Hack-That-Flood

SPONSORED BY RICK DAVIDS

Jordan Peralta (CE)
Rafi Simon (CE)
Habib Lawal (EE)
Andrew Osmanski (EE)

Andrew Osmanski

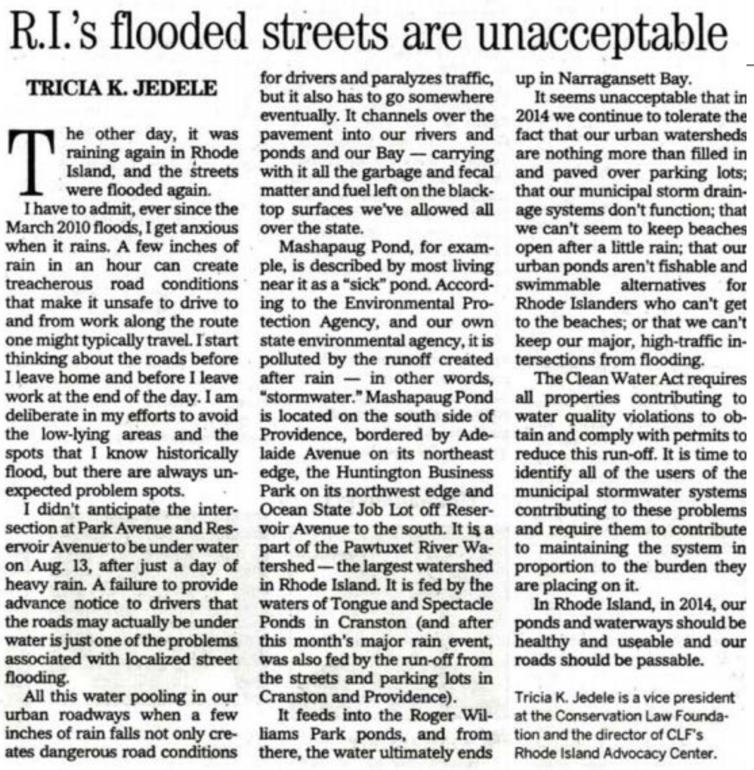
Acknowledgement

Sponsor and Project Director: Rick Davids
Capstone Instructor: Dr. Harish Sunak
Electrical Engineering consultant: Dr. Brice Loose
Computer Engineering consultant: Chris Damon

“Hack That Flood” Objective

Create and develop a smartphone application integrated with a remote, sensor platform that measures depth, acceleration, and GPS location. The smartphone application will determine, and alert if any integrated areas are subject to flooding.

Motivation



R.I.'s flooded streets are unacceptable

TRICIA K. JEDELE

The other day, it was raining again in Rhode Island, and the streets were flooded again.

I have to admit, ever since the March 2010 floods, I get anxious when it rains. A few inches of rain in an hour can create treacherous road conditions that make it unsafe to drive to and from work along the route one might typically travel. I start thinking about the roads before I leave home and before I leave work at the end of the day. I am deliberate in my efforts to avoid the low-lying areas and the spots that I know historically flood, but there are always unexpected problem spots.

I didn't anticipate the intersection at Park Avenue and Reservoir Avenue to be under water on Aug. 13, after just a day of heavy rain. A failure to provide advance notice to drivers that the roads may actually be under water is just one of the problems associated with localized street flooding.

All this water pooling in our urban roadways when a few inches of rain falls not only creates dangerous road conditions for drivers and paralyzes traffic, but it also has to go somewhere eventually. It channels over the pavement into our rivers and ponds and our Bay — carrying with it all the garbage and fecal matter and fuel left on the blacktop surfaces we've allowed all over the state.

Mashapaug Pond, for example, is described by most living near it as a "sick" pond. According to the Environmental Protection Agency, and our own state environmental agency, it is polluted by the runoff created after rain — in other words, "stormwater." Mashapaug Pond is located on the south side of Providence, bordered by Adelaide Avenue on its northeast edge, the Huntington Business Park on its northwest edge and Ocean State Job Lot off Reservoir Avenue to the south. It is a part of the Pawtuxet River Watershed — the largest watershed in Rhode Island. It is fed by the waters of Tongue and Spectacle Ponds in Cranston (and after this month's major rain event, was also fed by the run-off from the streets and parking lots in Cranston and Providence).

It feeds into the Roger Williams Park ponds, and from there, the water ultimately ends up in Narragansett Bay.

It seems unacceptable that in 2014 we continue to tolerate the fact that our urban watersheds are nothing more than filled in and paved over parking lots; that our municipal storm drainage systems don't function; that we can't seem to keep beaches open after a little rain; that our urban ponds aren't fishable and swimmable alternatives for Rhode Islanders who can't get to the beaches; or that we can't keep our major, high-traffic intersections from flooding.

The Clean Water Act requires all properties contributing to water quality violations to obtain and comply with permits to reduce this run-off. It is time to identify all of the users of the municipal stormwater systems contributing to these problems and require them to contribute to maintaining the system in proportion to the burden they are placing on it.

In Rhode Island, in 2014, our ponds and waterways should be healthy and useable and our roads should be passable.

Tricia K. Jedele is a vice president at the Conservation Law Foundation and the director of CLF's Rhode Island Advocacy Center.

Computer Engineering Requirements

Map with flood prediction

Set up server to host the map

Phone application to display the end result

Jordan Peralta

Electrical Engineering Requirements

Flotation Sensor Platform

Microcontroller Device

- Acceleration
- Depth
- GPS location

GSM Communication

Andrew Osmanski

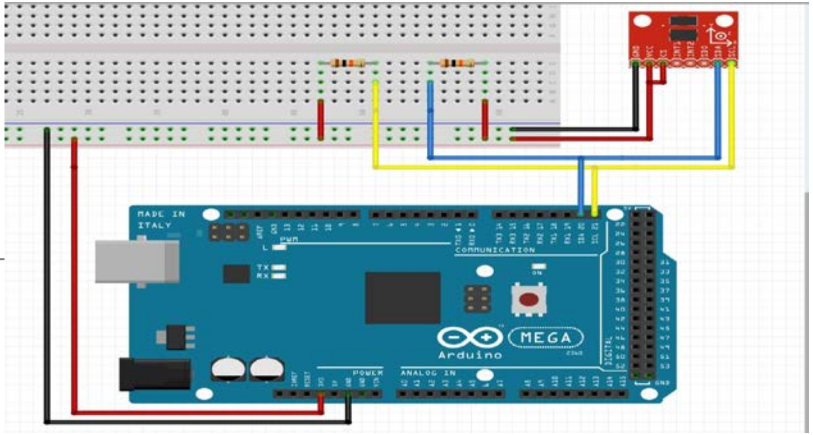
Arduino Mega 2560

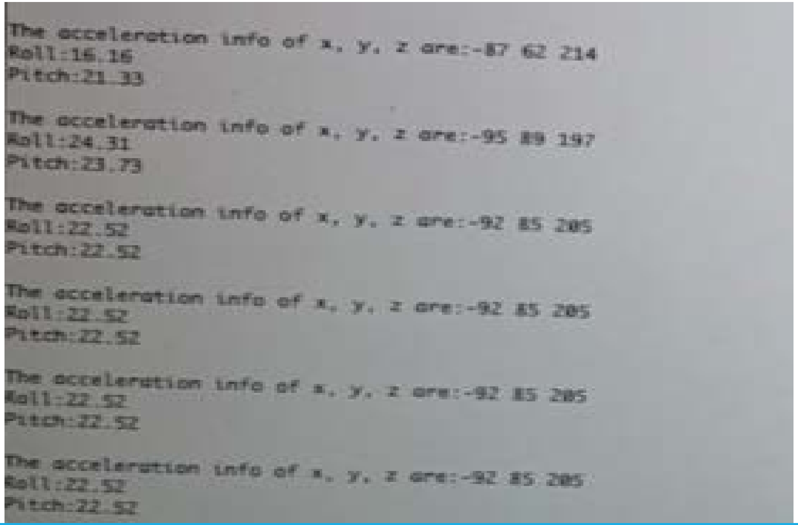
microcontroller board designed for complex projects

allow to power devices, collect outputs

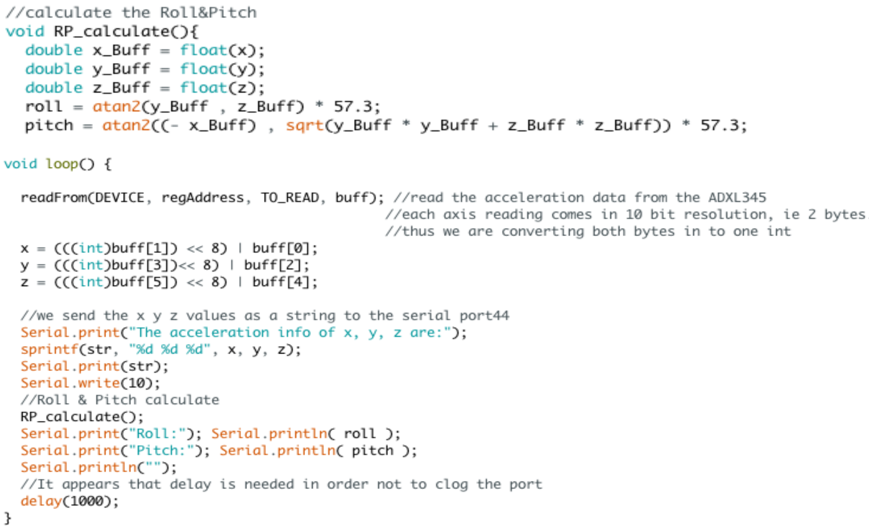
Sparkfun ADXL345 Accelerometer

X,Y, and Z axis
Roll, Pitch , and Yaw coordinates





```
The acceleration info of x, y, z are:-87 62 214
Roll:16.16
Pitch:21.33

The acceleration info of x, y, z are:-95 89 197
Roll:24.31
Pitch:23.73

The acceleration info of x, y, z are:-92 85 205
Roll:22.52
Pitch:22.52

The acceleration info of x, y, z are:-92 85 205
Roll:22.52
Pitch:22.52

The acceleration info of x, y, z are:-92 85 205
Roll:22.52
Pitch:22.52

The acceleration info of x, y, z are:-92 85 205
Roll:22.52
Pitch:22.52
```



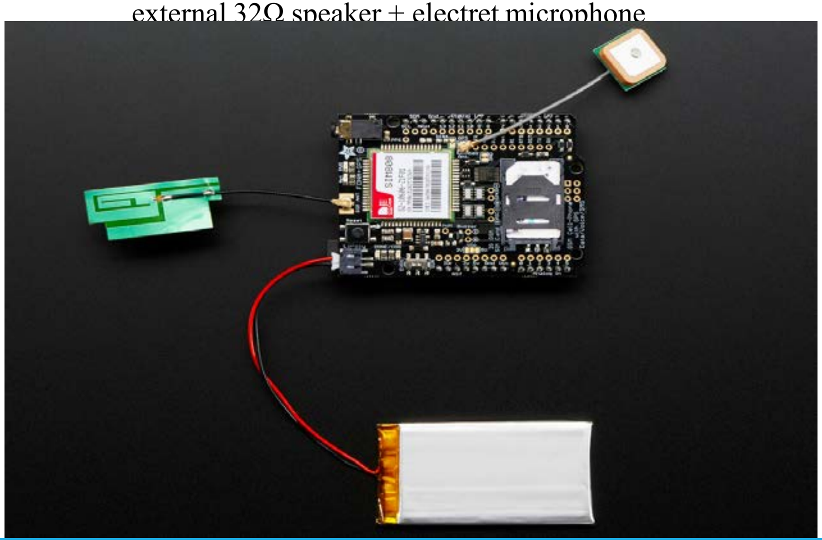
```
//calculate the Roll&Pitch
void RP_calculate(){
  double x_Buff = float(x);
  double y_Buff = float(y);
  double z_Buff = float(z);
  roll = atan2(y_Buff , z_Buff) * 57.3;
  pitch = atan2((- x_Buff) , sqrt(y_Buff * y_Buff + z_Buff * z_Buff)) * 57.3;

void loop() {

  readFrom(DEVICE, regAddress, TO_READ, buff); //read the acceleration data from the ADXL345
                                               //each axis reading comes in 10 bit resolution, ie 2 bytes.
                                               //thus we are converting both bytes in to one int
  x = (((int)buff[1]) << 8) | buff[0];
  y = (((int)buff[3])<< 8) | buff[2];
  z = (((int)buff[5]) << 8) | buff[4];

  //we send the x y z values as a string to the serial port44
  Serial.print("The acceleration info of x, y, z are:");
  sprintf(str, "%d %d %d", x, y, z);
  Serial.print(str);
  Serial.write(10);
  //Roll & Pitch calculate
  RP_calculate();
  Serial.print("Roll:"); Serial.println( roll );
  Serial.print("Pitch:"); Serial.println( pitch );
  Serial.println("");
  //It appears that delay is needed in order not to clog the port
  delay(1000);
}
```

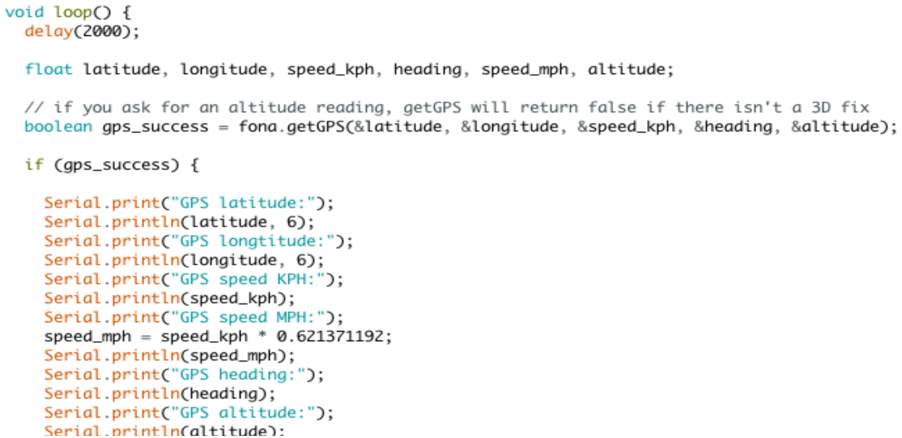
Adafruit FONA 808 Cellular + GPS shield

Quad-band 850/900/1800/1900MHz - connect onto any global GSM network with any 2G SIM;

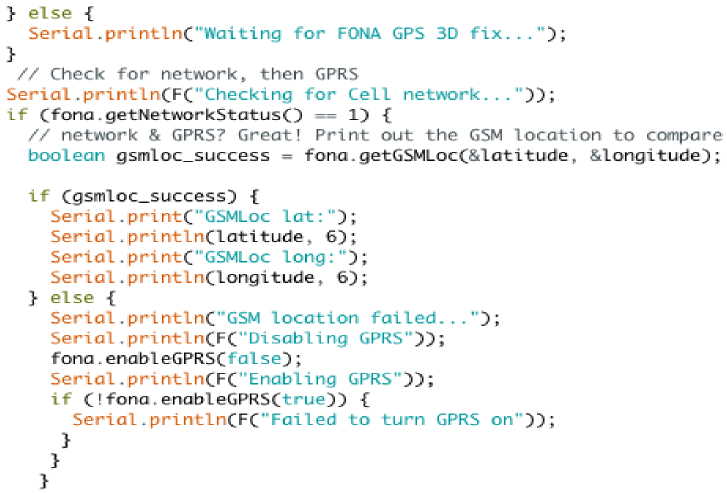
Fully-integrated GPS (MT3337 chipset with -165 dBm tracking sensitivity) that can be controlled and query over the same serial port;

Make and receive voice calls using a headset or an external 32Ω speaker + electret microphone





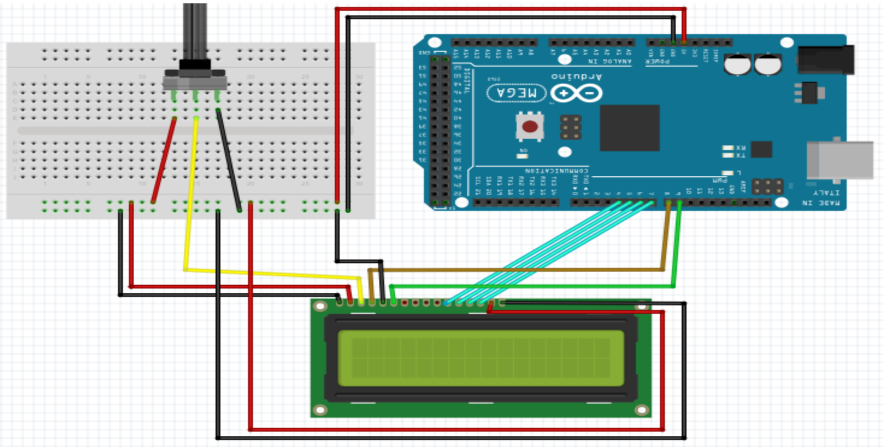
```
void loop() {
  delay(2000);

  float latitude, longitude, speed_kph, heading, speed_mph, altitude;

  // if you ask for an altitude reading, getGPS will return false if there isn't a 3D fix
  boolean gps_success = fona.getGPS(&latitude, &longitude, &speed_kph, &heading, &altitude);

  if (gps_success) {

    Serial.print("GPS latitude:");
    Serial.println(latitude, 6);
    Serial.print("GPS longtitude:");
    Serial.println(longitude, 6);
    Serial.print("GPS speed KPH:");
    Serial.println(speed_kph);
    Serial.print("GPS speed MPH:");
    speed_mph = speed_kph * 0.621371192;
    Serial.println(speed_mph);
    Serial.print("GPS heading:");
    Serial.println(heading);
    Serial.print("GPS altitude:");
    Serial.println(altitude);
```

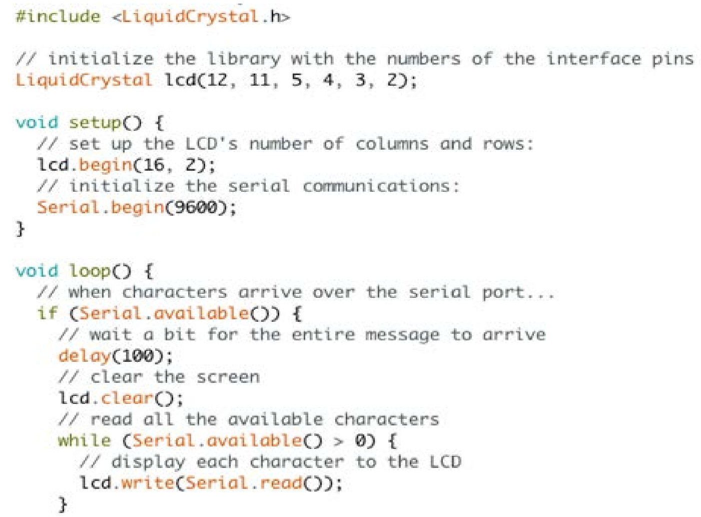


```
 } else {
   Serial.println("Waiting for FONA GPS 3D fix...");
 }
  // Check for network, then GPRS
 Serial.println(F("Checking for Cell network..."));
 if (fona.getNetworkStatus() == 1) {
   // network & GPRS? Great! Print out the GSM location to compare
   boolean gsmloc_success = fona.getGSMLoc(&latitude, &longitude);

   if (gsmloc_success) {
     Serial.print("GSMLoc lat:");
     Serial.println(latitude, 6);
     Serial.print("GSMLoc long:");
     Serial.println(longitude, 6);
   } else {
     Serial.println("GSM location failed...");
     Serial.println(F("Disabling GPRS"));
     fona.enableGPRS(false);
     Serial.println(F("Enabling GPRS"));
     if (!fona.enableGPRS(true)) {
       Serial.println(F("Failed to turn GPRS on"));
     }
   }
  }
```

16 x 2 LCD Screen

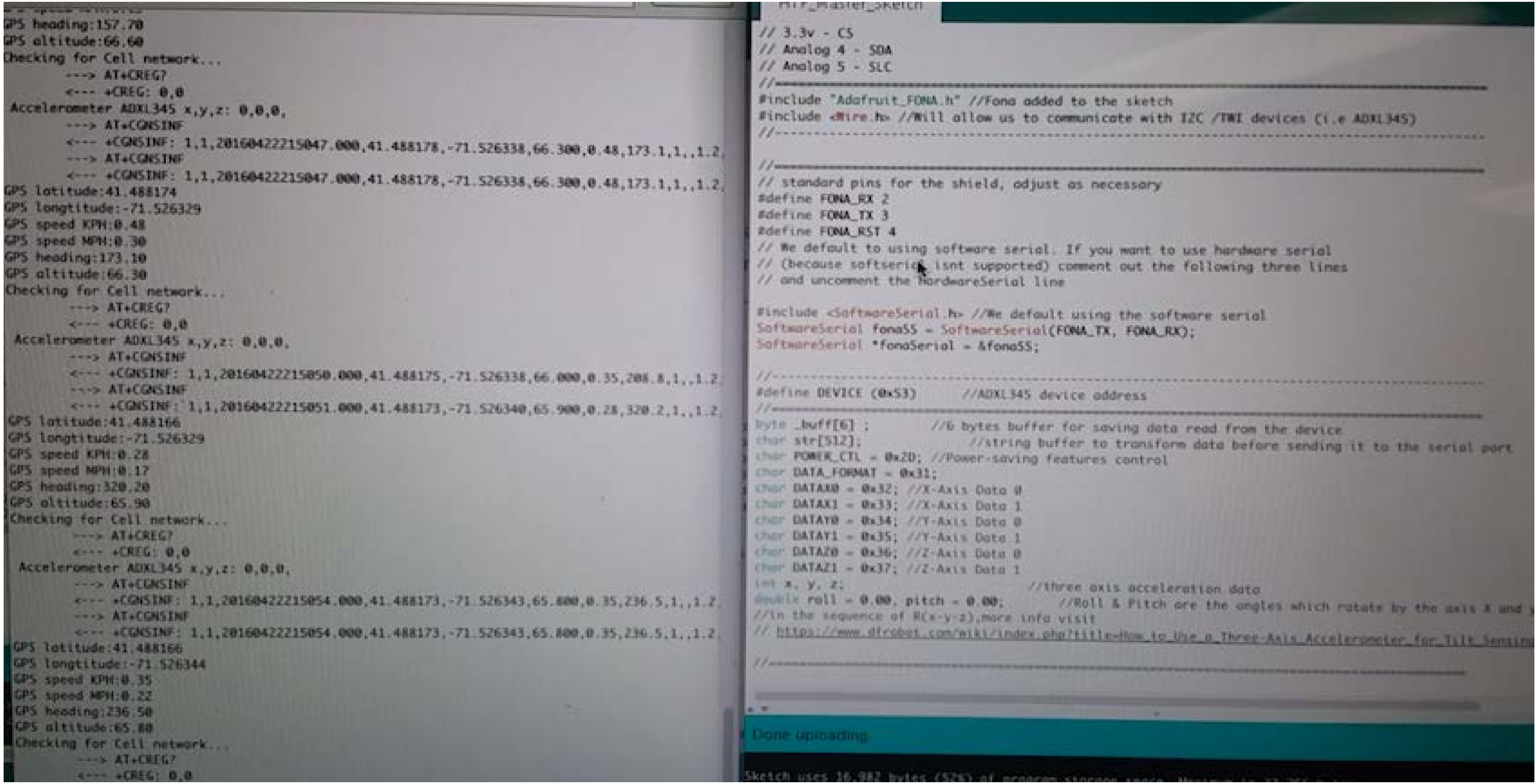
Display results
Auto Scroll





```
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // initialize the serial communications:
  Serial.begin(9600);
}

void loop() {
  // when characters arrive over the serial port...
  if (Serial.available()) {
    // wait a bit for the entire message to arrive
    delay(100);
    // clear the screen
    lcd.clear();
    // read all the available characters
    while (Serial.available() > 0) {
      // display each character to the LCD
      lcd.write(Serial.read());
    }
```

Results for GPS + Acceleration



```
GPS heading:157.70
GPS altitude:66.60
Checking for Cell network...
        ---> AT+CREG?
        <--- +CREG: 0,0
 Accelerometer ADXL345 x,y,z: 0,0,0,
        ---> AT+CGNSINF
        <--- +CGNSINF: 1,1,20160422215047.000,41.488178,-71.526338,66.300,0.48,173.1,1,,1.2,
        ---> AT+CGNSINF
        <--- +CGNSINF: 1,1,20160422215047.000,41.488178,-71.526338,66.300,0.48,173.1,1,,1.2,
GPS latitude:41.488174
GPS longtitude:-71.526329
GPS speed KPH:0.48
GPS speed MPH:0.30
GPS heading:173.10
GPS altitude:66.30
Checking for Cell network...
        ---> AT+CREG?
        <--- +CREG: 0,0
 Accelerometer ADXL345 x,y,z: 0,0,0,
        ---> AT+CGNSINF
        <--- +CGNSINF: 1,1,20160422215050.000,41.488175,-71.526338,66.000,0.35,208.8,1,,1.2,
        ---> AT+CGNSINF
        <--- +CGNSINF: 1,1,20160422215051.000,41.488173,-71.526340,65.900,0.28,320.2,1,,1.2,
GPS latitude:41.488166
GPS longtitude:-71.526329
GPS speed KPH:0.28
GPS speed MPH:0.17
GPS heading:320.20
GPS altitude:65.90
Checking for Cell network...
        ---> AT+CREG?
        <--- +CREG: 0,0
 Accelerometer ADXL345 x,y,z: 0,0,0,
        ---> AT+CGNSINF
        <--- +CGNSINF: 1,1,20160422215054.000,41.488173,-71.526343,65.800,0.35,236.5,1,,1.2,
        ---> AT+CGNSINF
        <--- +CGNSINF: 1,1,20160422215054.000,41.488173,-71.526343,65.800,0.35,236.5,1,,1.2,
GPS latitude:41.488166
GPS longtitude:-71.526344
GPS speed KPH:0.35
GPS speed MPH:0.22
GPS heading:236.50
GPS altitude:65.80
Checking for Cell network...
        ---> AT+CREG?
        <--- +CREG: 0,0
```

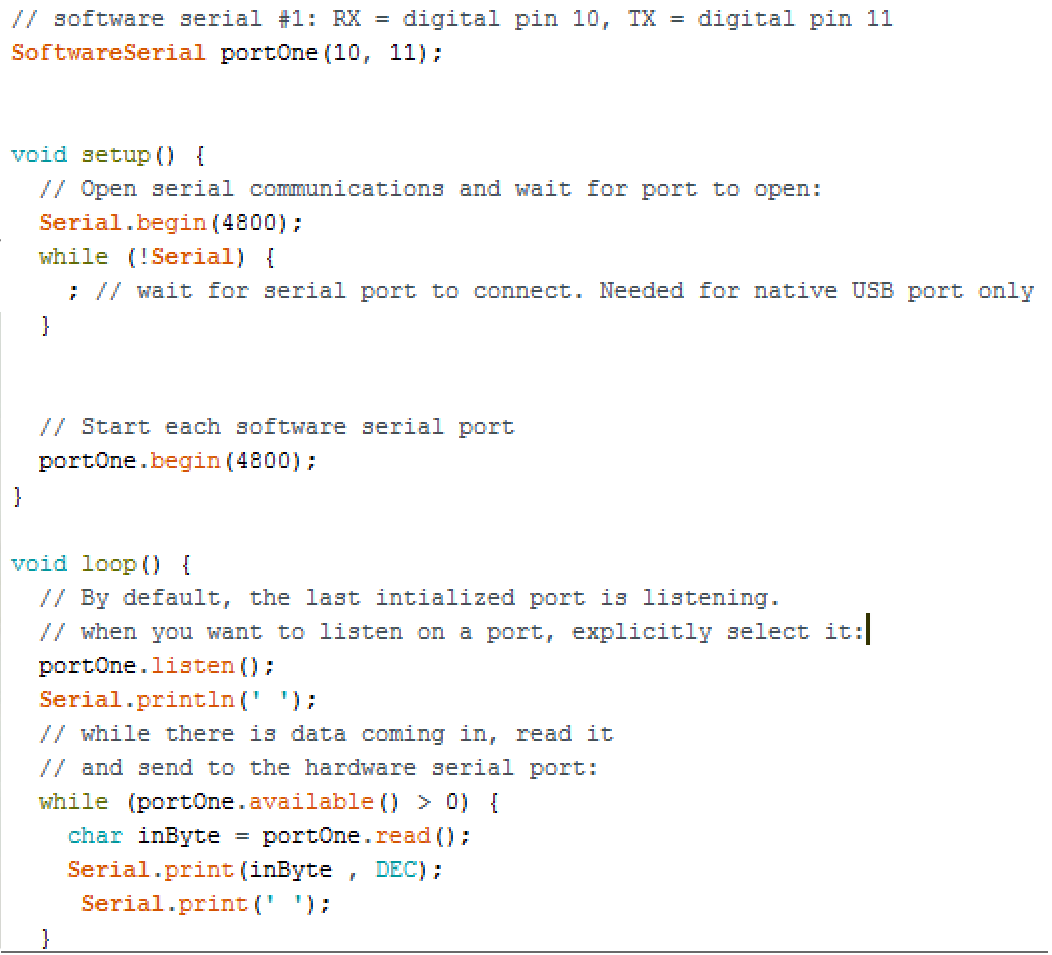
```
// 3.3v - CS
// Analog 4 - SDA
// Analog 5 - SLC
//==========================================================================
#include "Adafruit_FONA.h" //Fona added to the sketch
#include <Wire.h> //Will allow us to communicate with I2C /TWI devices (i.e ADXL345)
//--------------------------------------------------------------------------

//==========================================================================
// standard pins for the shield, adjust as necessary
#define FONA_RX 2
#define FONA_TX 3
#define FONA_RST 4
// We default to using software serial. If you want to use hardware serial
// (because softserial isnt supported) comment out the following three lines
// and uncomment the HardwareSerial line

#include <SoftwareSerial.h> //We default using the software serial
SoftwareSerial fonaSS = SoftwareSerial(FONA_TX, FONA_RX);
SoftwareSerial *fonaSerial = &fonaSS;

//--------------------------------------------------------------------------
#define DEVICE (0x53)      //ADXL345 device address
//==========================================================================
byte _buff[6] ;        //6 bytes buffer for saving data read from the device
char str[512];             //string buffer to transform data before sending it to the serial port
char POWER_CTL = 0x2D; //Power-saving features control
char DATA_FORMAT = 0x31;
char DATAX0 = 0x32; //X-Axis Data 0
char DATAX1 = 0x33; //X-Axis Data 1
char DATAY0 = 0x34; //Y-Axis Data 0
char DATAY1 = 0x35; //Y-Axis Data 1
char DATAZ0 = 0x36; //Z-Axis Data 0
char DATAZ1 = 0x37; //Z-Axis Data 1
int x, y, z;                         //three axis acceleration data
double roll = 0.00, pitch = 0.00;       //Roll & Pitch are the angles which rotate by the axis X and y
//in the sequence of R(x-y-z),more info visit
// https://www.dfrobot.com/wiki/index.php?title=How_to_Use_a_Three-Axis_Accelerometer_for_Tilt_Sensing

//==========================================================================
```

Done uploading.

```
Sketch uses 16,982 bytes (52%) of program storage space. Maximum is 32,256
```

CruzPro “Active” Thru-Hull Depth Transducer

- NMEA 0183 protocols
- Max. depth 450-feet



```
// software serial #1: RX = digital pin 10, TX = digital pin 11
SoftwareSerial portOne(10, 11);


void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(4800);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }


  // Start each software serial port
  portOne.begin(4800);
}

void loop() {
  // By default, the last intialized port is listening.
  // when you want to listen on a port, explicitly select it:
  portOne.listen();
  Serial.println(' ');
  // while there is data coming in, read it
  // and send to the hardware serial port:
  while (portOne.available() > 0) {
    char inByte = portOne.read();
    Serial.print(inByte , DEC);
     Serial.print(' ');
  }
```

Results for Depth Transducer

$SDDBT,015.7,f,004.8,M,002.6,F*0D <-- **Depth in Feet, Meters and Fathoms**

$SDMTW,023.8,C*3D <-- **Water temperature in degrees C**

$SDDPT,004.8,*75 <-- **Depth in Meters**

```
91 117 -35 125 93 -99 62 -97 -113 -93 -101 -89 -85 -111 -103 -27 -21 -73 89 119 119 123 87 0 -102 -74 54 54 107 -89 51 -89 62
-97 -113 -93 -101 -89 101 0 -102 -10 118 54 -69 -89 115 -85 -97 -99 -27 -21 0

91 117 -35 125 93 -99 62 -97 -99 -93 -111 -89 -85 -111 115 -27 -21 -73 89 119 119 123 87 0 -102 -10 86 54 -85 -89 51 -89 62 -
97 -99 -93 -111 -89 101 -128 -102 -10 -10 54 107 -89 115 -85 -97 -115 -27 -21 0
```

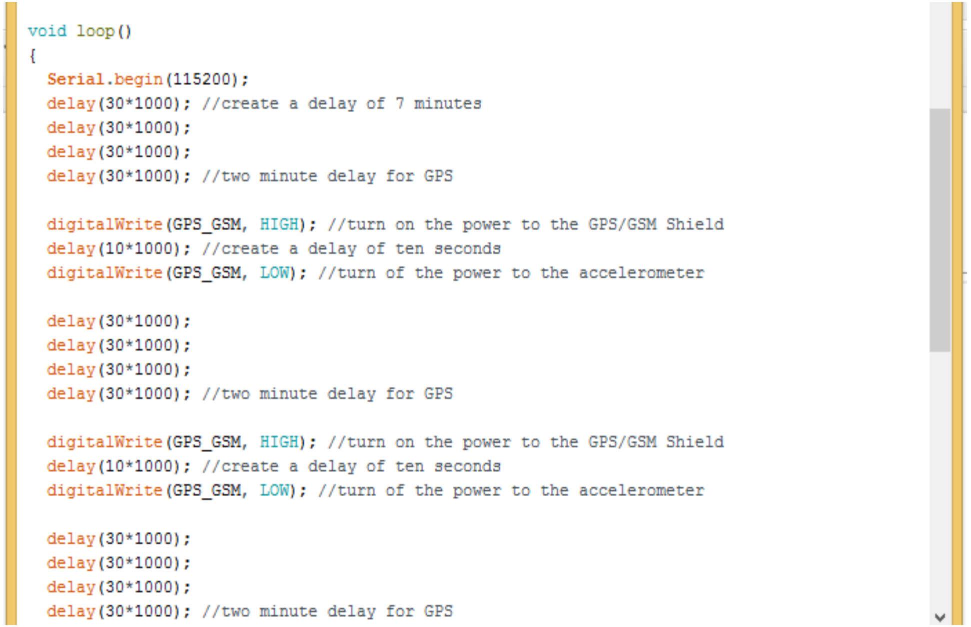
Power Analysis

- 12-VDC battery → 2000 mAH
- Transducer → 35 mA current draw
- Accelerometer → 0.04 mA current draw
- GPS/GSM Shield → 20 mA current draw
- Arduino Mega → 10 mA current draw

Andrew Osmanski

Power Regulation

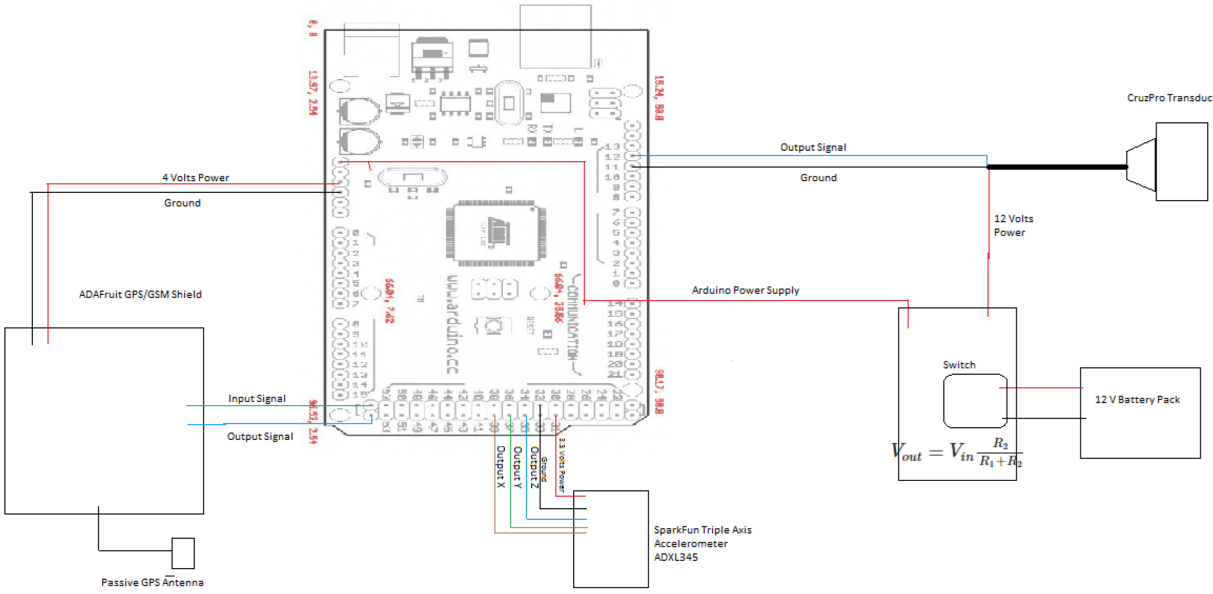
- Arduino Mega
  - Constant Power
- GPS/GSM Shield
  - Two Minute Delay
- Depth Transducer/Accelerometer
  - Eight Minute Delay



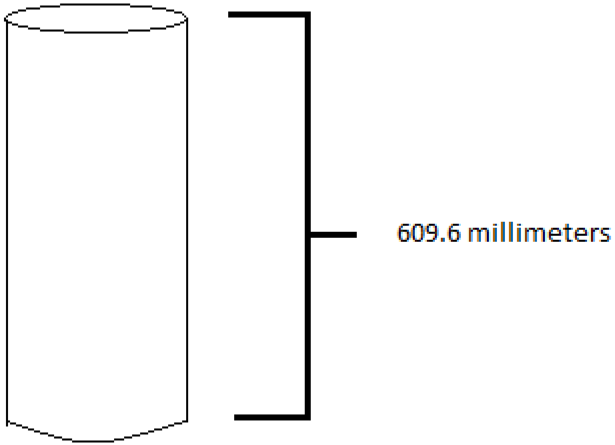
```
void loop()
{
  Serial.begin(115200);
  delay(30*1000); //create a delay of 7 minutes
  delay(30*1000);
  delay(30*1000);
  delay(30*1000); //two minute delay for GPS

  digitalWrite(GPS_GSM, HIGH); //turn on the power to the GPS/GSM Shield
  delay(10*1000); //create a delay of ten seconds
  digitalWrite(GPS_GSM, LOW); //turn of the power to the accelerometer

  delay(30*1000);
  delay(30*1000);
  delay(30*1000);
  delay(30*1000); //two minute delay for GPS

  digitalWrite(GPS_GSM, HIGH); //turn on the power to the GPS/GSM Shield
  delay(10*1000); //create a delay of ten seconds
  digitalWrite(GPS_GSM, LOW); //turn of the power to the accelerometer

  delay(30*1000);
  delay(30*1000);
  delay(30*1000);
  delay(30*1000); //two minute delay for GPS
```

Andrew Osmanski

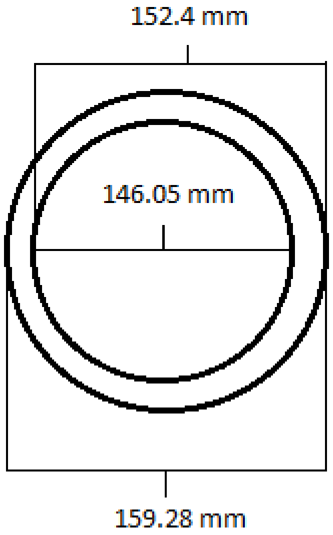
Schematic Diagram



Device Housing

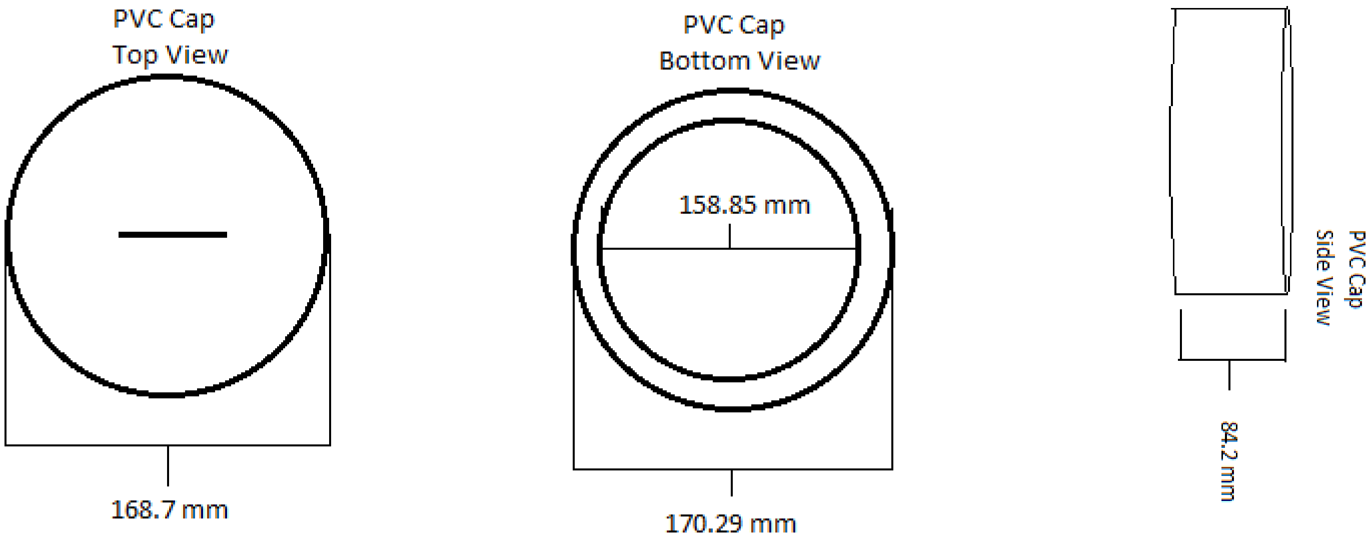


6-inch Diameter PVC Pipe

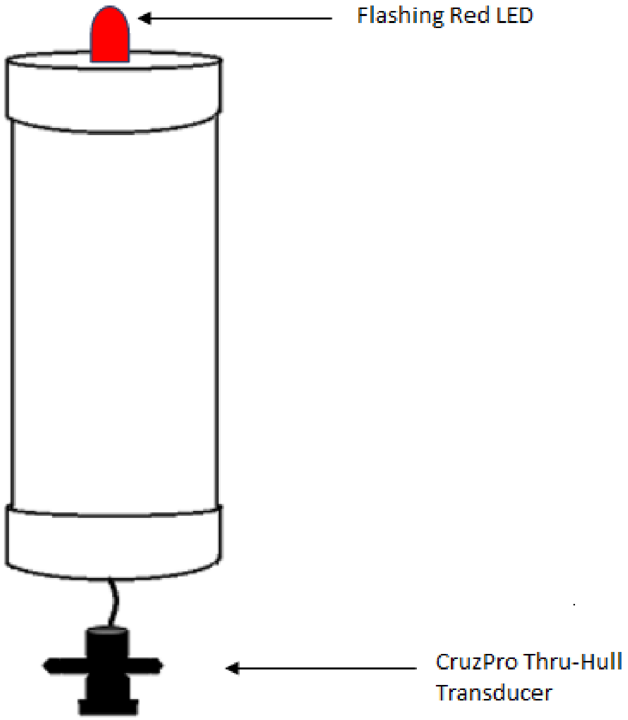


6-inch Diameter PVC Pipe
Top View

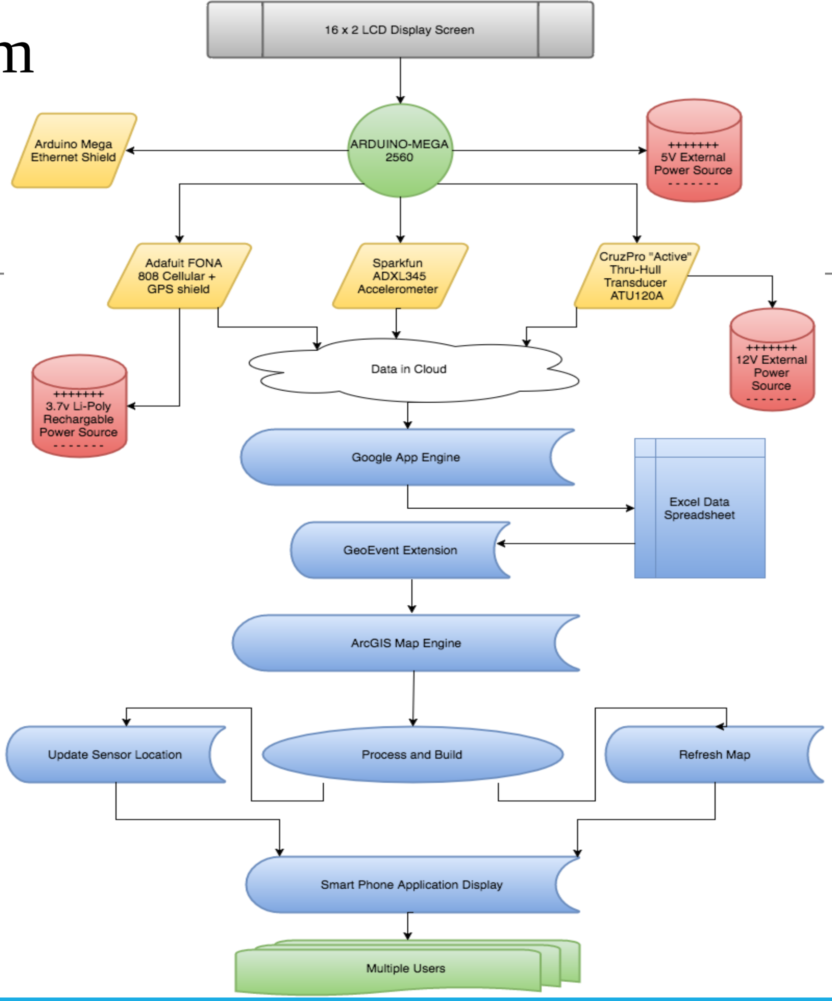
Device Housing



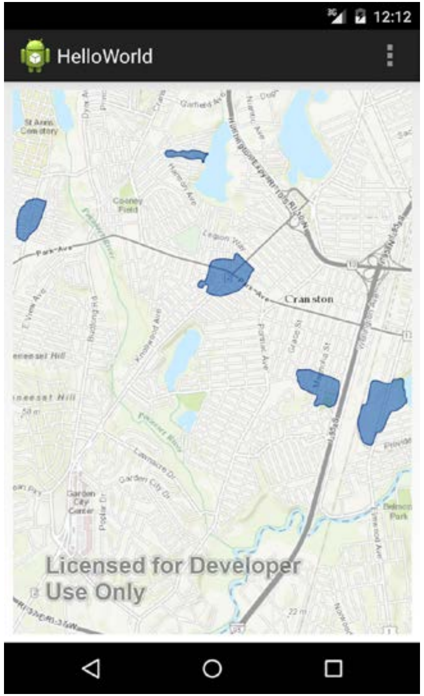
Device Housing

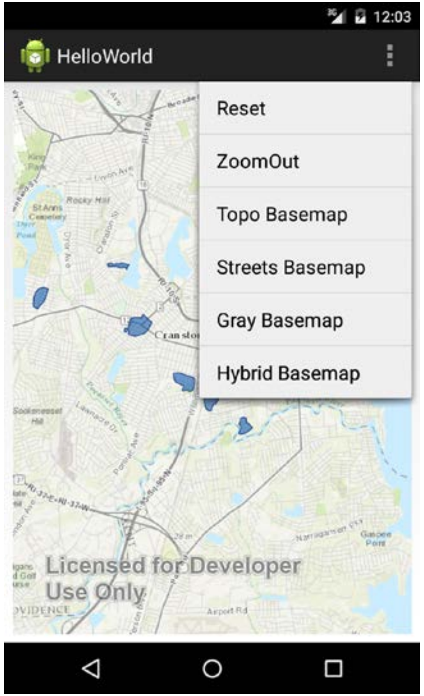


Flow chart Diagram


16 x 2 LCD Display Screen
ARDUINO-MEGA 2560
Arduino Mega Ethernet Shield
+++++++ 5V External Power Source - - - - - - -
Adafuit FONA 808 Cellular + GPS shield
Sparkfun ADXL345 Accelerometer
CruzPro "Active" Thru-Hull Transducer ATU120A
+++++++ 12V External Power Source - - - - - - -
+++++++ 3.7v Li-Poly Rechargable Power Source - - - - - - -
Data in Cloud
Google App Engine
Excel Data Spreadsheet
GeoEvent Extension
ArcGIS Map Engine
Update Sensor Location
Process and Build
Refresh Map
Smart Phone Application Display
Multiple Users

Mobile Application

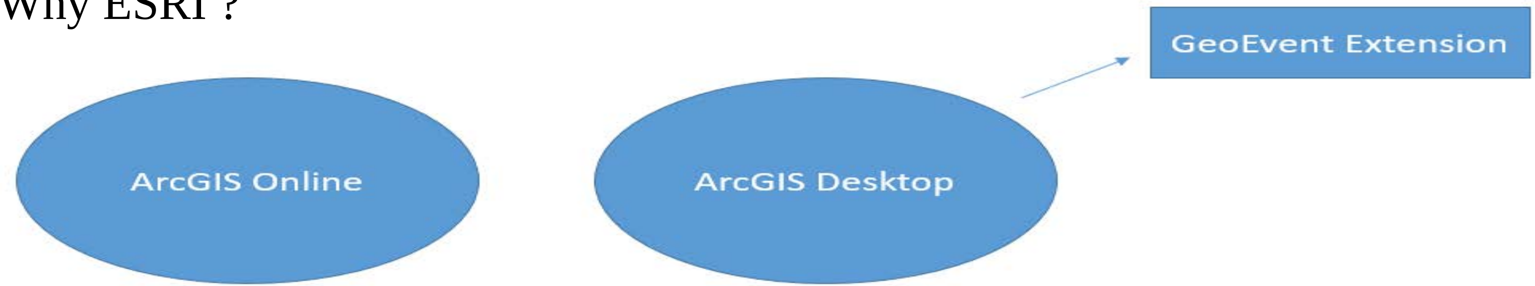




Maps with ESRI

ESRI: Geographic information system company

Why ESRI ?



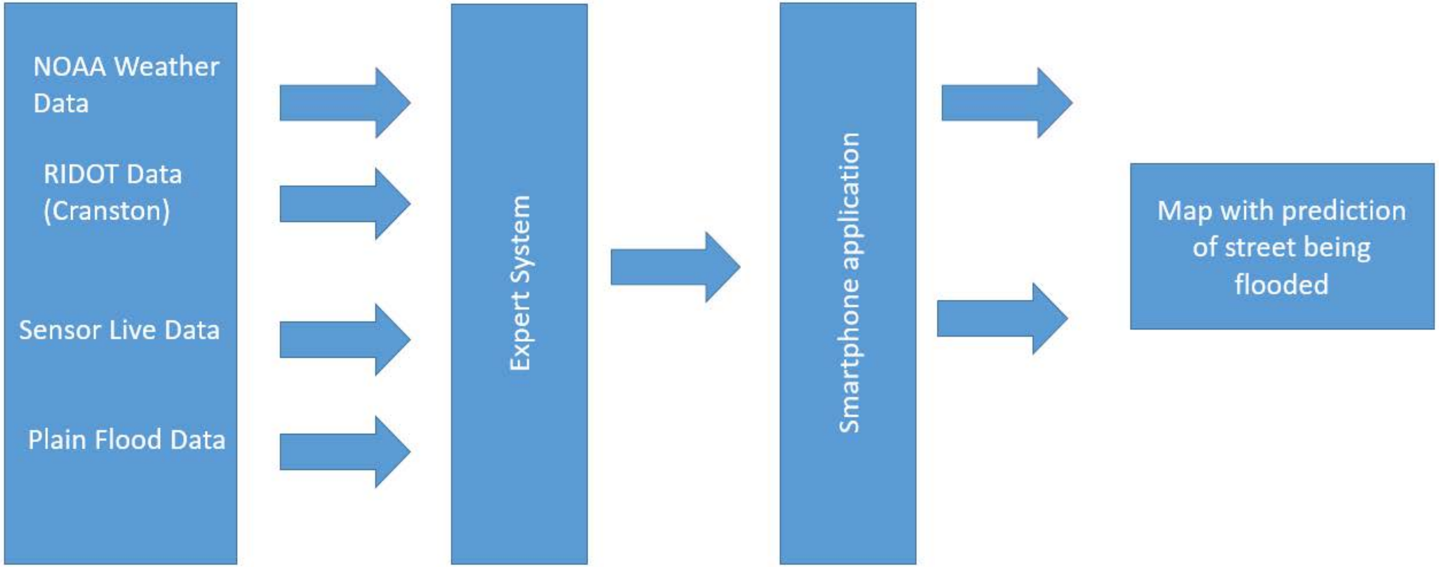
Publish map online

Create map

Add layers

Create Geospatial data

Software Architecture



Map Layers

FEMA 100 Flood data

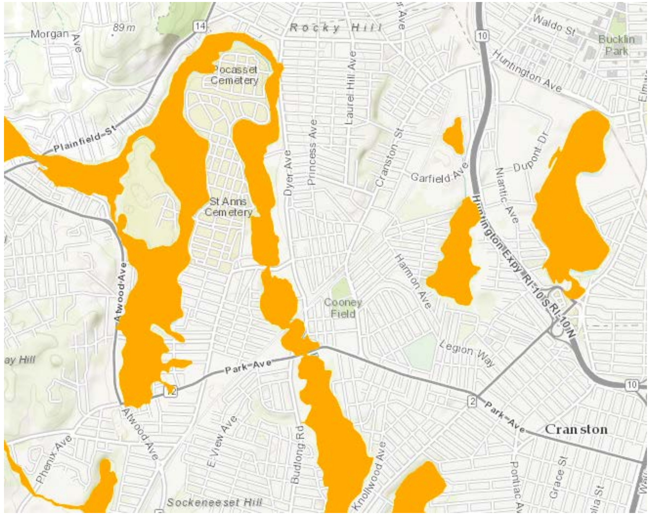
USGS Stream Gauges

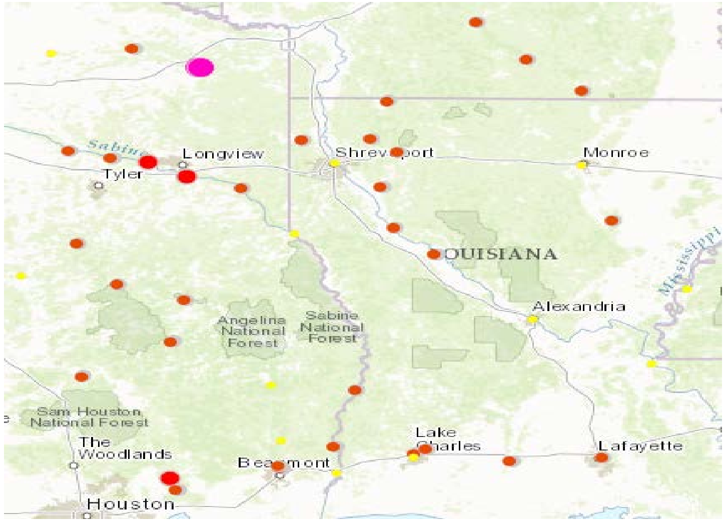
● Major flooding

● Moderate flooding

● Minor flooding

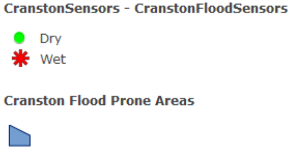
● Near flood stage





Map Layers

DOT Filters



CranstonSensors - CranstonFloodSensors

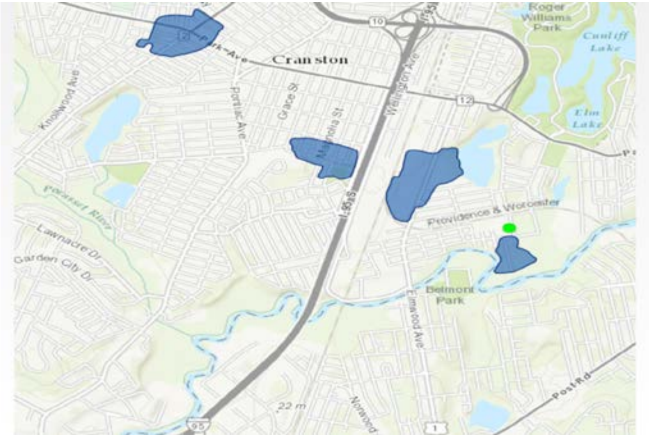
● Dry

✱ Wet

Cranston Flood Prone Areas

▨

City of Cranston Base Map



Sockeneeset Hill
Woodstock Ln
Church Hill Dr
Cranston St
58 m
Aqueduct Rd
Delway Rd
Leo Castiglioni Field
Pocasset River
Autumn St
Greening Ln
Glen Ridge Rd
Deerfield Rd
Budlong Rd
Bennington Rd
Wayside Dr
Crocus Dr
Hollyhock Dr
Whitewood Dr
Redwood Dr
Lawnacre Dr
Meshanticut Valley Pky
Applegate Rd
Dean Pky
S Hill Dr
E Hill Dr
Poppy Dr
Hybrid Dr
Begonia Dr
Iris Dr
Perennial Dr
Garden City Dr
22 m
Elm Dr
May Dr
Summit Dr
E Bel Air Rd
Glen View Dr
Susan Dr
Garden City
Althea Dr
Capuano Ave
W Blue Ridge Rd
Garden City Center
Poplar Dr
Cypress Dr
Coldbrook Dr
Alderbrook Dr
54 m
New London Ave
Midway Rd
Michael Dr
Belvedere Dr
Plantation Dr
Eden Crest Dr
Harvard St
Hoffman Ave
RI-37 E
RI-37 W
Long Ct
Laurel Dr
Sweetbriar
Mountain
Chapel View Blvd
Sockanosset Cross Rd
Dellwood Rd
Zinnia Dr
Eddy St
Webb
5
2

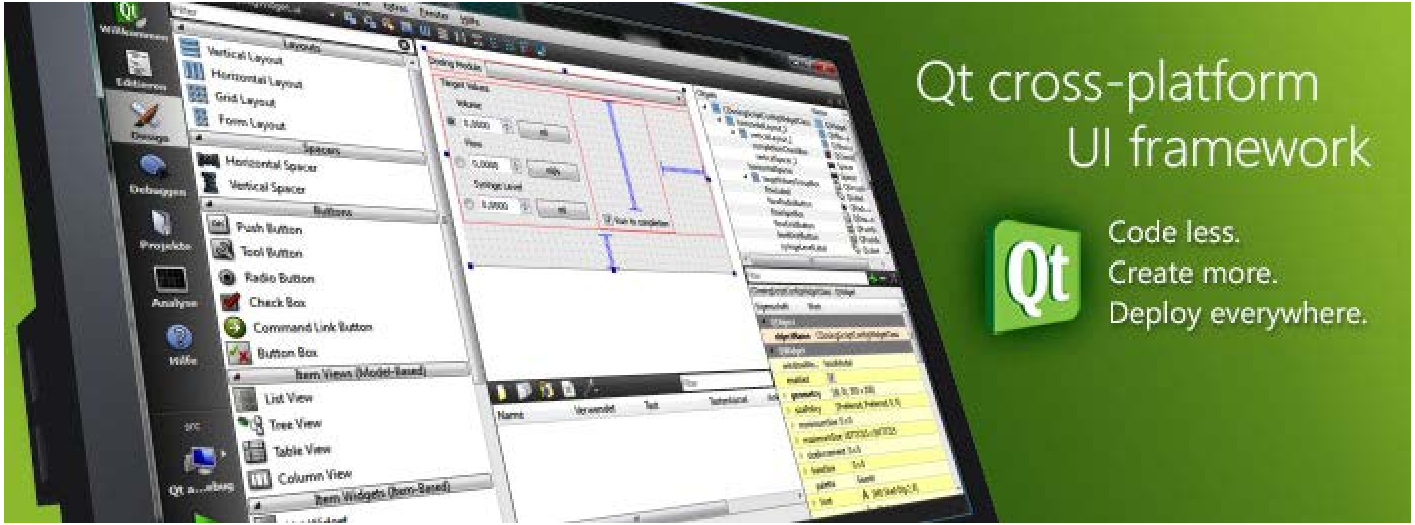
Design Constraints and Solutions

Multiple users one map

- Map Layers preference
  - Maps with all combinations of the layers.
  - Point to a map matches the desired layers
- One server
  - Multiple servers host Map Engine
  - load balancer to balance the traffic among servers
- Out of sync maps
  - Equal rate of accessing data file
  - Equal Refresh rate among all servers

Demo

Future Considerations



Future Considerations

Completed Housing

Contaminant Sensor

ABET Outcome C

Economy

Safety

Questions?