Auto Dialing Sump Pump Alarm System

ECET 491 / Senior Project Phase II

Project Coordinator: Paul Lin

Presented by: Fahad Nader

December 13, 2013

A Picture is Worth a Thousand Words!

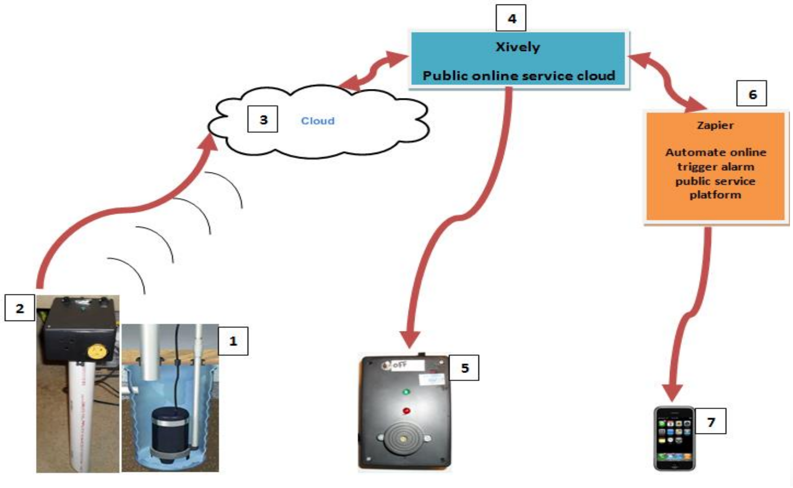
Problem Statement

- ❖ Flooding damages are common in the tri-state area
- ❖ Issues with the current mechanical sump pump floating ball switches
- ❖ Absence of auto dial alarm feature

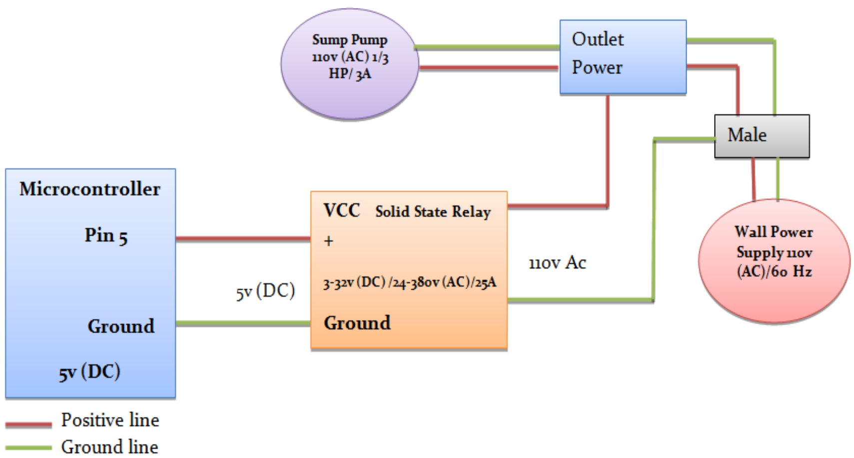
Solution

Auto Dialing Sump Pump Alarm System

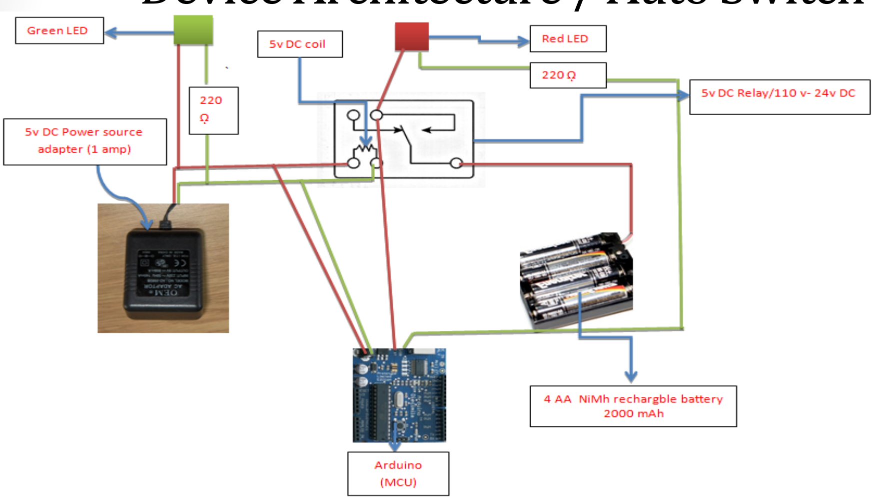
Device Architecture



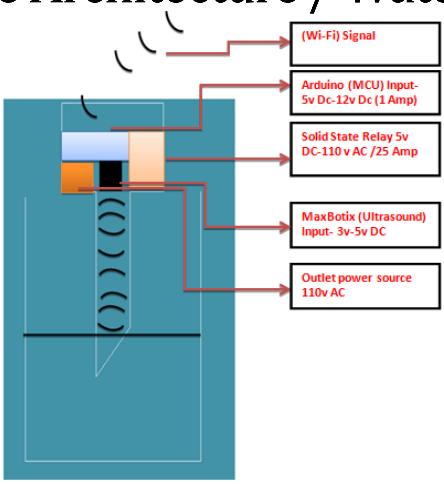
Device Architecture / Power Source



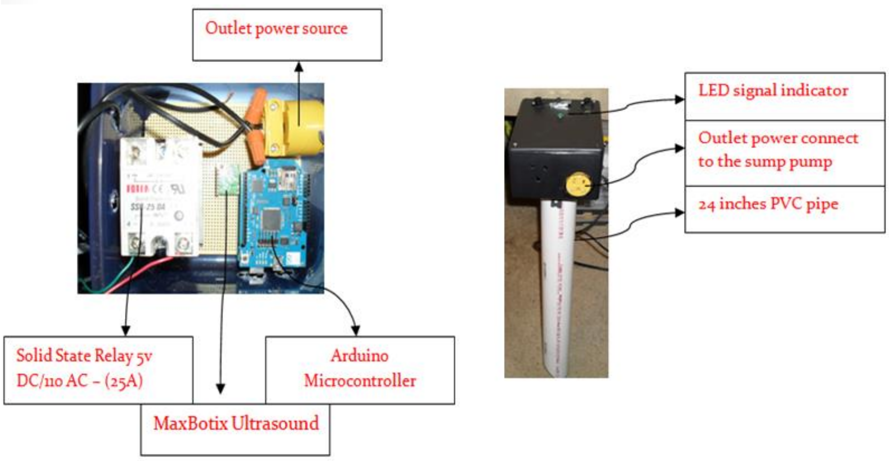
Device Architecture / Auto Switch



Device Architecture / Water Sensor

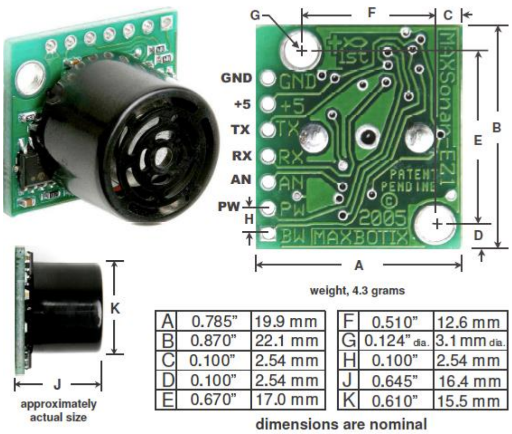


Device Architecture / Water Sensor



Device Architecture / Arduino Wi-Fi Shield

Device Architecture / Ultrasound Sensor



MaxSonar®-EZ1™
Data Sheet

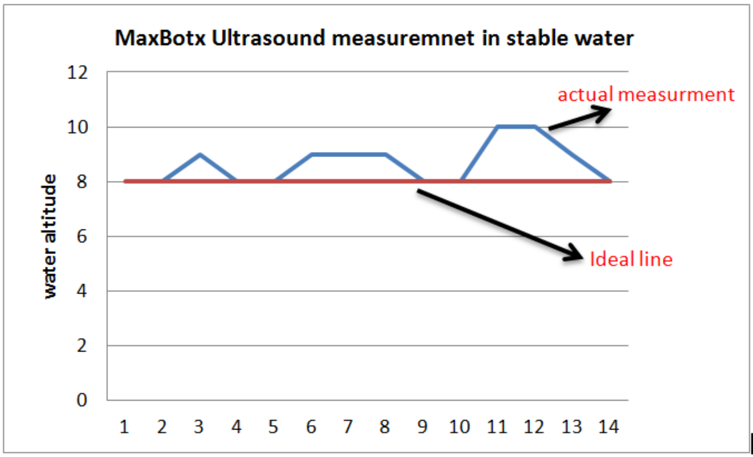
A	0.785"	19.9 mm	F	0.510"	12.6 mm
B	0.870"	22.1 mm	G	0.124" dia.	3.1 mm dia.
C	0.100"	2.54 mm	H	0.100"	2.54 mm
D	0.100"	2.54 mm	J	0.645"	16.4 mm
E	0.670"	17.0 mm	K	0.610"	15.5 mm

dimensions are nominal

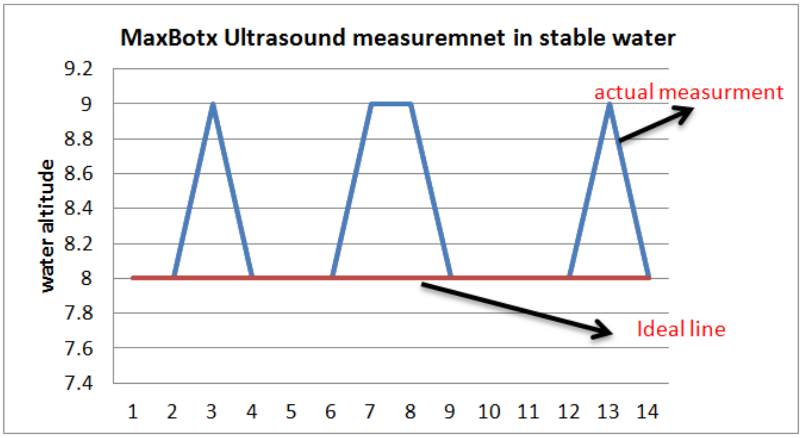
Device Architecture / Ultrasound Testing

Distance (inches)	MaxBotix Ultrasound (inches)	Accurate measurement (percentage) %
0	0	100
6	6	100
7	6.8	97
8	8	100
9	9	100
11	10.7	97
12	12	100

Device Architecture / Ultrasound Testing

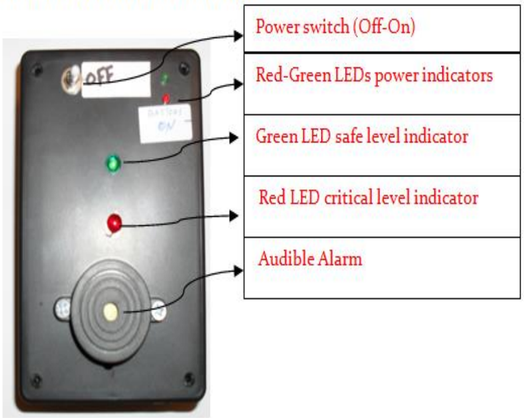


Device Architecture / Ultrasound Testing



Device Architecture / Water Level Indicator

Water level indicator module



Device Architecture / Battery Power Calculations

Arduino Battrey power source Ah calculations:

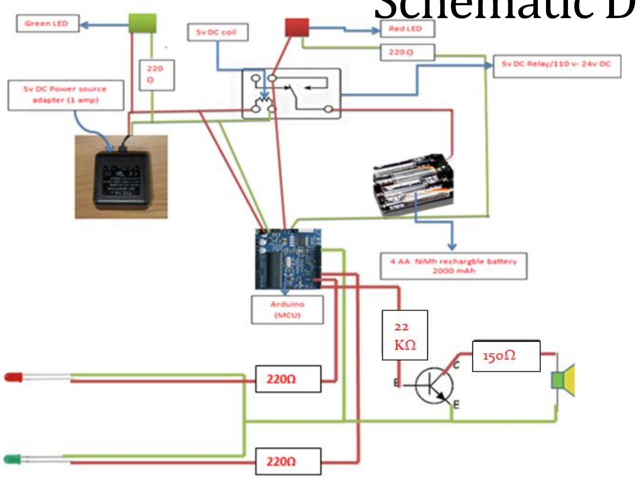
4 AA batteries have 2000 mAh, the arduino (MCU) consumes 60 mA

Battery Ah calculation is 2000/60mA = 33 hours ,

since battey acual usege is 60% only, so 33 *.60 = 20 hours

The battey power will last 20 hours.

Device Architecture / Water Indicator Schematic Diagram



Device Architecture / Ultrasound Testing

Water level indicator Resistors values calculations

The LEDs have 20 mA forward current and 2v drop cross it.

The source voltage is 5v DC, v= (5-2) = 3 v

R= v/I = 3/.02A = 150Ω

Power dissipation= I*V = 3v * .02A = 60 mW, under the resistor rating of 125 mW.

So for the green and red LEDs I used 220Ω instead 150Ω because it's easy to find it and still allows the LED to glow sufficiently bright. I used resistors 220Ω, ¼ w.

Device Architecture / Wi-Fi Signal Testing

The formula of the conversion from dBm to Watt is

Power (watt) = 1 watt * 10^ (dBm/10) = watt

-67 dBm = 1 watt * 10^ (-67/10) = 1.995*10^-10 watt

-75 dBm = 1 watt*10^ (-75/10) = 3.162*10^-11 watt

Table (2) the testing the Wi-Fi strength signal vs. distance

Distance	Wi-Fi Signal Strength (dBm)	Wi-Fi Signal Strength (Watt)
3	-67	19.95 nW
5	-67	19.95 nW
10	-69	12.50 nW
15	-72	31.62 nW
20	-75	63.09 nW

Device Architecture / Software Code

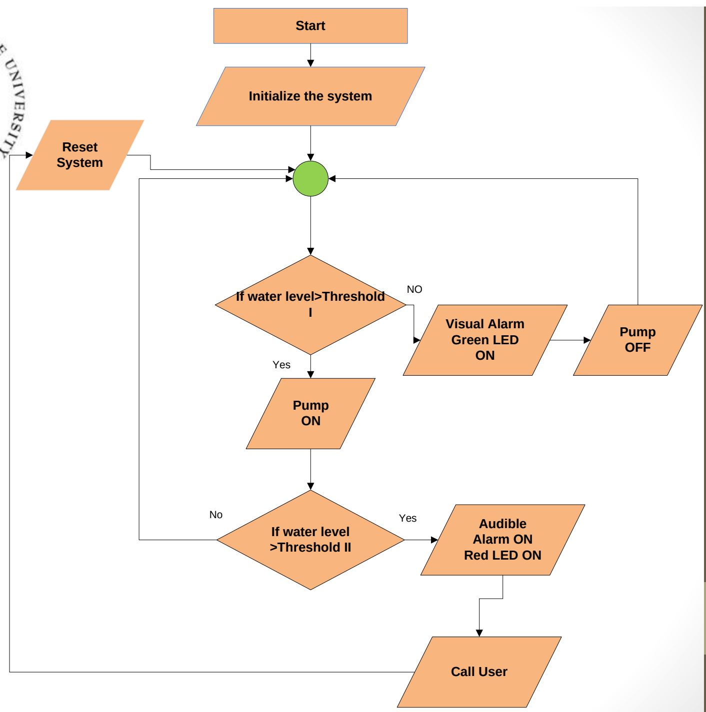
```
#include <SPI.h>
#include <WiFi.h>
#include <HttpClient.h>
#include <Xively.h>
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
char ssid[] = "HOME-1B37"; //  your network SSID (name)
char pass[] = "014E34CCE9A71F2E";    // your network password (use for WPA, or use as key for WEP)
int keyIndex = 0;            // your network key Index number (needed only for WEP)
int sensorPin = A0;
int safeZone =18;      // set threshold II
int safeZone1 =14; // set threshold I
int greenLed = 2,  buzzer = 3 , redLed = 4;
int relayPin = 5;
long value = 0;
 long duration;
int sensorValue=0;
int status = WL_IDLE_STATUS;
// Your Xively key to let you upload data
char xivelyKey[] = "83WAiTIagjtFImZVvMBZHoc23J660YslwLvWivtgBltoXK0s";
// Analog pin which we're monitoring (0 and 1 are used by the Ethernet shield)
// Define the strings for our datastream IDs
char sensorId[] = "sensor_reading";
XivelyDatastream datastreams[] = {
  XivelyDatastream(sensorId, strlen(sensorId), DATASTREAM_FLOAT),
};
// Finally, wrap the datastreams into a feed
XivelyFeed feed(366770728 , datastreams, 1 /* number of datastreams */);
WiFiClient client;
```

Device Architecture / Void Loop Function

This is the code I Added to the Xively orignal codes:

```
void loop() {
int sensorValue = analogRead(sensorPin)/1.5;   // this is the function of the water level measurement
 datastreams[0].setFloat(sensorValue);
 Serial.print("Read sensor value");
 Serial.println(datastreams[0].getFloat());
 Serial.println("Uploading it to Xively");
 int ret = xivelyclient.put(feed, xivelyKey);
```

Device Architecture / Flowchart



Device Architecture / Validation Testing

Table (3) – testing the complete system for 5 trials for the alarm response time

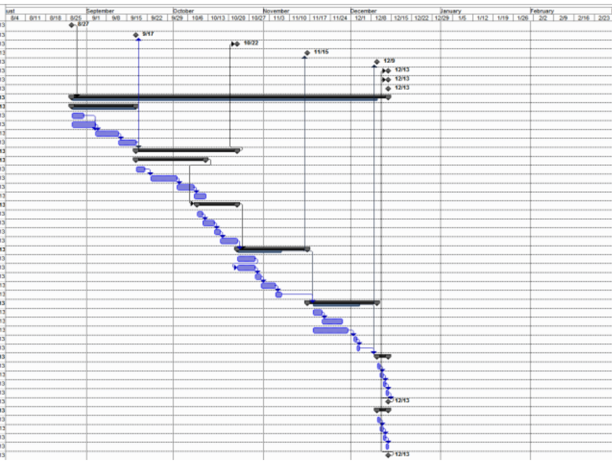
Number of trials	Alarm Response time (seconds)
1	33
2	37
3	45
4	38
5	42

Project Management / Cost

Tasks: Days	Days	Cost (Parts)	Dollar$
Research	5	Sump pump	80
Picking parts value	3	Arduino (MCU)	97
Order parts	9	Battery back up	10
Design the Control system	7	Water level sensor	15
Design water level sensor	5	relays , resistors, and wires	35
Build the system on bread board	7	Circuit Board	5
Test the system	3	Enclosure	10
Update the system	4	USB Port	10
Build the prototype system	5	Visual and audio local alarm	15
Integrate the system with auto dial	8	Wi-Fi shield/Ethernet Shield	65
Troubleshoot and retest the system	4	Totals	362$
Software Coding	5		
System verification complete	3		

Project Management / Time

Task Name	Duration	Start	Finish
Phase II Start	0 days	Tue 8/27/13	Tue 8/27/13
System Design Complete	0 days	Tue 9/17/13	Tue 9/17/13
Detailed Design Complete	0 days	Tue 10/22/13	Tue 10/22/13
Functional Prototype Complete	0 days	Fri 11/15/13	Fri 11/15/13
Verification Testing Complete	0 days	Mon 12/9/13	Mon 12/9/13
Final Report Submitted	0 days	Fri 12/13/13	Fri 12/13/13
Presentation Delivered	0 days	Fri 12/13/13	Fri 12/13/13
End of Semester	0 days	Fri 12/13/13	Fri 12/13/13
Prototype Development	**79 days**	**Tue 8/27/13**	**Fri 12/13/13**
System Design Phase	**16 days**	**Tue 8/27/13**	**Tue 9/17/13**
Design water level sensor	4 days	Tue 8/27/13	Fri 8/30/13
Design control system	6 days	Tue 8/27/13	Tue 9/3/13
Test the design and update on multisim	6 days	Wed 9/4/13	Wed 9/11/13
Final System Design Complete	4 days	Thu 9/12/13	Tue 9/17/13
Detailed Design Phase	**25 days**	**Wed 9/18/13**	**Tue 10/22/13**
Hardware Design	**18 days**	**Wed 9/18/13**	**Fri 10/11/13**
Picking the parts value	3 days	Wed 9/18/13	Fri 9/20/13
Ordering the parts	7 days	Mon 9/23/13	Tue 10/1/13
Building the system on bread board	4 days	Wed 10/2/13	Mon 10/7/13
Test and update the system	4 days	Tue 10/8/13	Fri 10/11/13
Task 4	**10 days**	**Wed 10/9/13**	**Tue 10/22/13**
Build the control system	2 days	Wed 10/9/13	Thu 10/10/13
Test the control system	2 days	Fri 10/11/13	Mon 10/14/13
Build water level sensor	2 days	Tue 10/15/13	Wed 10/16/13
Troubleshoot and update the system	4 days	Thu 10/17/13	Tue 10/22/13
System Integration Phase	**18 days**	**Wed 10/23/13**	**Fri 11/15/13**
Assemble complete system	4 days	Wed 10/23/13	Mon 10/28/13
Test complete system	4 days	Wed 10/23/13	Mon 10/28/13
Troubleshoot	2 days	Tue 10/29/13	Wed 10/30/13
Retest compelete system	3 days	Thu 10/31/13	Mon 11/4/13
Design update	2 days	Tue 11/5/13	Wed 11/6/13
System Verification Phase	**17 days**	**Sat 11/16/13**	**Mon 12/9/13**
Verify water sensor and control system	3 days	Mon 11/18/13	Wed 11/20/13
Verufy power supply back up system	5 days	Thu 11/21/13	Wed 11/27/13
Verify response time remote alarm	10 days	Mon 11/18/13	Fri 11/29/13
Verfy system testing	1 day	Mon 12/2/13	Mon 12/2/13
Verification Testing Complete	1 day	Tue 12/3/13	Tue 12/3/13
Final Report Development	**4 days**	**Tue 12/10/13**	**Fri 12/13/13**
Rough draft	1 day	Tue 12/10/13	Tue 12/10/13
Update first draft	1 day	Wed 12/11/13	Wed 12/11/13
Rewrite final report	1 day	Thu 12/12/13	Thu 12/12/13
Final report complete	1 day	Fri 12/13/13	Fri 12/13/13
Final Report Submitted	0 days	Fri 12/13/13	Fri 12/13/13
Presentation Development	**4 days**	**Tue 12/10/13**	**Fri 12/13/13**
write rough presentation	1 day	Tue 12/10/13	Tue 12/10/13
update presentation	1 day	Wed 12/11/13	Wed 12/11/13
rewrite the presentation	1 day	Thu 12/12/13	Thu 12/12/13
presentation complete	1 day	Fri 12/13/13	Fri 12/13/13
Presentation Delivered	0 days	Fri 12/13/13	Fri 12/13/13



Summary

- Requirement
- Risk

Demonstration / Sump Pump Control

Demonstration / Testing Alarm System

Demonstration / Back-Up Battery

Lessons Learned

The first lesson learned from this project was to get the water sensor ultrasound working properly with the software codes. I also learned how to embedded microcontroller in any projects in the future. I learned how to build electrical circuits and how to design circuits, which made me more confident to work as electrical engineer.

Questions?