Τεχνικές Διασύνδεσης

9ο εξάμηνο

Παναγιωτακόπουλος Θεόδωρος theopana3@ee.duth.gr
Μπαρμπέρης Δημήτρης dimibarb2@ee.duth.gr
Αγγελής Τζούχας angetzou@ee.duth.gr

1 Εισαγωγή

Το project που κατασκευάσαμε στα πλαίσια του μαθήματος «Τεχνικές Διασύνδεσης Ψηφιακών Συστημάτων» αποτελεί ένα σύστημα για την ανίχνευση εισβολέων σε ένα χώρο, καθώς και την ανίχνευση του φυσικού κινδύνου της φωτιάς. Το σύστημα αυτό μπορεί να επιβλέπεται από έναν απομακρυσμένο χρήστη διαδικτυακά.
Αναλυτικότερα, χρησιμοποιήσαμε δύο συσκευές οι οποίες επικοινωνούν μεταξύ τους. Η πρώτη ανιχνεύει την παραβίαση μιας πόρτας ή παραθύρου, την κίνηση και την ύπαρξη καπνού, ενώ η δεύτερη μετράει τη θερμοκρασία και στέλνει όλες τις μετρήσεις των αισθητήρων σε μία σελίδα που δημιουργήσαμε στο διαδίκτυο.

Για την υλοποίηση του project χρειάστηκαν τα εξαρτήματα:

- 2 Arduino Uno
- 1 Ethernet Shield
- 1 αισθητήρα θερμοκρασίας (TMP36)
- magnetic door switch
- RF (radio frequency) 315 MHz ζεύγος πομπού-δέκτη (transmitter-receiver module)
- 1 αισθητήρα απόστασης – κίνησης (ultrasonic sensor)
- 1 ρελέ (relay)
- 1 λάμπα
- 1 αισθητήρα καπνού (MQ2 gas sensor)
- 1 buzzer
- 2 Breadboards
- καλώδια

Τα παραπάνω υλικά τα προμηθευτήκαμε από το κατάστημα Hellas Digital στην Αθήνα και το συνολικό τους κόστος ήταν 54 ευρώ.

2 Συσκευή 1η – Ανιχνευτής εισβολέα, καπνού

Αυτή η συσκευή περιλαμβάνει το ένα Arduino, αισθητήρα απόστασης – κίνησης, αισθητήρα καπνού (MQ2 gas sensor), magnetic door switch, το ρελέ με τη λάμπα, buzzer κι έναν πομπό RF (radio frequency).

Ο **αισθητήρας απόστασης – κίνησης** λειτουργεί εκπέμποντας υπερήχους οι οποίοι ανακλώνται πάνω στα αντικείμενα που βρίσκονται μπροστά του και επιστρέφουν στον αισθητήρα, ο οποίος από τη χρονική διαφορά τις εκπομπής και της λήψης υπολογίζει την απόσταση. Αυτόν τον αισθητήρα τον χρησιμοποιούμε για την ανίχνευση κίνησης μέσω της μεταβολής απόστασης. Δέχεται γείωση GND, echo, trigger και Vcc. Τοποθετούμε το GND στο GND του Arduino, το VCC συνδέεται με τάση ίση με 5V, το Echo με το 12ο Digital pin, ενώ το Trigger καταλήγει στο 13ο Digital pin. Επίσης, για τον συγκεκριμένο αισθητήρα χρησιμοποιήθηκε μία επιπλέον βιβλιοθήκη (library). Αυτή ονομάζεται NewPing και μπορεί να βρεθεί στην σελίδα της Arduino στο παρακάτω link: `https://bitbucket.org/teckel12/arduino-new-ping/downloads` .

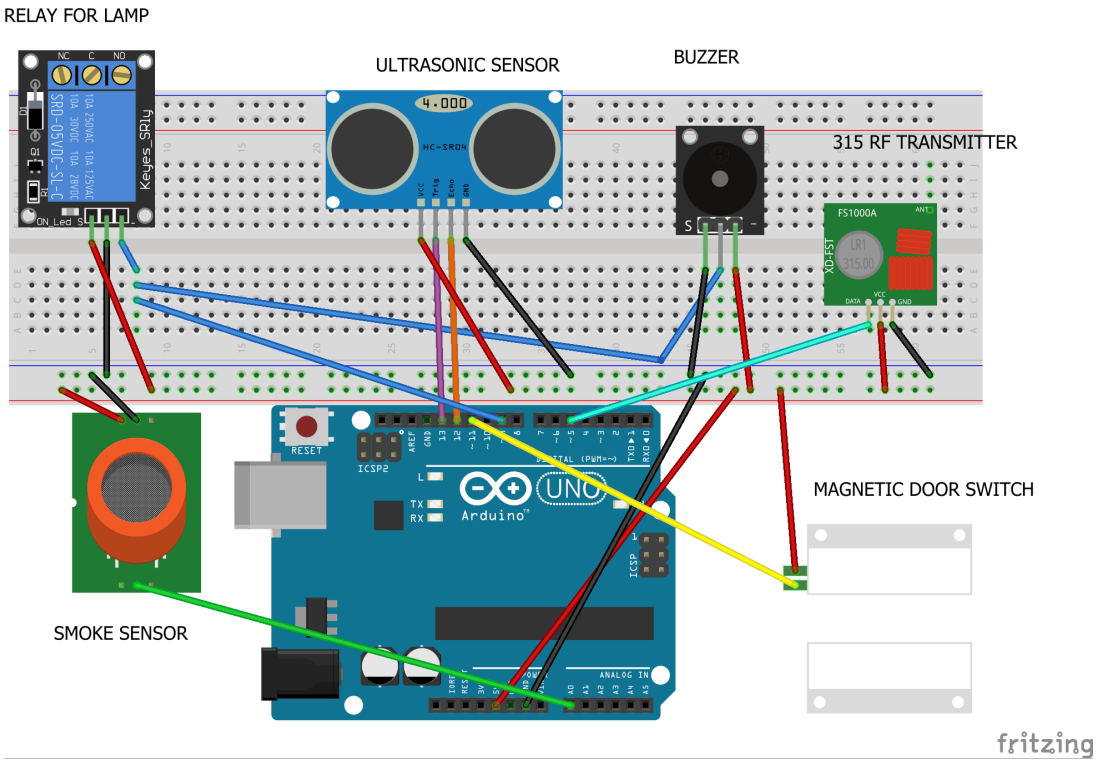
Ο **αισθητήρας καπνού**[2] λειτουργεί με το φαινόμενο απορρόφησης ακτινοβολίας. Ο καπνός μεταβάλλει τη σύσταση του υλικού που βρίσκεται μέσα στον αισθητήρα αλλάζοντας την αντίστασή του κι έτσι η τάση εξόδου Vout μεταβάλλεται, με υψηλότερη τάση εξόδου να σημαίνει περισσότερος καπνός. Το ποτενσιόμετρο που βρίσκεται πίσω από τον αισθητήρα ελέγχει την ευαισθησία του. Δέχεται ακόμη (Analog pin) A0, (Digital pin) D0, τροφοδοσία Vcc και γείωση GND. Στην παρούσα εργασία χρησιμοποιούμε το A0 του αισθητήρα και το ενώνουμε στο αντίστοιχο A0 (Analog pin 0) του Arduino και με την VCC και GND στα 5V και στο GND του Arduino αντίστοιχα.

Ο **αισθητήρας magnetic door switch** που ανιχνεύει το άνοιγμα της πόρτας. Όταν τα δύο εξαρτήματα έρχονται σε επαφή έλκονται οι μαγνητικές επαφές και το κύκλωμα κλείνει. Σε διαφορετική περίπτωση το κύκλωμα είναι ανοιχτό. Έχει δύο συνδέσεις. Η μία ενώνεται στην τάση που είναι ίση με 5 Volt και η άλλη πηγαίνει στο D11 του Digital In.

Το **ρελέ** εμφανίζει συμπεριφορά διακόπτη σε κυκλώματα υψηλής τάσης (πχ 220V). Ο διακόπτης μπορεί να ελεγχθεί από ένα control pin, παίρνει τροφοδοσία 5V (Vcc) και γείωση (GND). Τα δύο πρώτα συνδέονται στην τάση 5 V και GND αντίστοιχα, ενώ το VIN συνδέεται στο Digital pin 9 του Arduino.

Ο **buzzer** έχει ένα control pin που όταν δεχθεί μηδενική τάση παράγει ήχο, τροφοδοσία (Vcc τοποθετούμε 5V) και γείωση (GND). Το control pin (I-O) του buzzer το συνδέουμε στο Digital pin 9 του Arduino.

Ο **RF transmitter**[1] χρησιμοποιείται για την επικοινωνία με τον RF receiver ο οποίος βρίσκεται στη δεύτερη συσκευή. Δέχεται 3 pins, Data, Vcc, GND. Το Data συνδέεται στο Digital 5 του Arduino, το Vcc στο 5V και το GND στη γείωση. Αφιερώνεται ξεχωριστή ενότητα στην επικοινωνία μέσω του RF Module παρακάτω.



Σχήμα 1: Κύκλωμα 1ης συσκευής

Καλωδίωση

- Αισθητήρας Απόστασης - Κίνησης

  GND –> GND, ECHO –> PIN D12, TRIGGER –> PIN D13, VCC –> 5V

- Αισθητήρας Καπνού

  Vcc –> 5V, A0 –> A0, GND –> GND, D0 –> NO USE

- Ρελέ

  Vcc –> 5V, GND –> GND, Vin –> D9

- Buzzer

  Vcc –> 5V, GND –> GND, I-O pin –> D9

- RF - πομπός

  Vcc –> 5V, GND –> GND, Data –> D5

Ακολουθεί ο κώδικας της 1 ης συσκευής

```
#include <NewPing.h>
#include <VirtualWire.h>
#define TRIGGER_PIN  13  // Arduino pin tied to trigger pin on the ultrasonic sensor.
#define ECHO_PIN     12  // Arduino pin tied to echo pin on the ultrasonic sensor.
#define MAX_DISTANCE 200 // Maximum distance we want to ping for (in centimeters).
#define SENS 16  // How sensitive the alarm is in the change of distance detected by the utrasonic (distance) sensor.
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);
char controller[4];
const int gasPin = A0; //GAS sensor output pin to Arduino analog A0 pin
const int buzzPin = 9; //Buzzer control pin to Arduino digital 9 pin
const int doorPin = 11; // Door switch pin to Arduino digital 11 pin
const int transPin = 5; // RF transmitter Data pin to Arduino digital 5 pin
const int loops[] = {20,10,8};// For how many loops the notification for detected gas, motion and door respectively will last
int loopcounter[] = {0,0,0};
boolean gastrigger = false; // Gas notification (initialized to false)
boolean motiontrigger = false; // Motion notification (initialized to false)
boolean doortrigger = false; // Door notification (initialized to false)
int prev = -1; //Previous value of distance sensor
int sendevery = 0;

void setup()
{
  Serial.begin(9600); //Initialize serial port - 9600 bps
        pinMode(buzzPin,OUTPUT); //Set buzzPin to OUTPUT
        pinMode(doorPin,INPUT);
        digitalWrite(buzzPin,HIGH); //Initialization of buzzPin to HIGH
        vw_set_ptt_inverted(true);
        vw_set_tx_pin(transPin); // The pin where the data are sent in order to be transmitted (RF)
        vw_setup(4000);// speed of data transfer Kbps
}
void read(){
  int avg = 0;
  int i=0;
  unsigned int uS;
  while (i<5){ //Get 5 non-zero values
    delay(25);
   uS = sonar.ping(); //Get distance value
   if (uS > 0){
     avg += uS/US_ROUNDTRIP_CM;  // Convert ping time to distance in cm and add it to the avg
      i++;
   }
  }
  avg /= 5; //Gets average distance
  Serial.print("Ping: ");
  Serial.print(avg);
  Serial.println("cm");
  if (prev == -1){prev = avg;} // If first measurement
  int temp = abs(prev - avg); // Get the distance difference
  if (temp > SENS){ // If distance difference bigger than sensitivity
    motiontrigger = true;loopcounter[1] = 0; // Enable motion notification and initialize its counter
    Serial.println("Movement");
  }
  prev = avg;
}

void loop()
{       read();
        int svalue = analogRead(gasPin); //Get gas value in V
```

```
59          Serial.print("Smoke :");
60    Serial.println(svalue);
61          int dvalue = digitalRead(doorPin); //Get door value (0 or 1)
62          if ( dvalue == LOW){
63            Serial.println("Door Alert");
64            doortrigger = true; loopcounter[2] = 0;
65          }
66    delay(300); // Print value every 1 sec.
67          if (svalue > 205){ // Enable gas notification if gas value exceeds 205
68             Serial.println("Gas Alert");
69             gastrigger = true; loopcounter[0] = 0;
70          }
71          unsigned int state = 0;
72          // Keep notifications for #loops
73          // Calculate our current state depending on which notifications are enabled
74          if (gastrigger){
75           loopcounter[0]++;
76           state= state +1;
77          }
78          if (motiontrigger){
79           loopcounter[1]++;
80            state = state +2;
81          }
82          if (doortrigger){
83           loopcounter[2]++;
84           state = state +4;
85          }
86          if (loopcounter[0] == loops[0]){gastrigger=false;}
87          if (loopcounter[1] == loops[1]){motiontrigger=false;}
88          if (loopcounter[2] == loops[2]){doortrigger=false;}
89
90          if(gastrigger || motiontrigger || doortrigger){ //For any enabled notification
91            digitalWrite(buzzPin, LOW); // trigger the buzzer
92          }else{
93            digitalWrite(buzzPin,HIGH);
94          }
95          // Print our current notification state
96          Serial.print("gas: ");Serial.print(gastrigger);Serial.print(", motion: ");
      Serial.print(motiontrigger);Serial.print(", door :");Serial.println(doortrigger);
97          if (sendevery == 4){ // Every 4 loops send (RF) the current state
98          sendevery=0;
99          itoa(state,controller,10); // Convert integer to char array
100         Serial.println("Sending");
101         vw_send((uint8_t *)controller,4); // Transmitt
102         vw_wait_tx(); // Wait until the whole message is gone
103         Serial.println("Sent");
104         }
105         sendevery++;
106   }
```

Σχόλια κώδικα 1ης συσκευής

Ο κώδικας παρουσιάζεται με σχόλια, ωστόσο θα θέλαμε να τονίσουμε ιδιαίτερα ορισμένα σημεία που μας δυσκόλεψαν. Παρατηρήσαμε ότι ο αισθητήρας απόστασης – κίνησης αφενός εμφανίζει ως έξοδο σε τακτά χρονικά διαστήματα μηδενικά τα οποία δεν ανταποκρίνονται στην απόσταση που «βλέπει» και αφετέρου οι μετρήσεις απόστασης είναι ασταθείς. Για την επίλυση των δύο παραπάνω προβλημάτων αφενός παίρνουμε μόνο τις τιμές μεγαλύτερες του μηδενός και αφετέρου υπολογίζουμε τον μέσο όρο έξι μη μηδενικών μετρήσεων. Τα παραπάνω υλοποιούνται στη συνάρτηση *read()* (βλ. γραμμές *31-40*). Επιπλέον, για τη μεταφορά δεδομένων μέσω RF με την εντολή *vw_send* απαιτείται να δώσουμε ως είσοδο έναν πίνακα από bytes. Η εργασία μας απαιτεί τη μεταφορά ενός ακέραιου αριθμού (ο οποίος δηλώνει τη τρέχουσα κατάσταση των αισθητήρων) ο οποίος μετατρέπεται σε πίνακα χαρακτήρων (char array) μέσω της συνάρτησης *itoa()*, γιατί *char = byte*. Αυτό υλοποιείται στις γραμμές *99-102*.

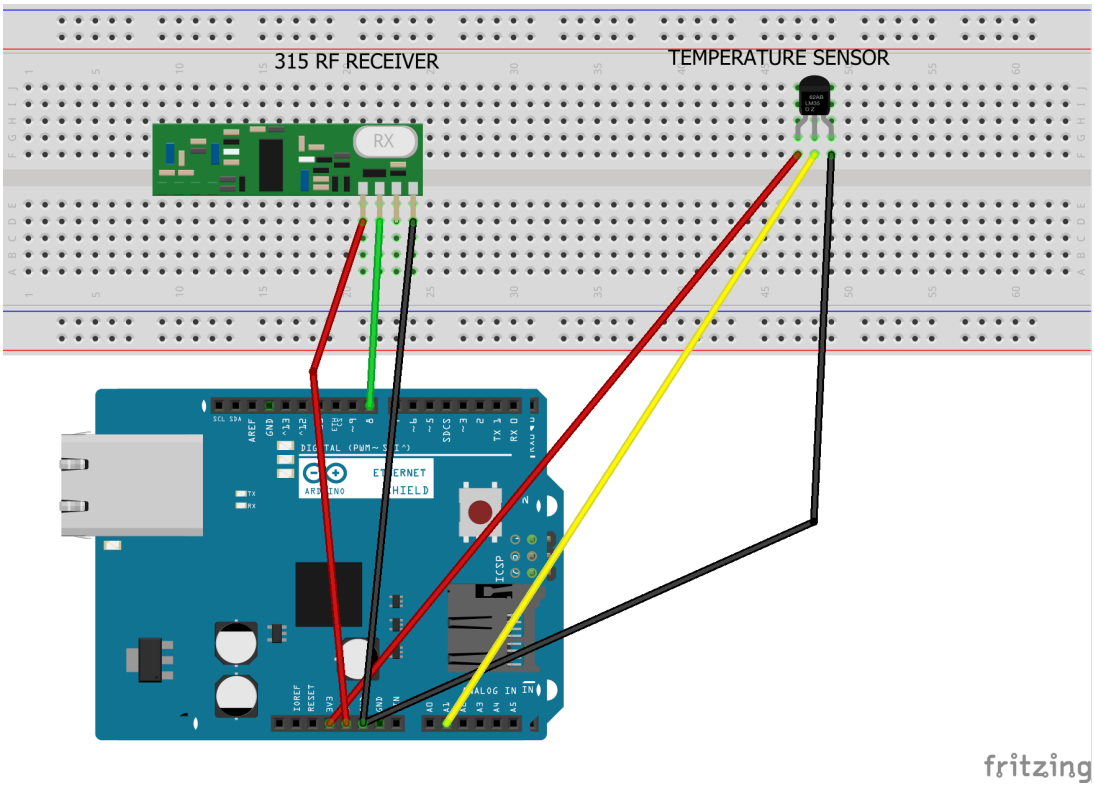
3 Συσκευή 2η – Μέτρηση θερμοκρασίας, επικοινωνία με το διαδίκτυο

Αυτή η συσκευή περιλαμβάνει ένα Arduino, αισθητήρα θερμοκρασίας, τον δέκτη RF και το Ethernet Shield.

Ο **αισθητήρας θερμοκρασίας**[3] μετράει τη θερμοκρασία του περιβάλλοντος μέσω της τάσης εξόδου η οποία είναι ανάλογη με τη μεταβολή της θερμοκρασίας σε βαθμούς Celsius. Δέχεται 3 pins, τροφοδοσία Vcc (την τροφοδοτούμε με τάση 3.3V από το pin Vcc του Ethernet Shield), τάση εξόδου (Vout), γείωση GND. Ο ακροδέκτης της τάσης εξόδου Vout του αισθητήρα συνδέεται με το A1 του Analog In του Arduino.

Το **Ethernet Shield**[4] χρησιμοποιείται για να επικοινωνήσει το Arduino με το Διαδίκτυο.

Ο **RF receiver**[1] χρησιμοποιείται για την επικοινωνία με τον RF transmitter ο οποίος βρίσκεται στην πρώτη συσκευή. Ο αισθητήρας RF receiver έχει 4 pins, τα οποία είναι με την σειρά από αριστερά προς τα δεξιά VCC, DATA, DATA και GND. Στην παρούσα εργασία χρησιμοποιούνται το VCC και το GND καθώς και ένα εκ των δύο DATA. Το VCC συνδέεται σε θύρα τάσης 5V. Το GND στη γείωση. Τέλος, το DATA port συνδέεται στο όγδοο Digital pin (D8). Αφιερώνεται ξεχωριστή ενότητα στην επικοινωνία μέσω του RF Module παρακάτω.



Σχήμα 2: Κύκλωμα 2ης συσκευής

Καλωδίωση

- Αισθητήρας θερμοκρασίας

  Vcc –> 3.3V, GND –> GND, Vout –> A1

- RF δέκτης

  Vcc –> 5V, GND –> GND, Data –> D8

Ακολουθεί ο κώδικας της 2 ης συσκευής

```
#include <SPI.h>
#include <Ethernet.h>
#include <VirtualWire.h>
char msg[4];
EthernetClient client;
boolean actions[] = {false,false,false}; // Notifications array
float temp;
// MAC address for the controller
byte mac[] = { 0x9E, 0xAD, 0xBA, 0xEF, 0xFE, 0xED };
char server[] = "www.edged.xyz";     // name address for server (using DNS), here put your URL
const int tempPin = 1; // Temperature pin -> Arduino analog pin 1
// Set the static IP address, dnServer, subnet, gateway
IPAddress ip(192, 168, 1, 95);
IPAddress dnServer(208,67,222,222);
IPAddress subnet(255,255,255,0);
IPAddress gateway(192,168,1,1);

void setup() {
 // Open serial communications and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect
  }
  vw_set_ptt_inverted(true); // Required for DR3100
  vw_set_rx_pin(8);
  vw_setup(4000);  // Bits per sec
  vw_rx_start();
  job();
  // start the Ethernet connection:
  Ethernet.begin(mac, ip, dnServer, gateway, subnet);
  delay(4000);
  //Call the function that will handle the connection:
  conn();
}

void loop() {
  // if there are incoming bytes available
  // from the server, read them and print them:
  if (client.available()) {
    char c = client.read();
    Serial.print(c);
  }

  // if the server's disconnected, stop the client:
  if (!client.connected()) {
    Serial.println();
    Serial.println("disconnecting.");
    Serial.println("");
    client.stop();

    delay(4000);
    job(); //call job function
    delay(1000);
    Serial.println("Reconnecting to server");
  // then connect again
    conn();
  }
}

void conn() {
  // print the Ethernet board/shield's IP address:
  Serial.print("My IP address: ");
```

```
63  Serial.println(Ethernet.localIP());
64
65  Serial.println("connecting...");
66  if (client.connect(server, 80)) {
67    Serial.println("connected");
68    // Make a HTTP GET request including the temperature and the notifications' values
       for smoke, motion and door:
69    client.println("GET /connect/submitdata.php?temp=" + String((int)temp) + "&motion=
      " + String(actions[1]) + "&smoke=" + String(actions[0]) + "&door=" + String(
      actions[2]) + " HTTP/1.0");
70    client.println("Host: www.edged.xyz"); // here put your URL
71    client.println("User-Agent: arduino-ethernet");
72    client.println("Connection: close");
73    client.println();
74  } else {
75    // if you didn't get a connection to the server:
76    Serial.println("connection failed");
77  }
78  }
79  // Function job gets transmitted message and calculates the state regarding the
      enabled notifications
80  void job(){
81    int i;
82    uint8_t buf[VW_MAX_MESSAGE_LEN];
83    uint8_t buflen = VW_MAX_MESSAGE_LEN;
84    while (!vw_get_message(buf, &buflen)){ // waits until it receives a message from
      RF transmitter
85    delay(500);}
86    for (i=0;i<buflen;i++){
87     msg[i] = char(buf[i]); // add each received byte to the char array msg
88    }
89    msg[buflen] = '\0'; // append the "terminate string" character
90    int state = atoi(msg); // convert char array to integer
91    Serial.println(state);
92    // Export current notifications from state variable:
93    if (state >= 4){
94      actions[2]=true;
95      Serial.print("Door Opened");
96      state-=4;
97    }else{
98      actions[2]=false;
99    }
100   if (state%2) {
101     actions[0]=true;
102     Serial.print("Gas");
103     state-=1;
104   }else{
105     actions[0]=false;
106   }
107   if (state == 2){
108     actions[1]=true;
109     Serial.print("Motion trigered");
110   }else{
111     actions[1]=false;
112   }
113   //Calculate temperature from voltage reading:
114   int reading = analogRead(tempPin);
115   float voltage=reading*3.3; // Vcc=3.3V
116   voltage/=1024.0;
117   temp = (voltage-0.3)*100;
118 }
```

Σχόλια κώδικα 2 ης συσκευής

Και στη δεύτερη συσκευή θα θέλαμε να τονίσουμε ορισμένα λεπτά σημεία. Καταρχάς, επισημαίνεται ότι για τη σωστή λειτουργία του Ethernet Shield πρέπει να μην χρησιμοποιούνται τα digital pins *10-13* για τη διασύνδεση εξαρτημάτων, καθώς αυτά απαιτούνται για την επικοινωνία μεταξύ του Ethernet Shield και του Arduino. Επίσης, για να ερμηνεύσουμε τα δεδομένα που λαμβάνουμε μέσω της επικοινωνίας RF κάθε byte που λαμβάνεται αποθηκεύεται σε ένα πίνακα από χαρακτήρες, όπου στο τέλος αυτού του πίνακα τοποθετείται ο ειδικός χαρακτήρας τερματισμού string. Έπειτα χρησιμοποιούμε την συνάρτηση *atoi()* (αντίστροφη της συνάρτησης *itoa()*) για να μετατρέψουμε τον πίνακα χαρακτήρων σε ακέραιο. (γραμμές *86-90*)

4 RF Module – Επικοινωνία των δύο Arduino

Το RF module αποτελείται από ένα πομπό (transmitter) που βρίσκεται στην πρώτη συσκευή κι έναν αποδέκτη (receiver) που βρίσκεται στη δεύτερη συσκευή.
Μέσω αυτής της RF επικοινωνίας μεταφέρονται ασύρματα οι μετρήσεις των αισθητήρων της 1ης συσκευής (δηλαδή αισθητήρας καπνού, απόστασης – κίνησης και magnetic door switch) και τα οποία στέλνονται online μέσω του Ethernet Shield. Επίσης, στέλνεται online και η μέτρηση της θερμοκρασίας από την 2η συσκευή.
Αξίζει σε αυτό το σημείο να σημειωθεί ότι παρότι τα τεχνικά χαρακτηριστικά (specifications) του RF module δηλώνουν για τον transmitter: Launch distance: 20-200 meters (different voltage, different results) και Operating voltage: 3.5-12V, ωστόσο δεν ανταποκρίνονται στην πραγματικότητα, καθώς έπειτα από δοκιμές με διάφορες τιμές τάσης η μέγιστη απόσταση λειτουργίας παρέμενε περίπου 2m. Έπειτα από σχετική έρευνα συμπεράναμε ότι αρκετές φορές η επιλογή πάρα πολύ φθηνών εξαρτημάτων εγκυμονεί τον κίνδυνο να μην ανταποκρίνονται στα διαφημιζόμενα χαρακτηριστικά τους.
Για να μεταφέρουμε την πληροφορία μέσω του RF module χρησιμοποιείται μια μορφή κωδικοποίησης. Ειδικότερα, η κωδικοποίηση που χρησιμοποιούμε περιλαμβάνει:

Ενεργοποιημένοι Αισθητήρες	Σήμα που μεταδίδει ο Receiver
Smoke Sensor	1
Ultrasonic Sensor	2
Smoke και Ultrasonic Sensors	3
Magnetic Door Switch Sensor	4
Smoke και Magnetic Door Switch Sensors	5
Ultrasonic και Magnetic Door Switch Sensors	6
Όλα μαζί	7

Για την σωστή λειτουργία του RF module απαιτείται η χρήση της βιβλιοθήκης (Library) VirtualWire. Μπορεί να βρεθεί στο παρακάτω link: `https://www.pjrc.com/teensy/td_libs_VirtualWire.html`

5 Διαδικτυακή πρόσβαση

Δημιουργήσαμε μία ιστοσελίδα με την οποία επικοινωνεί το Ethernet Shield και στην οποία ανεβάζει τα δεδομένα από τους αισθητήρες (θερμοκρασίας, magnetic door switch, καπνού και απόστασης – κίνησης). Τα δεδομένα αυτά μπορεί να τα δει ο χρήστης οπουδήποτε κι αν βρίσκεται, μέσα από το URL: `http://www.edged.xyz/connect/` (γενικά: `http://www.yourdomain.smth/connect/`).

Για τη δημιουργία μιας αντίστοιχης σελίδας μπορείτε να χρησιμοποιήσετε ένα από τα πολλά free hosting services που υπάρχουν στο web, όπως 000webhost.com, hostinger.gr κ.α. Στη συνέχεια, θα πρέπει να ανεβάσει κανείς τα αρχεία index.php και submitdata.php μέσω ftp client ή μέσω του web interface του hosting service. Σημειώνεται ότι το αρχείο submitdata.php θα δημιουργήσει ένα τρίτο αρχείο data.php, στο οποίο αποθηκεύονται οι τιμές των μετρήσεων. Το URL της ιστοσελίδας που θα χρησιμοποιηθεί θα πρέπει να μπει στον κώδικα της δεύτερης συσκευής στο όνομα του server και στο όνομα του Host (τα δύο σημεία υποδεικνύονται από τα σχόλια).

Παρακάτω παρουσιάζονται οι κώδικες των δύο αυτών αρχείων:

submitdata.php

```
<?php
  // Checks if the variables exist and then
  // creates php code in the file data.php that stores the current values of the
    variables
  // but also the last time they were triggered (set to 1)
if(isset($_GET['temp'],$_GET['door'],$_GET['motion'],$_GET['smoke'])) {
  if(file_exists('data.php')){
    include 'data.php';
  } else {
    $last_door = 0;
    $last_motion = 0;
    $last_smoke = 0;
  }
  $temp = $_GET['temp'];
  $door = $_GET['door'];
  $smoke = $_GET['smoke'];
  $motion = $_GET['motion'];
  $filename = 'data.php';
  date_default_timezone_set('Europe/Athens');
  $now = date('l jS \of F Y h:i:s A');
  if ($door)
    $last_door = $now;
  if ($smoke)
    $last_smoke = $now;
  if ($motion)
    $last_motion = $now;
  $var = "<?php\n" . '$time = ' . "'" . $now . "'" . ';$temp = ' . $temp . ';$door = '
     . $door . ';$smoke = ' . $smoke . ';$motion = ' . $motion;
  $var = $var  . ';$last_door = \'' . $last_door . '\';$last_smoke=\'' . $last_smoke .
     '\';$last_motion=\'' . $last_motion . "';\n?>";
  file_put_contents($filename, $var);
  echo 'Data Submited';
  return 0;
}else {
  echo 'Error';
  }
?>
```

index.php

```php
<?php
// Imports the data from data.php and prints them
include('data.php');
echo $time . "\r\n";
echo '<br>Temperature: ' . $temp . "\r\n";
echo '<br>Door: ' . $door . "\r\n";
echo '<br>Smoke: ' . $smoke . "\r\n";
echo '<br>Motion: ' . $motion . "\r\n";
echo '<br>Last_Door: ' . $last_door . "\r\n";
echo '<br>Last_Smoke: ' . $last_smoke . "\r\n";
echo '<br>Last_Motion: ' . $last_motion . "\r\n";
?>
```

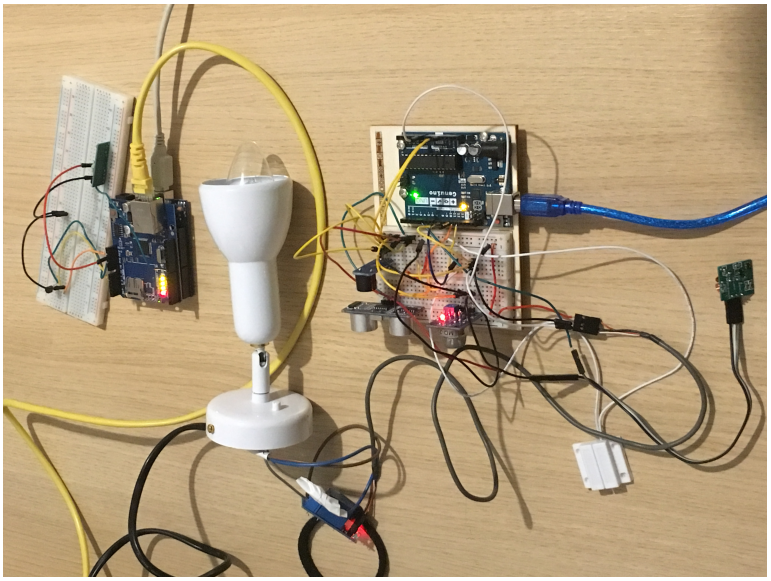
6 Παρουσίαση Τελικού αποτελέσματος

Παρακάτω παρουσιάζονται τα τελικά αποτελέσματα της εργασίας μας.

Wednesday 18th of January 2017 11:00:20 PM
Temperature: 19
Door: 0
Smoke: 0
Motion: 0
Last_Door: Wednesday 18th of January 2017 10:58:17 PM
Last_Smoke: Wednesday 18th of January 2017 11:00:09 PM
Last_Motion: Wednesday 18th of January 2017 10:59:24 PM

Εδώ φαίνεται η καταγραφή των μετρήσεων στην ιστοσελίδας μας. Πιο συγκεκριμένα, στην πρώτη γραμμή φαίνεται η τελευταία ενημέρωση. Στη συνέχεια παρουσιάζεται η τρέχουσα κατάσταση και πότε είχαμε την τελευταία ειδοποίηση.

Εδώ φαίνεται ο σχεδιασμός του συστήματος ασφαλείας σε φωτογραφία.



Σχήμα 3: Τελικό κύκλωμα

Παρακάτω παρουσιάζεται και το link του github για το project:
https://github.com/dbarmperis/Digital-Systems-Interfaces.git

Αναφορές

[1] Markus Ulfberg *Arduino: Sending integers over RF with VirtualWire*. http://genericnerd.blogspot.gr/2012/07/arduino-sending-integers-over-rf-with.html.

[2] Aritro Mukherjee. *Smoke Detection using MQ-2 Gas Sensor*. https://create.arduino.cc/projecthub/Aritro/smoke-detection-using-mq-2-gas-sensor-79c54a.

[3] lady ada. *TMP36 Temperature Sensor*. https://learn.adafruit.com/tmp36-temperature-sensor/using-a-temp-sensor.

[4] arduino.cc. *Web Client Repeating*. https://www.arduino.cc/en/Tutorial/WebClientRepeating.