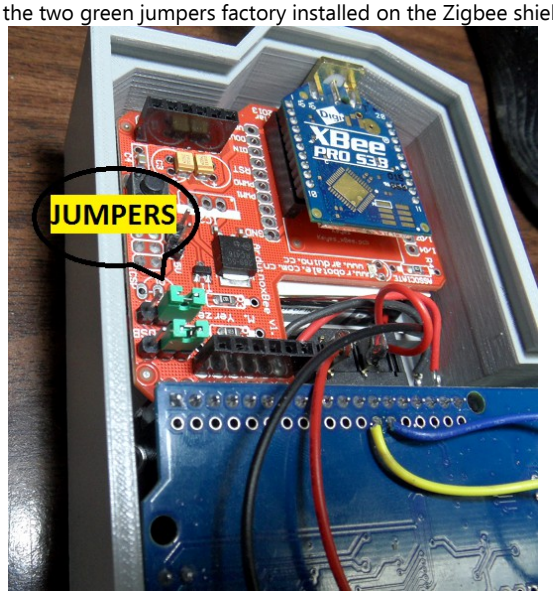
Red/Black wires were not cut but the insulation was split using automatic wire strippers. The Center and Left Pins of the Buzzer were cut just long enough to solder wires to them.



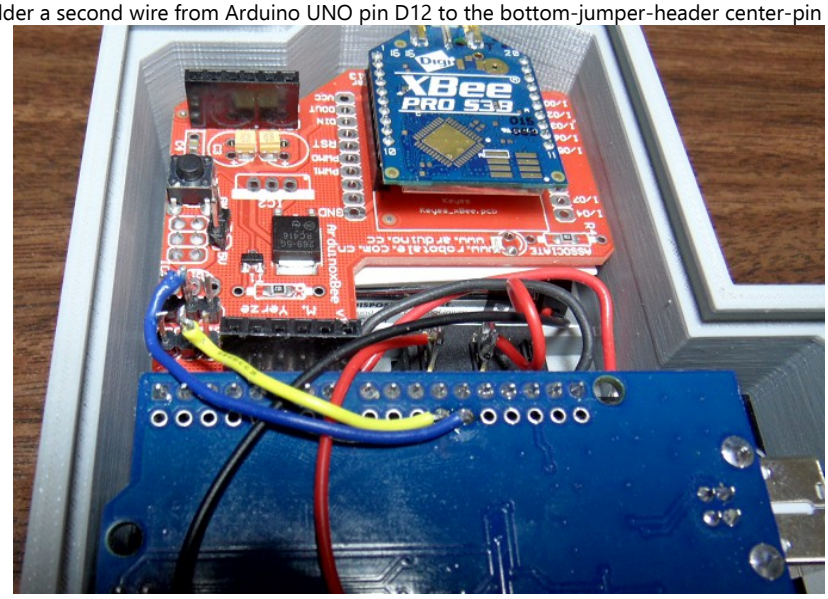
3. Cut the remaining ICSP pins as short as possible to allow the unit to fit inside the case

D. Communications

1. Remove the two green jumpers factory installed on the Zigbee shield



2. Connect Arduino UNO pins D11 & D12 to the Zigbee Communication Header
    a) From an upside-down view; left-top of Arduino UNO; Count the pin holes to the right D0,D1,D2.... to D11
    b) Solder a wire from Arduino UNO pin D11 to the top-jumper-header center-pin of the Zigbee shield
    c) Solder a second wire from Arduino UNO pin D12 to the bottom-jumper-header center-pin of the Zigbee shield



✔ Further Details for the Curious

- Xbee and Arduino both use Serial Communications –- **but** they both cannot use the same Serial Port at the same time.
    - In order to Upload Firmware to the Arduino the Xbee must be disconnected from the lines (No Jumpers)
    - In order to Connect to the Xbee through USB Serial the Arduino Microcontroller must be disconnected (*UN-usable for our Arduino UNO*)
    - Thus; We add Jumper-wires to put the Xbee on a different "Serial Port" when we need access to both Serial Devices.

- The Factory Jumpers on the Zigbee Shield come with two modes of operations
    - Xbee Position = Xbee UART Serial connects to Arduino Serial
    - USB Position = Xbee UART Serial connects to USB Serial (Computer); Requiring the removal of the micro-controller.

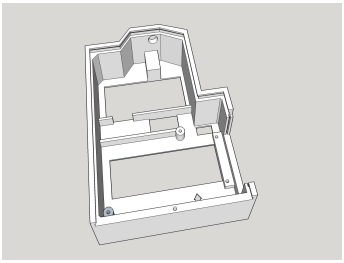
- Jumper-Wires for Debug
    - Computer USB Serial → Arduino - When NO jumpers exist; the Xbee is disconnected from RX/TX Pins and thus allows clean USB Serial Monitor.
    - Routes Xbee UART → D11 & D12 - The SoftwareSerial Library allows UART communications on D8 & D9 which is wire-jumped to the Xbee RX/TX.

- Pin Orientation
    - The Jumpers on the Zigbee Shield are aligned Top->Bottom Exactly the same as the Xbee Plug labeled DIN/DOUT are aligned
        - On the shield (DIN is the closest jumper to Digital Pins Header on the Shield)

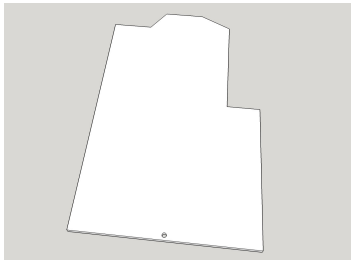
2.2 Hardware Assembly

A. 3D-Prints

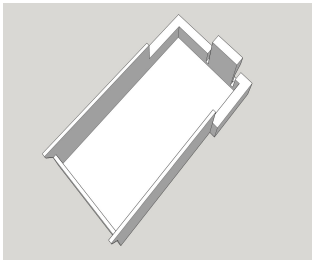
✔ Print the Following Models on a 3D-Printer
- Files Located at https://github.com/tgit23/AgIrrigationRemoteControl/tree/master/HandRemote/3D-Prints-STL
- Total Filament = 83.8 cm^3 @ $0.03/$cm^3$ ~ $2.50



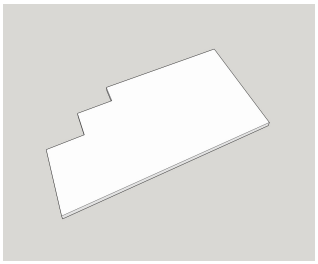
HandRemote-Case.stl (56.4 cm^3)



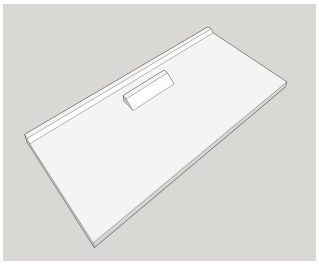
HandRemote-BackCover.stl (19.1 cm^3)



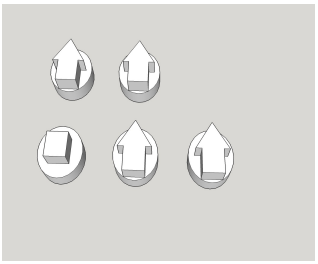
HandRemote-BatteryCover.stl (4.4 cm^3)



HandRemote-BatteryInsulatorPlate.stl (1.5 cm^3)



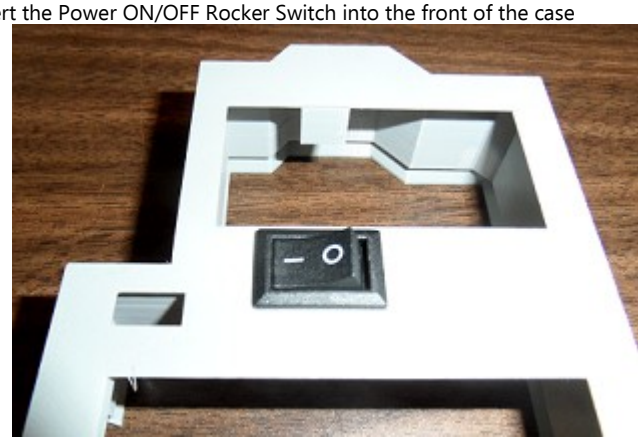
HandRemote-SideCover.stl (2.0 cm^3)



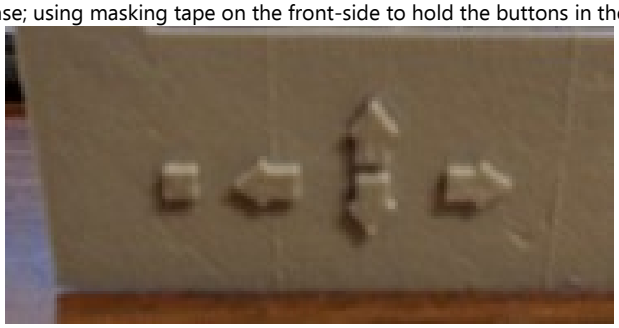
HandRemote-Buttons.stl (0.4 cm^3)

B. Install Electronics

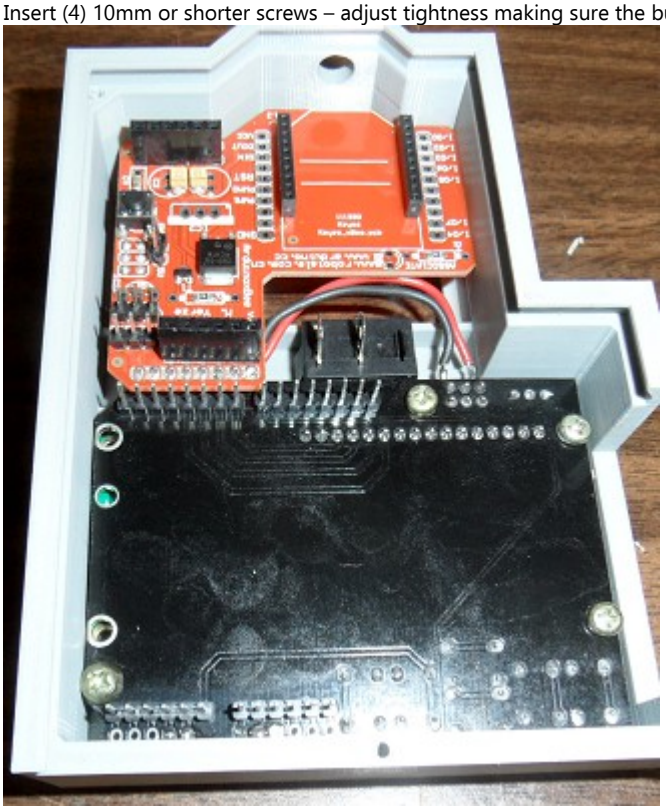
1. Insert the Power ON/OFF Rocker Switch into the front of the case



2. Place the buttons in their proper place inside the case; using masking tape on the front-side to hold the buttons in their place



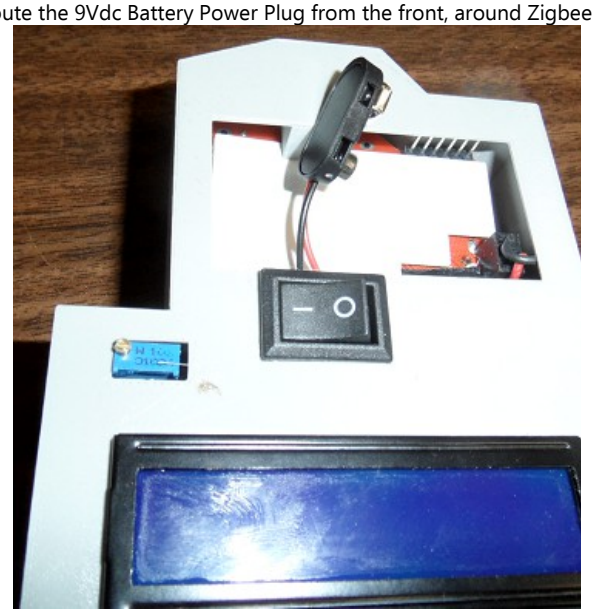
3. Install the LCD, Buzzer, Zigbee **Electronics Assembly** (from Step #2.1) into the Case (Arduino UNO Removed)
   a) Insert (4) 10mm or shorter screws – adjust tightness making sure the buttons still click (*too tight will pinch buttons down permanently*)



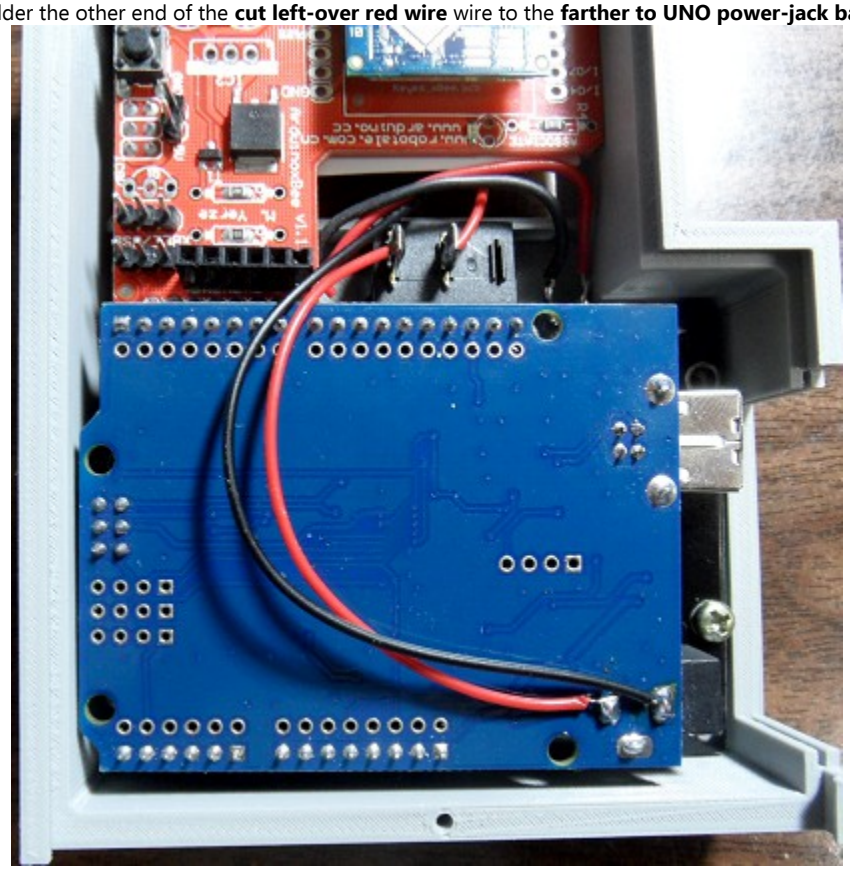
4. Plug the Arduino UNO back onto the LCD Keypad Shield
5. Slide the Xbee into the Top-Hole and then plug into the Zigbee Shield
6. Attach the Antenna to the Xbee RPSMA Connector

C. Wire in Battery Power

1. Install Plugs
   a) Insert the HandRemote-BatteryInsulatorPlate.stl 3D-Print inside the battery hole
   b) Route the 9Vdc Battery Power Plug from the front, around Zigbee shield, to the back



2. Wire in the Switch and Power to the Arduino UNO
   a) Solder the **Black wire** to the **closest to UNO power-jack backside pin** (as pictured)
   b) Leaving an inch-or-so slack on the 9VDC plug; measure and cut the **Red wire** on the **closest rocker-switch connector**
   c) Strip the end and solder the wire to the rocker switch connector
   d) Strip the end of the **cut left-over red wire** and solder it into the **"other" connector on the rocker-switch**
   e) Solder the other end of the **cut left-over red wire** wire to the **farther to UNO power-jack backside pin** (as pictured)



3. Check for Power
   a) Plug in a 9V-Battery into the 9V plug
   b) Flip the rocker-switch to power-on (marked with a "-") and verify that the LCD display lights

4. Attach Covers to Case
   a) Insert the HandRemote-BatteryCover.stl 3D-Print into the cases battery-hole and snap down
   b) Slide the HandRemote-BackCover.stl 3D-Print in from the bottom then insert a screw in the bottom-center
   c) Slide the HandRemote-SideCover.stl 3D-Print in the USB side until it snaps into place

2.3 Operating Instructions

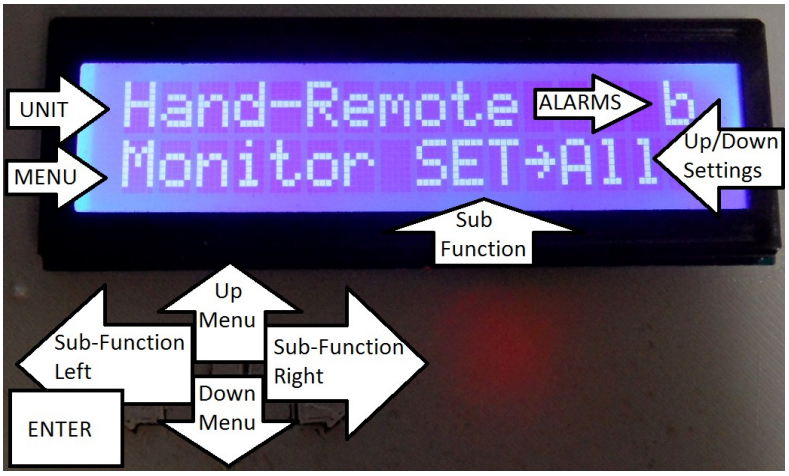
Notice: Firmware must be uploaded to a newly assembled unit before it can operate; visit #4.1.A SYSTEM OPERATIONS| Firmware | Uploading for details

A. Introduction

✔ The display has a Top Row and a Bottom Row that are adjusted using (Arrow Buttons or Menu-Items as shown below in parenthesis)

- Top Row
  - Left (*Pump Set*) - The Unit that is currently under control or being monitored
  - Right (*Alarm Enter*) - The Alarms for the current unit that are 'ON' (Identifying letters)

- Bottom Row
  - Left (*Up-Down*) - The current Menu-Item of the Unit
  - Middle (Right-Left) - The Current Sub-Function of the Menu-Item
    - (=) - The Read Value for the current Menu-Item
    - SET→ - Set a Value for the current Menu-Item
    - (!=) or (!<) - Alarm boundary; Activate alarm when value is either equal-to an option or is less-than a number
    - (!≠) or (!>) - Alarm boundary; Activate alarm when value is either not-equal to an option or is greater-than a number
  - Right (Up-Down) - The Value (of or for) the Unit → Menu-Item's → Sub-Function



✔ Maneuvering the display items can be represented as in the table shown below in Blue
- The Hook-Up PIN Column is NOT part of the Display Menu; It shows where the items are **wired** to the Pump-Controller
- Disabled items are Skipped when maneuvering through the display table
- The 'Wiring' Column shows how each Menu-Item corresponds to its controlling or monitoring hardware

		SUB-FUNCTIONS (←LEFT || RIGHT→)				Controller-Wiring
	UNIT	MENU ITEMS (=) Up/Down Items ENTER to refresh	SET (→) Up/Down Changes ENTER to Apply	LOW ALARM (!=) (!<) Up/Down Changes ENTER Toggles On-Off	HI ALARM (≠) (!>) Up/Down Changes ENTER Toggles On-Off	Wire to PIN (NOT for Display)
(DOWN || UP)	Hand-Remote	Monitor	(SET→) Sel (SET→) All	DISABLED	DISABLED	
	Hand-Remote	Pump	(SET→) Ditch (SET→) Canal	DISABLED	DISABLED	
	Hand-Remote	Battery(B)	DISABLED	(!<) When less than	(!>) When greater than	A2
	Ditch Pump	Power(P)	(SET→) On (SET→) Off	(!=) Equals "On" (!=) Equals "Off"	(! ≠) Not Equal "On" (! ≠) Not Equal "Off"	
	Ditch Pump	Water (L)	DISABLED	(!<) When less than	(!>) When greater than	V64(Trig=D6,Echo=D5
	Ditch Pump	Pressure(R)	DISABLED	(!<) When less than	(!>) When greater than	A3
	Canal Pump	Power(P)	(SET→) On (SET→) Off	(!=) Equals "On" (!=) Equals "Off"	(! ≠) Not Equal "On" (! ≠) Not Equal "Off"	D7
	Canal Pump	Pressure(R)	DISABLED	(!<) When less than	(!>) When greater than	A3
	Canal Pump	Pressure(S)	DISABLED	(!<) When less than	(!>) When greater than	A4

** Note: V64 is Virtual Pin #64; Virtual Pins are memory spaces inside the Pump-Controller and therefore the Pump-Controller firmware actually identifies the connecting pins. This is used when Pump-Controller processing is done before a legitimate value can be determined.*

B. Unit Selection

✔ The following Units are identified by default firmware
- Hand-Remote - The unit with the display being described here; This unit is **NOT select-able**
- Ditch Pump - Setup as a Pump-Controller with Transceiver ID #10
- Canal Pump - Setup as a Pump-Controller with Transceiver ID #11

✔ The Unit is selected using the (Hand-Remote → Pump) Menu-Item.
- Press Up/Down until display reads (Hand-Remote / Pump = ??)
- Press the Right-Arrow to [SET] (Hand-Remote SET → ??) the Pump to control and/or monitor
- Press Up/Down to to PICK the Value to [SET]
- Press the Square-Button (Enter) to apply the Setting

C. Monitoring

✔ Monitoring is required for checking readings against alarm settings
- Monitoring is automatically started when no buttons on the Hand-Remote have been pressed for 15-Seconds
- Monitoring is seen on the display as Menu-Items are iterated from one item to the next
- Only Menu-Items that are configured to have attached alarms are monitored

- ✔ Monitoring type selection
  - ○ The Hand-Remote can monitor the currently selected unit (Hand-Remote / Monitor SET → Sel)
  - ○ OR can monitor all of the defined units (Hand-Remote / Monitor SET → All)

D. Alarms

- ✔ Alarms check monitored values against desired values (Low / High boundaries) and notifies the user (buzzing) if boundaries are violated
  - ○ Some menu items will have alarms while others will not – this is configured in the firmware and should be on a use-case scenario
  - ○ Anytime a button is pressed an active alarm (buzzing) with quite until the monitored value is read again and violates the boundaries
  - ○ To permanently shut off the active alarm (buzzing) the alarm itself must be turned 'OFF'

- ✔ Alarms can be either turned ON or OFF
  - ○ ON alarms are identified by the single character in the Top-Right of the Display
  - ○ Alarms are switched from ON → OFF or OFF → ON by
    - ▪ Press UP/DOWN until the Menu Item you'd like to turn an Alarm On/Off for - is selected
    - ▪ Press RIGHT button to select which alarm to turn On/Off; Either the Low Alarm (!=, !<) or the High Alarm (! ǂ, !>)
    - ▪ Press ENTER button to Toggle the Alarm On/Off (An On-Alarm goes Off; An Off-Alarms goes On)

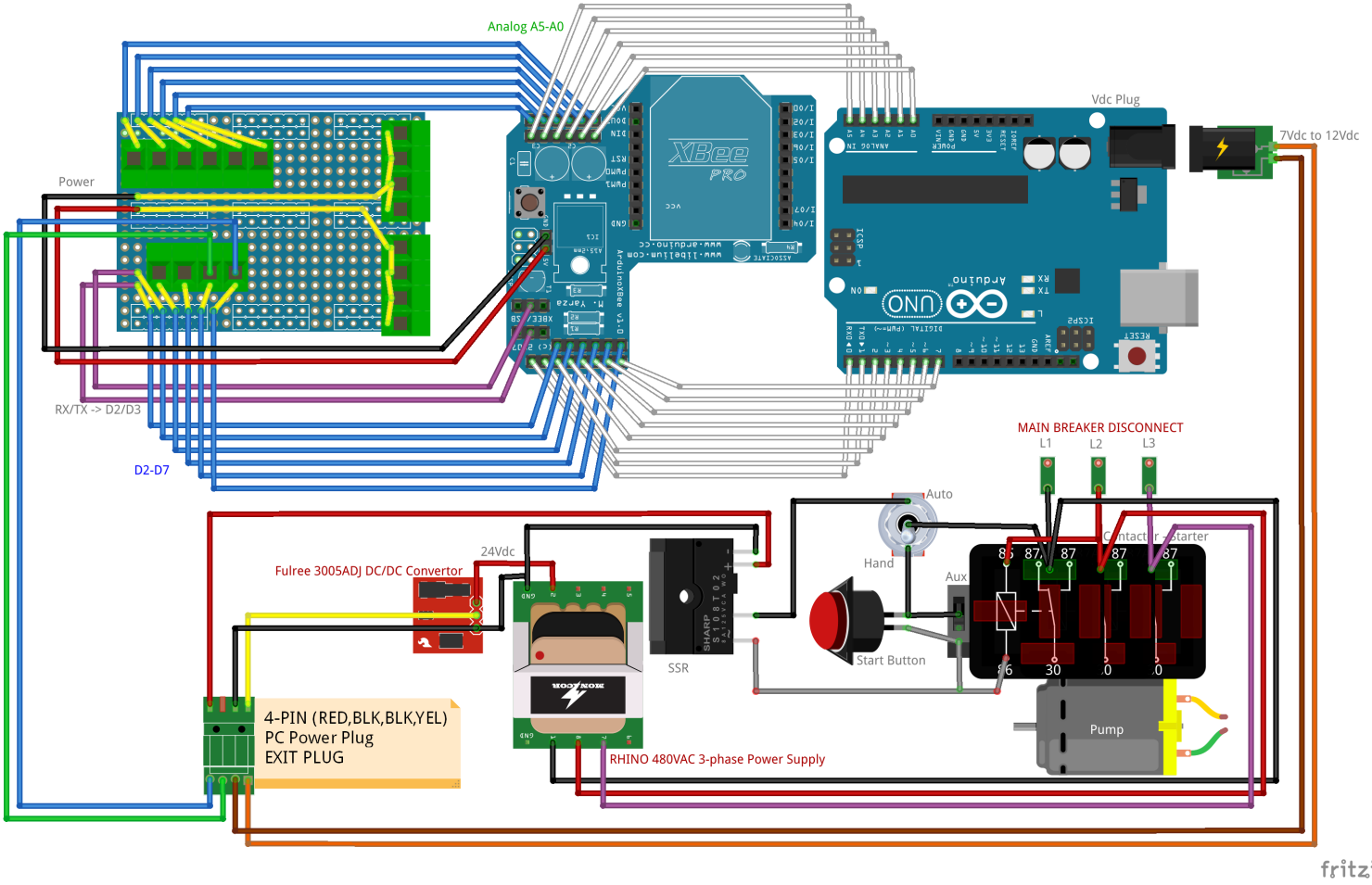
- ✔ To Set an alarm **value**
  - ○ Press UP/DOWN until the Menu Item you'd like to set an alarm boundary for is selected
  - ○ Press the RIGHT button as many times as it takes to see one of the following indicators in the Middle of the Bottom Row
    - ▪ (!=) - Sound an alarm if the value becomes EQUAL to the one indicated
    - ▪ (!ǂ) - Sound an alarm if the value does NOT-EQUAL equal the one indicated
    - ▪ (!<) - Sound an alarm if the value becomes LESS-THAN the one indicated
    - ▪ (!>) - Sound an alarm if the value becomes GREATER-THAN the one indicated
  - ○ Use the UP/DOWN buttons to select the comparison value or option
  - ○ Press ENTER button to apply the selected comparison value (Note; Applying a value will also Toggle the On/Off status of the Alarm)

- ✔ Note: When the Hand-Remote looses signal the alarms are shut-off automatically until signal is restored

3. PUMP-CONTROLLER

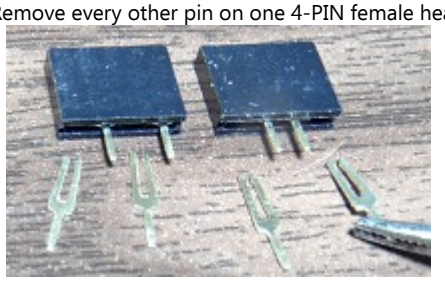
✔ Automation
- ○ Automatically turn off the pump if no water exists
- ○ Set to Automatically turn on pump when water height is okay

Wiring Diagram of the PUMP CONTROLLER - (Screw Terminal Perfboard → Zigbee Shield → Arduino UNO)

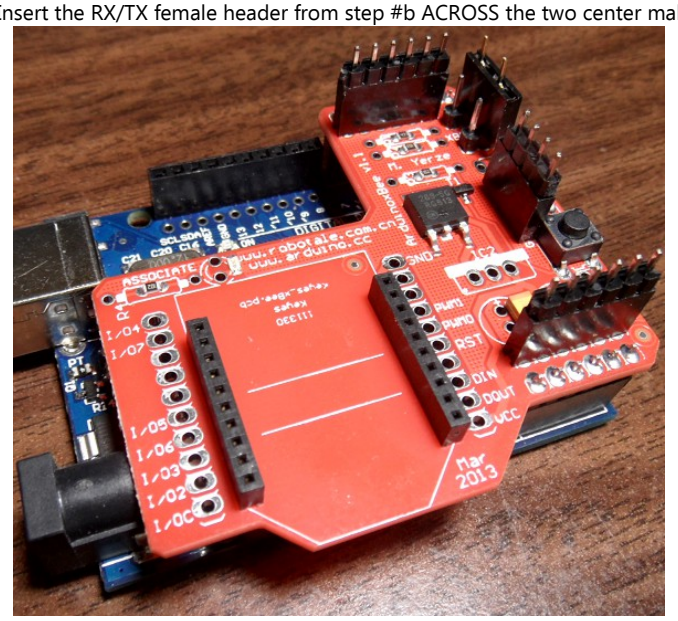


3.1 Perfboard Assembly

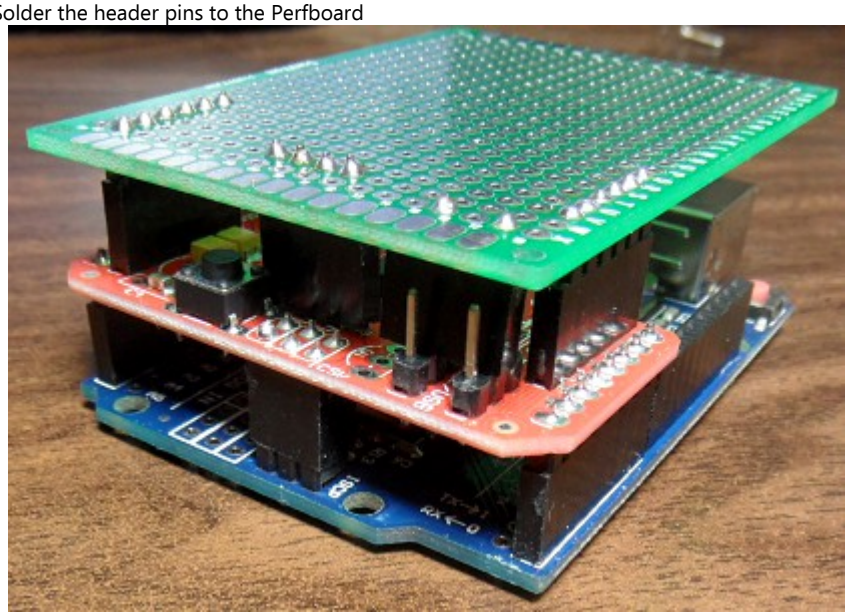
1. Install Headers
   a) Remove outer pins on one 4-PIN female header (POWER)
   b) Remove every other pin on one 4-PIN female header (RX/TX)



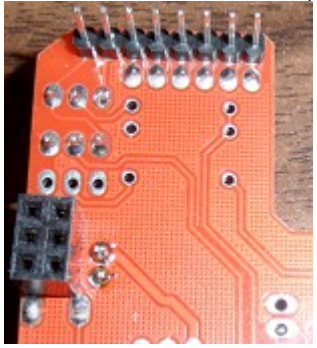
c) Remove the (2) Green Jumpers on the Zigbee Shield and plug it onto the Arduino UNO
d) Plug a 6-Pin Male Header into the Analog Header of the Zigbee Shield (A5 → A0)
e) Plug a 6-Pin Male Header into the Digital Header of the Zigbee Shield (D2 → D7)
f) Insert the POWER female header from step #a into the Zigbee Shield holes labeled 5V & GND (next to the reset button)
g) Plug a 4-Pin Male Header into the top of the POWER female header
h) Insert the RX/TX female header from step #b ACROSS the two center male header (XBEE/USB) pins of the Zigbee Shield



2. Solder the header pins to a 5cmx7cm 2.54mm pitch Perfboard
a) Align and place the Perfboard onto the header pins
b) Solder the header pins to the Perfboard



3. Solder the Zigbee female POWER header pins
a) Unplug the Arduino UNO from the Zigbee Shield leaving the Perfboard attached
b) Solder the female POWER header pins to the holes of the Zigbee shield



4. Add wires to the header pins
    a) Cut (12) wires approximately 1" long and strip one end.
    b) Cut (2) wires approximately 3" long and strip one end.

    c) Unplug the Perfboard from the Zigbee Shield
    d) Insert (6) 1" wires from the Zigbee-side of the Perfboard into the holes just **below** the analog male pin headers, bend and solder
    e) Insert (4) 1" wires from the Zigbee-side of the Perfboard into the 4-right (D4-D7) holes just **above** the digital male pin headers, bend and solder
    f) Insert (2) **3"** wires from the Zigbee-side of the Perfboard into the right of the POWER male pin headers (Center-2), bend and solder
    g) Attach (2) 1" wires from the Top-side of the Perfboard across RX/TX female pin headers and solder to D2, D3 Digital pins as shown and trim.

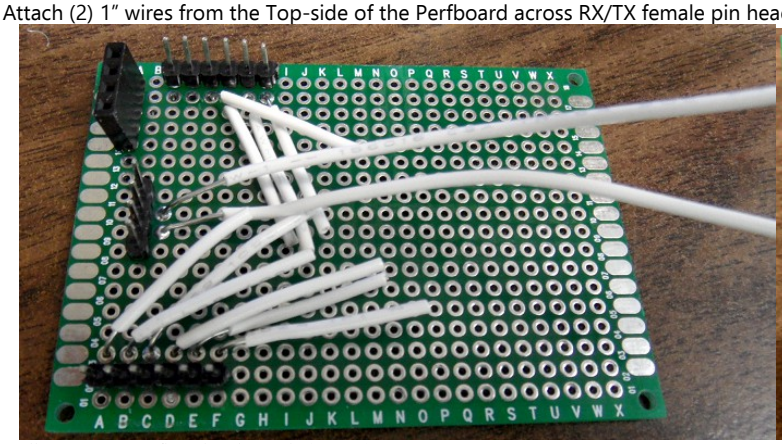


Illustration 3: Perfboard Zigbee-Side (Bottom)

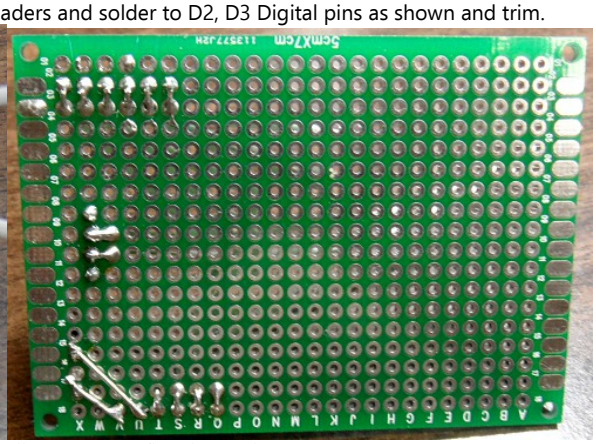
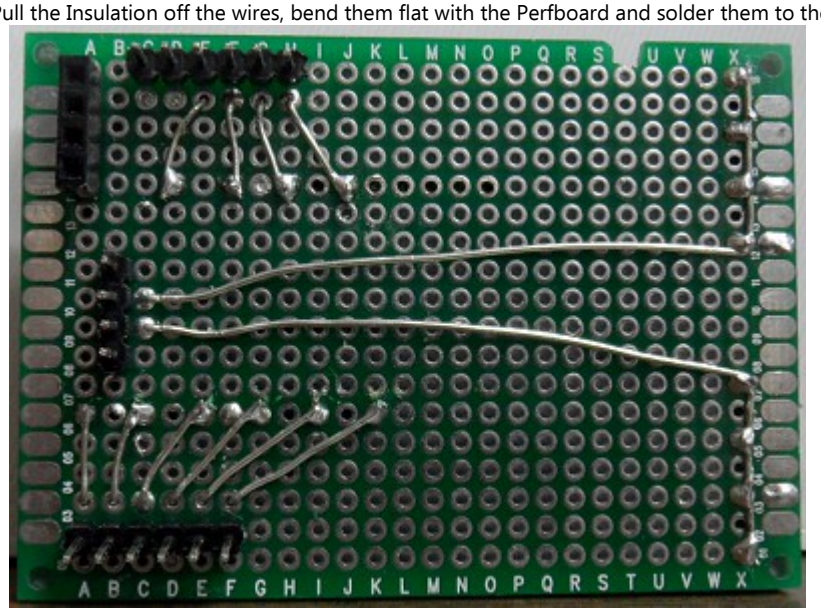


Illustration 4: Perfboard Top-Side

5. Install the Screw Terminals
    a) Place the HandRemote-PerfboardCover.stl 3D-Print on the top of the Perfboard lining up grooves with header pin solder joints
    b) Insert (3) 2x Screw Terminals along the Analog section starting at the Right and working left to allow sliding the blocks together at their edges
    c) Insert (2) 2x Screw Terminals along the Digital section with the same method above
    d) Insert (2) 2x Screw Terminals along the 5VDC section (same as above)
    e) Insert (2) 2x Screw Terminals along the GND section (same as above)

6. Solder the Screw Terminals to their corresponding header wires
   a) Turn the Perfboard over holding the screw terminals into place
   b) Sold each screw terminal pin to the Perfboard
   c) Pull the Insulation off the wires, bend them flat with the Perfboard and solder them to their corresponding pin (See picture or Wiring Diagram)



3.2 Hardware Assembly

A. Print Models (3D-Prints)

✔ 3D-Print the Following Models @ https://github.com/tgit23/AgIrrigationRemoteControl/tree/master/PumpController/3D-Prints-STL
- PumpController-Case.stl
- PumpController-Cover.stl
- PumpController-USBCover.stl
- PumpController-AntennaCover.stl

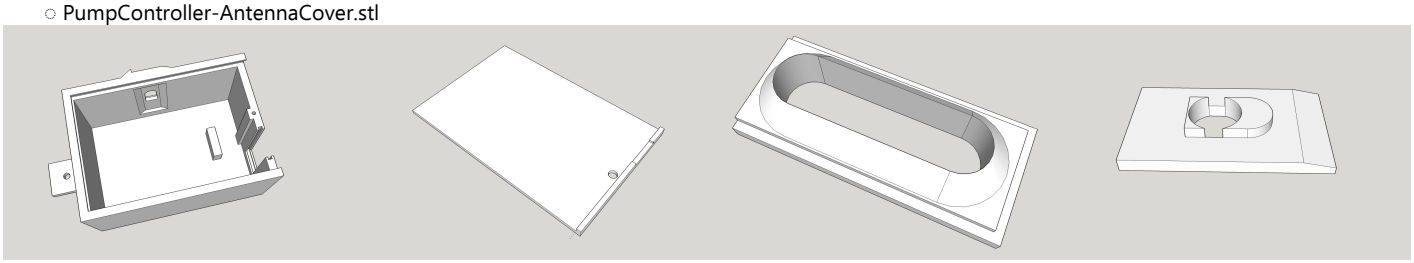
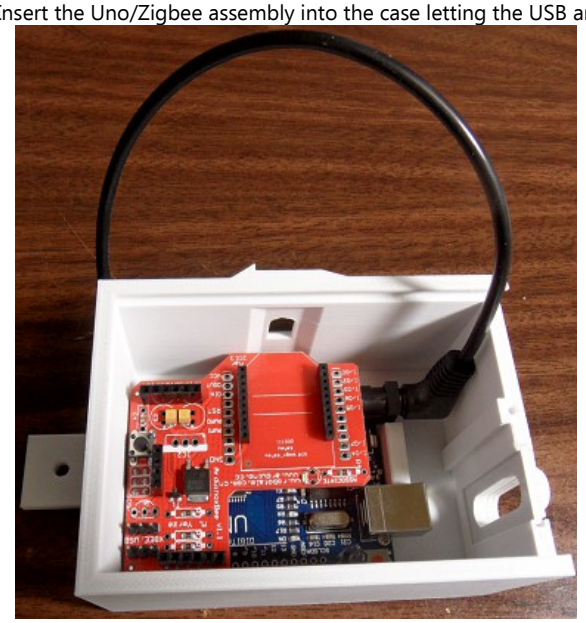


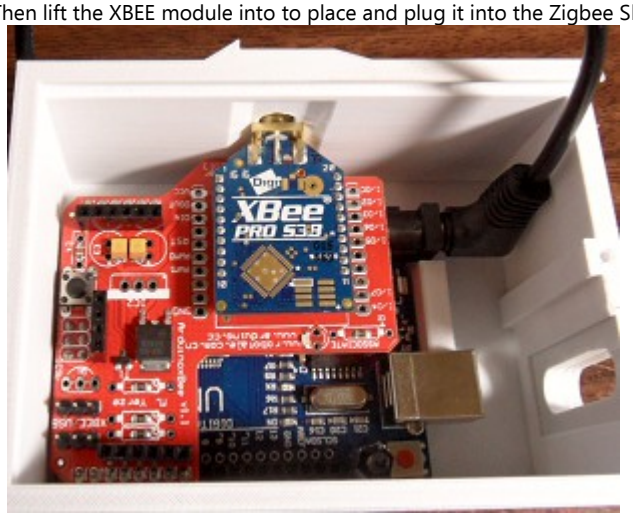
Illustration 5: Case.stl *Illustration 6: Cover.stl* *Illustration 7: USBCover.stl* *Illustration 8: AntennaCover.stl*

B. Install Electronics

1. Install UNO and Zigbee connected into the Case
   a) Remove the Screw Terminal Perfboard if attached and connect the Arduino UNO & Zigbee Shield
   b) Plug in the Right-Angle 9Vdc Power Plug for Arduino Power Jack
   c) Insert the Uno/Zigbee assembly into the case letting the USB and power Jack sit on each side of the case stub.

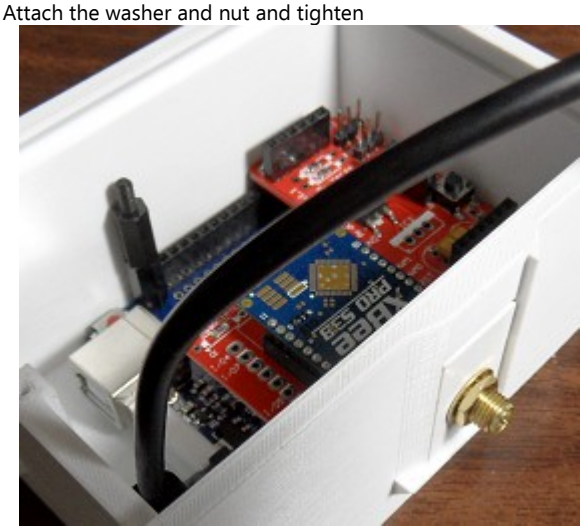


2. Install the XBEE module
   a) First slide the XBEE RPSMA Connector into the case slot
   b) Then lift the XBEE module into to place and plug it into the Zigbee Shields Xbee plug-in



3. Install the Xbee Antenna Cover
   a) Slide the ANTENNACOVER.STL 3D-Print onto the RPSMA connector from the outside of the case

b) Attach the washer and nut and tighten



4. Plug the Perfboard into the Zigbee shield

3.3 Installation Instructions

3.3.1 Prepare for Installation

A. DIN Rail Components

1. Cut to Length a portion of DIN Rail
   a) A length that will comfortably fit the Power Supply, Solid State Relay, and Fulree DC/DC Converter
   b) A length that will fit comfortably somewhere inside the Pump Power Panel
   c) Recommending approximately 5-1/2" or 4-hole slots in length

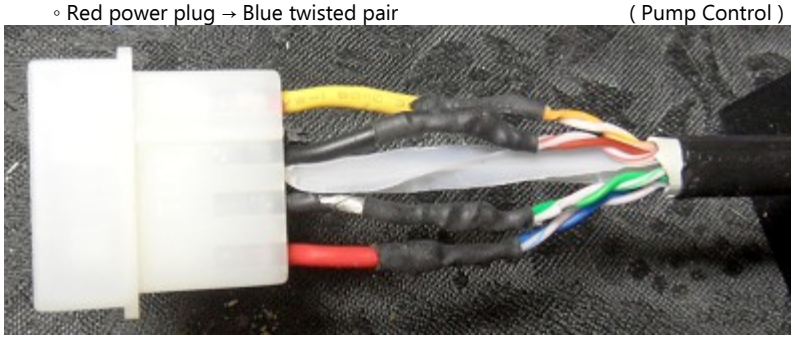
2. Wire RHINO power supply to Fulree DC/DC Converter
   a) Attach the RED(+) IN wire of the Fulree DC/DC Converter to the RHINO power supply at the top labeled (24VDC+)
   b) Attach the BLACK(-) IN wire of the Fulree DC/DC Converter to the RHINO power supply at the top labeled (24VDC-)

3. Wire GND to the Solid State Relay
   a) Attach a wire (Recommend BLACK or GREEN) from the "other" 24VDC- on top of the RHINO power supply to A1 of the Solid State Relay

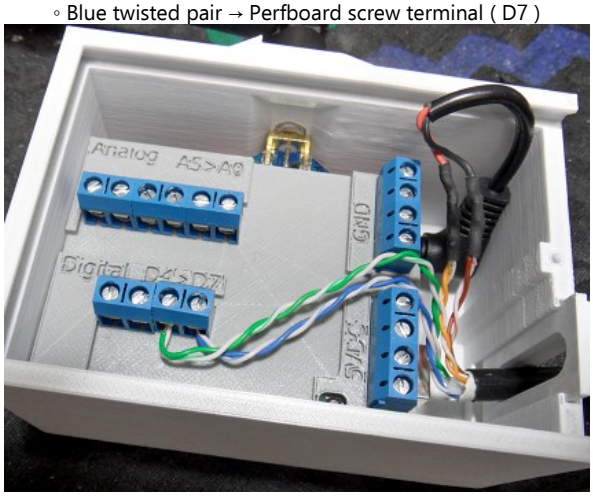
B. Pump-Controller Power Cable

1. Select a place to mount the Pump-Controller on the Pole next to the Pumps electrical panel
   a) Measure and cut a length of Cat5e outdoor cable to reach from that spot to the bottom of the electrical panel

2. Create Cable to connect the PUMP-CONTROLLER → PUMP-PANEL-**EXIT PLUG** which will be installed in 3.3.2-Section D-Step#3
   a) Separate the two connecting ends of a Computer power plug (shown in the BOM) by cutting the wires in the middle
   b) Wire up the **MALE end** of the cable and save the FEMALE end for the PUMP-PANEL-**EXIT PLUG**
      - Strip the wire ends of the Cat5e twisted pair wires
      - Strip the wire ends of the male Computer power plug
      - Solder the Computer power plug to the twisted pair wires as shown
        - Yellow power plug → Orange twisted pair (9Vdc+)
        - Black next to Yellow power plug → Brown twisted pair (9Vdc GND)
        - Black next to Red power plug → Green twisted pair (Extra Accessory Wire)
        - Red power plug → Blue twisted pair (Pump Control)



c) Wire up the Pump-Controller end (Other end of the same wire in step#a above)
- Strip the outer insulation from the Cat5e Cable approximately 4" from the end
- Strip and wire the twisted pairs as follows
  - Orange twisted pair → Red wire of the angled 9Vdc UNO Power plug
  - Brown twisted pair → Black wire of the angled 9Vdc UNO Power plug
  - Green twisted pair → Perfboard screw terminal (D6)
  - Blue twisted pair → Perfboard screw terminal (D7)



3.3.2 Wiring Power & Control

This section is a guide to installing the required power supply and pump ON/OFF switching mechanisms inside the pumps electrical panel.

✔ The following **SUPPLIES** will be needed
- DIN Rail cut to length (Recommend 5-1/2")
- The RHINO Power Supply
- The Fulree DC/DC Converter mounted to the PumpController-FulreeDINMount.stl 3D-Print
- The Solid State Relay
- 12-Gauge wire rated at 600V or more

✔ The following **TOOLS** will be needed
- Voltmeter that reads up to or beyond 500VAC
- Cordless drill
- Center tap
- Bit (#21 for #10-32 Machine Screw)
- Threading tap (#10-32 for #10-32 Machine Screw)
- (2)-Machine Screws
- Protective insulated and flash protection personal gear
- Wire strippers and cutters

BE SURE THE MAIN BREAKER IS TURNED OFF AND CAREFULLY CHECK THAT THERE IS NO VOLTAGE ON THE OUTPUT OF THE MAIN BREAKER WITH A PROPER METER!!! 480VAC is deadly; and can cause severe flash burn and/or death. (e.g. A meter on the wrong setting can "flash" explode causing "flash" burns on you even at a distance away). I DISCLAIM ANY AND ALL RESPONSIBLITY FOR ANY RISK OR HARM OCCURED BY FOLLOWING ANY INSTRUCTION IN THIS MANUAL.

See https://www.youtube.com/watch?v=6hpE5LYj-CY

A. Observe Pumps Electrical Panel

Most 3-phase irrigation pumps that do not have variable frequency drives(VFD) or inverters are setup similarly. Below is the details of two pump panels as an ***example or reference*** to the pumps power circuit. This information was gathered to determine the wiring and power needed to control the pumps power.

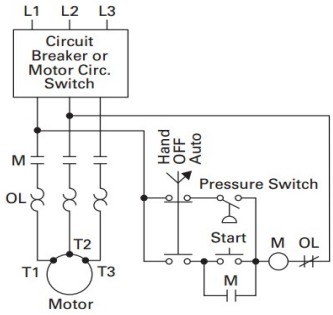
✔ Details

○ Box - Eaton Irrigation Pump Control Type 3R Enclosure Rainproof
- http://www.eaton.com/Eaton/ProductsServices/Electrical/ProductsandServices/AutomationandControl/EnclosedControl/NEMA/PumpPanels/index.htm

Type 3R Industrial Pump Panel—C440 SSOL

Wiring Diagram

Typical Wiring Diagram



○ Wiring Diagram Description
- Labels
  - L1, L2, L3 - Are the three HOT wires of the 3-Phase Power
  - M - The Starter Contactor (see Panel picture; white block with a blue center) contains:
    - Activation Coil - Shown as a circle in the wiring diagram just right of the Start Button (Activates the "Hammer")
    - Main Contacts - The Contacts that connect the L1,L2,L3 lines → OL → Motor (Sometimes called the "Hammer")
    - Hold Contact - The Contact wired in parallel with the Start Button and mounted on the Right-Side of the Starter Contactor Block

*When **HAND** is selected the "control circuit" (L1 → L2 loop) is still open until the Start Button is pressed. When the Start button is pressed it activates the (M) Coil which in-turn activates the "Hold Contact" (the --| |-- in parallel to the Start Button). This right-side contactor holds the "control circuit" closed until there is a power failure bump or the Hand-Off-Auto Switch breaks open the circuit loading the (M) activate coil.*

*This Projects SSR-Relay will be connected where the "Pressure Switch" is shown in this wiring diagram and will use the **AUTO** switch selection. This setup will require that the Arduino **NOT re-activate** power after a power failure has occurred until a specified time delay or turned back on. The click in HAND circuit is setup so the power company can re-activate power after a power failure without the over-load of all motors kicking on at the same time.*

○ Starter Contactor - Eaton-Cuttler/Hammer 45-Amp Model # AN16GNO Series B1 (Need to check Contactor Model #)
- MODEL # Decoding
  - A = Starter
  - N = NEMA
  - 1 = Non-Reversing
  - 6 = Starter w/C306 Over Load Relay
  - G = NEMA Size #2 (45-Amp)
  - N = Normally Open
  - O = Horizontal Mount
- https://www.galco.com/buy/Cutler-Hammer-Div-of-Eaton-Corp/AN16GN0AB
- http://www.eaton.com/Eaton/index.htm
- http://www.eaton.com/Eaton/ProductsServices/Electrical/ProductsandServices/AutomationandControl/ContactorsStarters/ElectromechanicalContactorsandStarters/FreedomSeries/FreedomSeries/index.htm#tabs-2

- Starter Contactor (Coil) Amperage
  - See Page 9 "AC COIL DATA" for NEMA Size #2 Shows 230VA(Volt-amperes which is VA=V*A) so 230VA/480Vac = 0.479Amps @ 480V
  - http://www.eaton.com/ecm/idcplg?IdcService=GET_FILE&allowInterrupt=1&RevisionSelectionMethod=LatestReleased&noSaveAs=0&Rendition=Primary&dDocName=998056282226
- Auxiliary Contact - Cutler Hammer Model #C320KGS1 10-Series A2 (480VAC @ 1.5 Amps?)

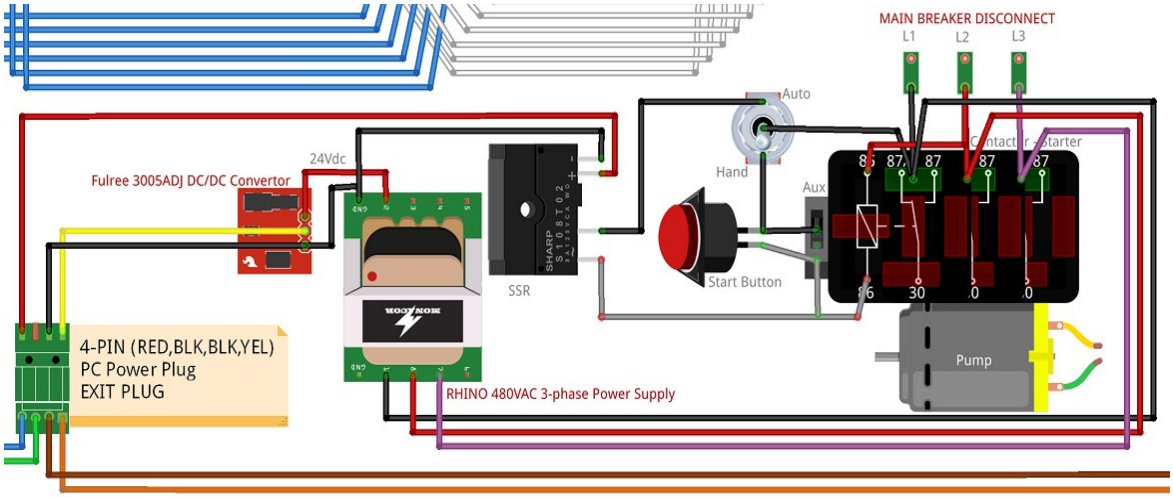
B. DIN-Rail Components

1. Drill two holes in desired location, punch, tap-thread, and mount the DIN rail horizontally
   a) Recommend using a #10-32 x 1/2" Machine Screw
   b) Drill bit size for #10-32 is #21 – 5/32" - 0.1590"

2. Snap on the RHINO power supply to the DIN rail in the center
3. Snap on the Solid State Relay to the DIN rail on the right of the RHINO power supply
4. Snap on the Fulree DC/DC Converter to the DIN rail to the left of the RHINO power supply

C. Pump Panel wiring

NOTICE: All wiring in this section should be done with 600V rated wire



1. Wire the RHINO Power Supply
   a) Connect RHINO terminal L1, L2, L3 to the top terminals of the Contactor (i.e. Starter / Hammer) L1, L2, L3 terminals
   b) Connect RHINO GRND terminal to a bold on the panels back-plane

2. Wire the Solid State Relay
   a) Connect Solid State Relay (15) → **AUTO** Terminal of the HAND-OFF-AUTO switch
   b) Connect Solid State Relay (16) → Contactor Activation Coil side connected to the Auxiliary Contact & Start Button (Grey in diagram above)
      - Follow the HAND-OFF-AUTO wire to the start button
      - Wire Solid State Relay (16) on the **OPPOSITE side** of the start button - as the wire from HAND-OFF-AUTO to the Start Button

3. Wire in the female **Exit Plug**
   a) With the **Female end** of the Computer power plug from 3.3.1.A **Exit Plug** insert wires through a bottom hole in the Pumps Electrical Panel
   b) Connect the **Yellow & Black wires from the Fulree** DC/DC converter to the Yellow/Black wires of the female end of the Computer power plug
   c) Connect the Solid State Relay **(A2) to the RED** wire of the female end of the Computer power plug
   d) Tape off the Black wire next to the Red wire (*Can be used as with an "optional" auxiliary contact to verify pumps power status*)

3.4 Optional Accessories

3.4.1 Power Alternative

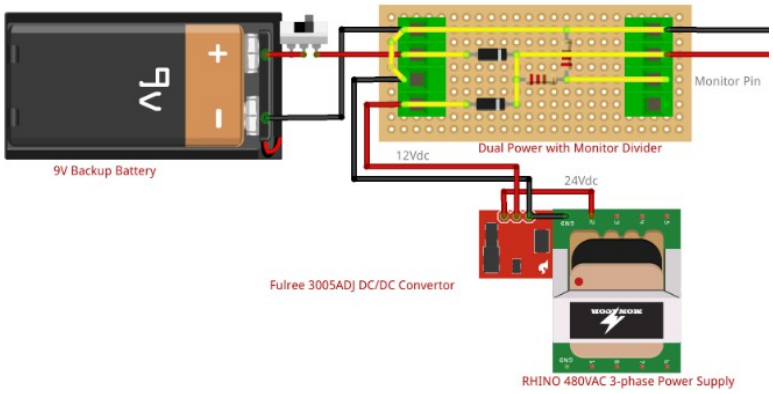
A. 9V Rechargeable Batteries ($18.90)

It is highly recommended to purchase high output 300mAH or more rechargable 9V Batteries; as the Hand-Remote will eat threw batteries pretty quickly.

At time of writing EBL 4x 600mAHh 9V Li-ion Rechargeable Batteries with Charger could be purchased for $18.90
http://www.ebay.com/itm/300974624904

NOTE: Even if a USB power is plugged in – If the power switch is turned ON it will still draw power from the battery. If you plug in USB for power be sure the power switch on the Hand-Remote unit is turned OFF.

B. Pump-Controller Backup Power



- ✔ Car Battery for Main Pump-Controller Power (Currently No Implementation Plan)
  - ○ Requires an auxiliary contact to monitor the Pumps power source so the pump can wait a while after a power bump to restart the pump.

- ✔ Emergency Backup 9V Battery Power (Currently No Implemented Plan)
  - ○ Requires either
    - ▪ An auxiliary contact to monitor the Pumps power source
    - ▪ A voltage monitor that can determine when the emergency power has started and the Pumps power source is gone.
  - ○ Advantages
    - ▪ Allows reading of water levels during a power outage
    - ▪ Allows (though risky) the automatic restarting of the pump once power is restored

***** After considering the benefit and risk of having back-up power the implementation plan was canceled for the time being***

3.4.2 Pressure Transducer

- ✔ Features
  - ○ Offers a way to remotely monitor the water pressure
  - ○ Default programming on pin A3

A. Materials ($9.59)

- ✔ 5V 0-1.2 MPa Pressure Transducer Sensor Oil Fuel Diesel Gas Water Air Sensor
  - ○ http://www.banggood.com/Pressure-Transducer-Sensor-Oil-Fuel-Diesel-Gas-Water-Air-Sensor-p-1007341.html - $9.59

B. Installation

1. Remove the current Pressure Gauge from its location

2. Screw in the Pressure Transducer (Typically does fit the same as the pressure gauge)

3. Wire up the Transducer as follows
   a) Transducer RED wire (+5V) → Cat5e ORANGE
   b) Transducer BLACK wire (GND) → Cat5e BROWN
   c) Transducer YELLOW wire (SIGNAL) → Cat5e BLUE

4. Wire the other end of the Cat5e Wire into the Pump-Controller
   a) Cat5e ORANGE → 5VDC
   b) Cat5e BROWN → GND
   c) Cat5e BLUE → A3

C. Firmware Adjustments

The current Firmware supports Pressure measured in PSI.

http://forum.arduino.cc/index.php?topic=376384.0

3.4.3 Ultrasonic Water Level

✔ Features
- Offers a way to remotely monitor the water level in a ditch
- Default programming on pins D4 (TRIG) and D5 (ECHO) – *On Pump-Controllers Firmware*

A. Materials ($10.77/ea)

- DC 5V Waterproof Ultrasonic Module Distance Measuring Transducer Sensor
  - http://www.banggood.com/DC-5V-Waterproof-Ultrasonic-Module-Distance-Measuring-Transducer-Sensor-p-1094462.html - $ 10.77

Illustration 9: Ultrasonic Distance Sensor

- (1) 4-PIN Female Dupont Header

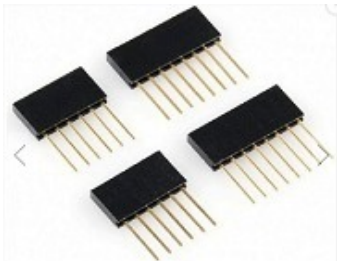


Illustration 10: Female Pin Headers

- (~30-feet) 4-lead wire required to reach from the ditch to the Pump-Controller (See Outdoor Data Wire in BOM)

B. 3D-Prints

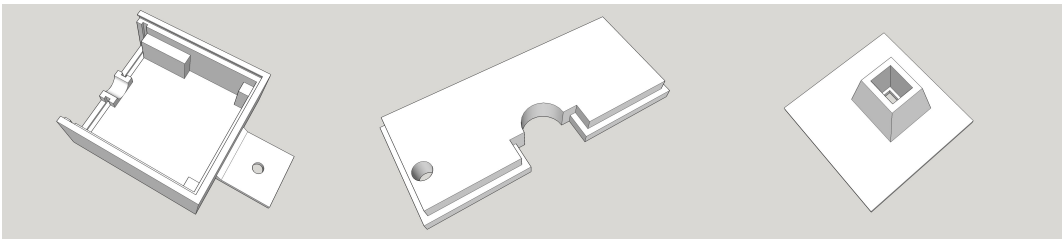


Illustration 11: SonicCircuitBoardCase.stl *Illustration 12: SonicCableCover.stl* *Illustration 13: SonicCircuitBoardCover.stl*

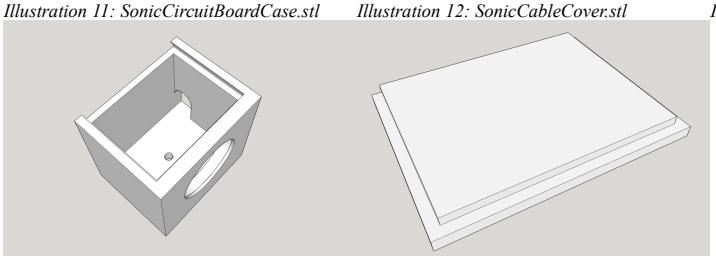


Illustration 14: SonicHeadCase.stl *Illustration 15: SonicHeadCover.stl*

C. Firmware Adjustments

Current Pump-Controller and Hand-Remote Firmware is setup to support Ultrasonic Water Level Meter

3.4.4 Auxiliary Contact ($43.75)

✔ Auxiliary contact, side mounted, 1 N.O. Contact (NOTE: This must match/fit the pump panels Contactor Unit)
- https://www.automationdirect.com/adc/Shopping/Catalog/Motor_Controls/Eaton_Cutler-Hammer_Contactors_-z-_Starters_-z-_Overloads/Auxiliary_Contacts/C320KGS1 - $43.75/ea

✔ Notes
- Auxiliary Contactor must be purchased according to the contactor they will FIT on.
  - The above suggested purchase will ONLY work with the NEMA-2 Model # AN16GNO Series B1 Contactor
- The Auxiliary Contactor typically already used by the Start-lock CANNOT be used for this purpose as it has 480V wires already attached to it.
- Often a used aux contactor can be found much cheaper on ebay

✔ Benefits
- Offers a way to monitor the actual (i.e. directly) the pump power status

3.4.5 Flow Meter

Did not implement due to high costs and limited benefits

http://www.banggood.com/TUF-2000M-TS-2-Digital-Ultrasonic-Flowmeter-Flow-Meter-Ultrasinic-Flow-ModuleRTU-p-1087832.html ($186.77)

4. SYSTEM OPERATIONS

4.1 XBEE Setup

The XBEE RF Module is setup from the factory to work without any additional changes. However, often it is very wise to assign non-factory setting in order to keep your RF communications from being interfered with by other XBEE devices in the neighborhood. Changes to the ID NETWORK ID (described below) will need to be preformed on all HAND-REMOTES and all PUMP-CONTROLLERS

1. Upload Firmware with XBEE Configuration Set
   a) To change the XBEE settings – Follow Steps #1 to #3 in #4.2.A SYSTEM OPERATIONS|Firmware|Uploading

      With the Firmware file open find the following line of code; which should be very close to the top

```
#define XBEECONFIG 0            // 1 to enter XBEE Configuration Mode
```

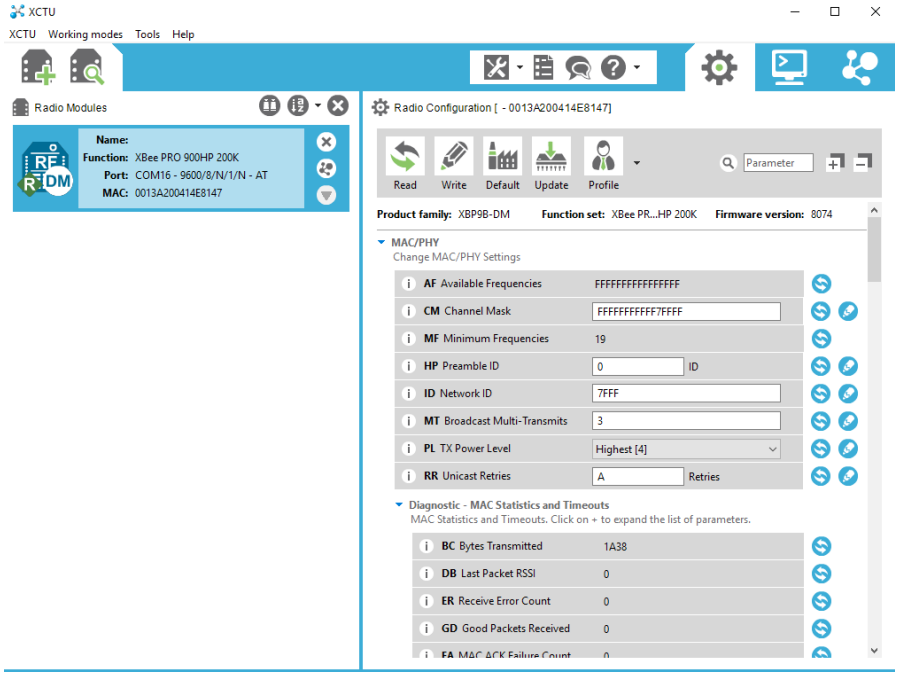
      Change it to

```
#define XBEECONFIG 1            // 1 to enter XBEE Configuration Mode
```

   b) Then proceed to upload the firmware to the Unit as outline in Step #4 in #4.2.A SYSTEM OPERATIONS|Firmware|Uploading

2. Install Digi-XCTU
   a) The Digi XCTU software can be gotten from https://www.digi.com/products/xbee-rf-solutions/xctu-software/xctu

3. Run Digi-XCTU



   a) Using the + (Add Radio Modules) mark in the top left corner enter the unit's COM port (The Arduino COM Port)
   b) The main values that may be of interest are:
      - ID Network ID - This sets a Unique "Channel" to keep your units communications separate from any others out there (Must be same on all units)
      - PL TX Power Level - If devices are close; a lower power setting can preserve battery life

4. Save Settings to the Xbee and Reload Firmware
   a) Be sure to click the "Write" button in XCTU to save any settings that were made
   b) Close XCTU
   c) In Arduino Sketch IDE restore the following line to its original

```
#define XBEECONFIG 0            // 1 to enter XBEE Configuration Mode
```

   d) Upload the Firmware to the Unit again as outline in Step #4 in #4.2.A SYSTEM OPERATIONS|Firmware|Uploading

4.2 Uploading Firmware

1. Install Arduino Sketch IDE *(If NOT installed already)* https://www.arduino.cc/en/Main/Software

2. Install the PeerIOSerialControl Library *(If NOT installed already)*
   a) In a web-browser; go to https://github.com/tgit23/PeerIOSerialControl
   b) On Green Clone or download Choose Download Zip
   c) Save to a place you'll remember – like Documents or Desktop
   d) Run the Arduino Sketch IDE
   e) Choose Menu Item Sketch → Include Library → Add .zip Library
   f) Select the PeerIOSerialControl Library file saved in step 'b' above

3. Download the Firmware
   a) In a web-browser; go to https://github.com/tgit23/AgIrrigationRemoteControl
   b) On Green Clone or download Choose Download Zip
   c) Save to a place you'll remember – like Documents or Desktop
   d) Unzip the Folder
   e) In Arduino Sketch IDE choose File → Open and Select
      - File HandRemote.ino for the HandRemote Firmware
        - Location Example: \Downloads\AgIrrigationRemoteControl-master\AgIrrigationRemoteControl-master\HandRemote
      - File PumpController.ino for the PumpController Firmware
        - Location Example: \Downloads\AgIrrigationRemoteControl-master\AgIrrigationRemoteControl-master\PumpController\PumpRemote

4. Compile and Upload the Firmware
   a) Plug in a USB cable from the Computer to the unit
   b) Select the Port the units USB is connected to; in Sketch menu Tools → Port
      - To determine Port; Open Windows Device Manager → Ports (COM & LPT) a new COM?? port appears right after plugging in the cable
   c) Select the Board; Sketch menu Tools → Board → Arduino/Genuino UNO
   d) In Arduino Sketch IDE; Press the Right-Arrow next to the Check mark in the Top-Left Corner to upload the firmware onto the unit

4.3 Customizing Firmware

4.3.1 Pump-Controller

✔ The Pump-Controller Firmware preforms the following duties
- Wait for Hand-Remote Requests and generate responses
- Calculate values (Virtual-Pin Values) for special devices that require a process to obtain a value (like ping/echo for Ultrasonic level measurements)
- Retain a Power-Off status after a power failure has occurred at the pump (Arduino is designed to keep digital pins low after a power failure)

NOTE: Wiring, Timing, and Almost all system functions are implemented on the Hand-Remote Firmware

A. Virtual Pins

Virtual pins are non-hardware related value identifiers that are created and assigned by the Pump-Controller firmware and then accessed by the Hand-Remote as a "Pin" value. An example case is the ultrasonic water level; The pump-controller firmware pings and times an echo of the ultrasonic distance meter using its D4 (Trigger) and D5 (Echo) pins and stores the measured value on "virtual pin 64". When the Hand-Remote queries for a value on pin-64 it actually gets the measured value which is a combination of D4 & D5 already calculated by the Pump-Controller.

✔ An Example of a Virtual Pin setup on the Pump-Controller firmware

```
void loop(){

  XBee.Available();


#if US_PRESENT>0
  // Read UltraSonic water level
  int ulCurrentTime = millis();
  if ( ulCurrentTime > ulLastPing + 1000 ) {
    XBee.VirtualPin(64, sonar.ping_in() );
    ulLastPing = ulCurrentTime;
  }
#endif

}
```

4.3.2 Hand-Remote

✔ The Hand-Remote Firmware preforms the following duties

- Storing the values in non-volatile memory (EEPROM) so user values aren't lost every-time the power is turned off
- Updating the current status (MAIN Values) of the Menu-Items and checking for Alarm boundaries.

- Processes User Input
  - UP/DOWN user buttons will increments or decrements the Menu-Items
  - RIGHT/LEFT user buttons will increments or decrements the Sub-Values of each Menu-Item
  - SELECT (square) user button will preform a value **SET**; or update MAIN values when they are being displayed

- Menu-Items can be associated/connected to (**Location**)
  - Pump-Controllers Control (OUTPUT) Pins - Example; Power control, Valve control or any other "control" device
  - Pump-Controllers Status (INPUT) Pins - Example; Measuring water levels, water pressure or any other "status" device
  - Hand-Remote Functions - Example; Setting which Pump-Controller the firmware will control and monitor
  - Hand-Remote INPUT/OUTPUT Pins - Example; An input pin that read the batteries voltage

- Menu-Items have **Sub[?]** Values, such as
  - The actual/read/**MAIN** value of the Menu-Items - Example; Power is currently either ON/OFF (current status)
  - **SET**ting a Value for the Menu-Item - Example; SET the Power to either ON/OFF
  - Setting a Too **LO**w **ALARM** Value - Example; When pressure is below 10 sound the Low Alarm
  - Setting a Too **HI**gh **ALARM** Value - Example; When pressure is above 100; sound the High Alarm

- Menu-Items can have Option Values (An Optional way of handling Menu-Item value assignments)
  - The number of options is limited by the "#define MAXOPTIONS" number in the firmware (line #85)
  - Options associates a common MAIN-Value to a Text identifier; for example ON/OFF or PUMP-FIELD, PUMP-GARDEN and etc...
  - A Menu-Items value must be a select-able set of Options or a Number but cannot be both.

- Menu-Item Limits
  - There can be a maximum of 30 Menu-Items defined. This is limited by the amount of EEPROM memory of the Arduino UNO
  - Each Hand-Remote can control up to a maximum of 4 Pump-Controllers

A. Creating a Menu-Item

1. Find the **SetupMenu()** function in the Hand-Remote firmware file; This is where all the Menu-Items are defined

2. Index - Determine where in the list of Menu-Items you'd like the new item to appear and use that index number
   a) For Example; Menu[5] would appear after Menu[4]
   b) If Menu[5] already exists; Change Menu[5] to Menu[6] and 6 to 7 and so forth until the end of the list is achieved.
   c) Menu[MONITOR] which is (0) and MENU[PUMPIDX] which is (1) are used in other places of the firmware and should ***NEVER BE CHANGED!***

3. Text - Assign the Menu-Item a defining Text
   a) The text is what will be displayed on the Hand-Remote display
      • Example; `Menu[5].Text="New-Item";`

4. Options - Determine IF the Menu-Item will have textual "Options" or is just a numerical value
   a) If the Menu-Item will report numerical values such as Voltage, Water Level, Pressure etc... Skip to step #5
   b) Determine the Menu-Items Options and assign each a;
      • Text Example; `Menu[5].Option[0].Text = "On";`
      • Value Example; `Menu[5].Option[0].Value = HIGH;`

5. Location - Determine the Location of the new Menu-Items value and where it will be gotten and set

   - The available Hardware Locations are;
     - PUMP0_PIN - The value is set/got from Menu[PUMPIDX].Option[0] defined Pump-Controller on an I/O Pin
     - PUMP1_PIN - The value is set/got from Menu[PUMPIDX].Option[1] defined Pump-Controller on an I/O Pin
     - PUMP2_PIN - The value is set/got from Menu[PUMPIDX].Option[2] defined Pump-Controller on an I/O Pin
     - PUMP3_PIN - The value is set/got from Menu[PUMPIDX].Option[3] defined Pump-Controller on an I/O Pin
     - ALL_PUMPS_PIN - The value is set/got from ALL defined Menu[PUMPIDX].Option's on their I/O Pin
     - HAND_PIN - The value is set/got from the Hand-Remote's I/O Pin
     - HAND_EPROM - The value is NOT any I/O Pin but instead is a value stored in the Hand-Remotes EEPROM memory
     - HAND_PROG - The value is NOT an I/O Pin but is instead used as a volatile memory spot (Seldom used)

   NOTE: *Hardware PUMPx_PIN locations can be compounded (+) to be pump specific; see the Analog Pin example below*

   - Add the Pin Type and Number to the '_PIN' Hardware Locations;

- ▫ A - An Arduino Analog Pin; Example; `Menu[5].Location = PUMP0_PIN + PUMP2_PIN + A+5;`
- ▫ D - An Arduino Digital Pin; Example; `Menu[5].Location = ALL_PUMPS_PIN + D+3;`

6. Sub[SET] – Determine IF the user should be allowed to set a value on the Location (Output Pins like turning Power ON/OFF)

   a) If the Menu-Item is for monitoring status and the user will not be setting its value (INPUT)... Skip to step #7
   b) If the Menu-Item needs to allow the user the ability to set the value (OUTPUT)
   - Set the "State" to SETTABLE to allow the user to SET the value
   - Example; `Menu[5].Sub[SET].State = SETTABLE;`

7. Sub[??ALARM] – Determine if the value should be monitored with an alarm
   a) LOALARM
   - If the Menu-items value is numeric and the value should be checked for getting too small
   - OR If the Menu-item value is an "option" and the value should be checked to see if it is EQUAL
     - Assign an LOALARM identifier Example; `Menu[5].Sub[LOALARM].ID = 'c';`

   b) HIALARM
   - If the Menu-items value is numeric and the value should be checked for getting too large
   - OR If the Menu-Item value is an "option" and the value should be checked to see if it is NOT-EQUAL
     - Assign a HIALARM identifier Example; `Menu[5].Sub[HIALARM].ID = 'C';`

8. Proceed to #4.3.2.3.Updating Changes|outline

B. Deleting a Menu-Item

1. Find the menu items index (ie. Menu[index]) you'd like to delete by identifying it by its' Menu[Index].Text setting
2. Select ALL Menu[index] lines with the same index
3. Press delete
4. Starting from the top make sure the indexes are in order; for example if you deleted [3], change [4] to [3], [5] to [4] and etc.. for all entries
5. Proceed to #4.3.2.3.Updating Changes|outline

C. Updating Changes

✔ After making the necessary Menu-Item Changes

- ○ Check Menu Indexes - Verify that all Menu-Items are indexed in order (all Menu indexes must be sequential)

- ○ Check NUM_MENU_ITEMS
  - ▪ Find `#define NUM_MENU_ITEMS` number in the firmware (around line #84)
  - ▪ Verify that the number assigned to it is one larger than the largest Menu index
    - ▫ For Example; If the last Menu-Item is `Menu[15]` then make sure `#define NUM_MENU_ITEMS` is `16` or greater

- ○ Check MAXOPTIONS - If Menu options were used
  - ▪ Find "`#define MAXOPTIONS`" number in the firmware (around line #85)
  - ▪ Verify that the number assigned to it is one larger than the largest 'index' in Option[index] that has been identified.
    - ▫ For Example; If `Menu[5].Option[4]` is used then make sure `#define MAXOPTIONS` is `5` or greater

D. Menu-Item Setup (Example)

The menu items are defined in the **SetupMenu()** *Function identified by line 'void SetupMenu() {'. Each item in the Menu has a numeric index (i.e. Menu[index-#] below the constant 'MONITOR' is assigned index-0 and 'PUMPIDX' is assigned index-1).*

```
/*****************************************************************************************************************//**
 * @brief  Setup the LCD menu
 * @remarks
 * - Allows a single spot customization to users configuration
 * - Be sure to change 'NUM_MENU_ITEMS' when adding/deleting items
 * - Display will show the items in the same order as they are defined here
 * @code
 *   exmaple code
 * @endcode
*******************************************************************************************************************/
void SetupMenu() {

  //vvvvvv[ HAND-REMOTE MENU ITEMS ]vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
  Menu[MONITOR].Text = "Monitor";                    // Create a menu item for monitoring ALL Pump-Controllers
  Menu[MONITOR].Location = HAND_EPROM;               // Value is stored in the Hand-Remotes EEPROM memory
  Menu[MONITOR].Sub[SET].State = SETTABLE;           // User can SET the value

  Menu[MONITOR].Option[0].Text = "Sel";              // Monitor "Sel" Pump at a time   - Option #0 = Selected
  Menu[MONITOR].Option[0].Value = 0;                 // "Sel" will be the value 0      - One = 0
  Menu[MONITOR].Option[1].Text = "All";              // Monitor "All" Pumps            - Option #1 = All
```

```
  Menu[MONITOR].Option[1].Value = 1;                  // "All" will be the value 1      - All = 1
  Menu[MONITOR].LastOptionIdx = 1;                    // Last Option Index defined      - Number of Options - 1

  //-------------------------------------------------------------------------------------------------------------
  Menu[PUMPIDX].Text = "Pump";                        // Menu Item used to select the Pump-Controller
  Menu[PUMPIDX].Location = HAND_EPROM;                // Selected pump is stored in EEPROM
  Menu[PUMPIDX].Sub[SET].State = SETTABLE;            // Allow this value to be 'SET' by the user

  // Define the Pump-Controllers ( A maximum of 4; index 0->3 )
  // Check that "#define MAXOPTIONS" allows the number of indexes used ( default; 2 )
  Menu[PUMPIDX].Option[0].Text = "Canal";             // Pump can be "Canal"            - Option #0 = Canal
  Menu[PUMPIDX].Option[0].Value = 10;                 // "Canal" Value=TRANSCEIVER_ID   - Canal = TransceiverID #10
  Menu[PUMPIDX].Option[1].Text = "Ditch";             // Pump can be "Ditch"            - Option #1 = Ditch
  Menu[PUMPIDX].Option[1].Value = 11;                 // "Ditch" Value=TRANSCEIVER_ID   - Ditch = TrasceiverID #11
  Menu[PUMPIDX].LastOptionIdx = 1;                    // Last Pump Option Index         - # of Pump-Controllers - 1

  //-----------------------------------------
  Menu[2].Text = "Battery(B)";                  // Create a menu item for monitoring the Battery
  Menu[2].Location = HAND_PIN+ A+1;             // Battery level is gotten from the Hand-Remote pin A1
  Menu[2].Sub[LOALARM].ID = 'b';                // A Low Alarm is identified by a lower-case 'b'

  //vvvvvv[ PUMP-SPECIFIC MENU ITEMS ]vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
  //-----------------------------------------
  Menu[3].Text = "Power(P)";                    // Create a menu item for Power Control
  Menu[3].Location = ALL_PUMPS_PIN+ D+7;        // Power is set/got on all Pump-Controller's on pin [D7]

  Menu[3].Sub[SET].State = SETTABLE;            // Allow this Value to be 'SET' by the user
  Menu[3].Sub[LOALARM].ID = 'p';                // A Low Alarm is identified by a lower-case 'p'
  Menu[3].Sub[HIALARM].ID = 'P';                // A High Alarm is identified by an upper-case 'P'

  Menu[3].Option[0].Text = "Off";               // Power can be "Off"             - Option #0 = Off
  Menu[3].Option[0].Value = LOW;                // "Off" will be the value 'LOW'  - Off = LOW
  Menu[3].Option[1].Text = "On";                // Power can be "On"              - Option #1 = On
  Menu[3].Option[1].Value = HIGH;               // "On" will be the value 'HIGH'  - On = HIGH
  Menu[3].LastOptionIdx = 1;                    // Last Option Index defined      - Number of Options - 1

  //-----------------------------------------
  Menu[4].Text = "Water (L)";                   // Create a menu item for Water Level Transducer
  Menu[4].Location = PUMP1_PIN+ A+64;           // Water Level is read from VIRTUAL (Pump-Controllers firmware) pin 64
  Menu[4].Sub[LOALARM].ID = 'l';                // A Low Alarm is identified by a lower-case 'l'
  Menu[4].Sub[HIALARM].ID = 'L';                // A High Alarm is identified by an upper-cse 'L'

  //-----------------------------------------
  Menu[5].Text = "Pressure(R)";                 // Create a menu item for the Primary Pressure Transducer
  Menu[5].Location = ALL_PUMPS_PIN+ A+3;        // The 'signal' is gotten on all Pump-Controllers on pin [A3]
  Menu[5].Sub[LOALARM].ID='r';                  // A Low Pressure alarm is identified by a lower-case 'r'
  Menu[5].Sub[HIALARM].ID='R';                  // A High Pressure alarm is identified by an upper-case 'R'

  //-----------------------------------------
  Menu[6].Text = "Pressure(S)";                 // Create a menu item for the Secondary Pressure Transducer
  Menu[6].Location = PUMP0_PIN+ A+4;            // Menu-item is for Pump-Option #0 (Canal) on Pin (A4)
  Menu[6].Sub[LOALARM].ID='s';                  // A Low Pressure alarm is identified by a lower-case 's'
  Menu[6].Sub[HIALARM].ID='S';                  // A High Pressure alarm is identified by an upper-case 'S'

  //------------[ Start-Up the Display ( DO NOT CHANGE! )]-------------
  GetItem(PUMPIDX);                             // Read the last selected Pump-Controller from EEPROM
  idx = STARTIDX;SubIdx=MAIN;                   // Set where the Menu will start
  GetItem();                                    // Get starting Menu item
  LCD_display();                                // Update the display
}
```

5. OTHER RESOURCES / LINKS

- ✔ Support
  - You can email me at tgit28@gmail.com with any questions, comments and or requests.
- ✔ RF Controllers
  - Commercial
    - http://www.forbixindia.com/electronics/remote-motor-control/
    - http://www.remotecontroleverything.com/long-range-wireless-remote-control-3-phase-power-water-pump/
    - http://www.sprinklerwarehouse.com/remote-control-guide-s/6282.htm
    - http://www.sprinklerwarehouse.com/Hunter-Remote-Controllers-Timers-s/109.htm
    - http://rayshobby.net/opensprinkler/
    - Rain Bird Controller wiring diagram http://www.lawnh2o.com/rainbird/PDF/Wiring-Manual.pdf
  - Home-made Projects
    - https://www.instructables.com/id/COMPACT-AND-ROBUST-AUTOMATED-AGRICULTURE-REMOTE/
- ✔ Cellular Controllers

- Commercial
  - http://www.myfieldnet.com/
  - https://www.ag-rite.com/system-description
  - 380V 50Hz 500W (TOO SMALL) https://www.alibaba.com/product-detail/GSM-industrial-three-phase-power-switch_60270579733.html ($100-$200)
  - Looks cheap http://www.bieneelectronics.com/products/br_application.htm

- Home-made Projects
  - https://www.instructables.com/id/SMS-controlled-Wireless-Irrigation-System/
  - https://www.stavros.io/posts/arduino-powered-irrigation-system/

✔ Component considerations
- Atmel328 with RF boards https://www.digitalsmarties.net/products/jeenode
- ALL about Solid State Relays http://www.phidgets.com/docs/Solid_State_Relay_Primer
- Remote Relay https://www.controlanything.com/Relay/Relay/XSC_PROXR
- RF Remote http://www.remotecontroleverything.com/long-range-wireless-remote-control-3-phase-power-water-pump/ ($83 – A serious alternative @ 2000-meter?? range)

✔ Digi Xbee Links
- Sparkfun Xbee User Guide https://learn.adafruit.com/xbee-radios/
- Sparkfun Xbee Buying Guide https://www.sparkfun.com/pages/xbee_guide
- Xbee 1-WATT (up to 40-miles) https://www.digi.com/products/xbee-rf-solutions/embedded-rf-modules-modems/xtend-module#overview (S3B is .25 Watts)
- Xbee Homepage https://www.digi.com/products/xbee-rf-solutions
-