

Προαιρετική Εργασία Ενσωματωμένων Συστημάτων

Κωνσταντίνος Παρασκευόπουλος

¹ Computer Engineering & Informatics Department, Πανεπιστήμιο Πατρών
st1072608@ceid.upatras.gr

Abstract. Εργασία σχετικά με τον σχεδιασμό ενσωματωμένου συστήματος που έχει ως βασική λειτουργία την προστασία κάποιου οικήματος από φωτιά, διαρροή νερού και διαρροή επικίνδυνων αερίων

Keywords: Ενσωματωμένα Συστήματα, Αισθητήρες, Μικροελεγκτής, Προστασία σπιτιού, Προστασία, Φωτιά, Διαρροή.

1 Δήλωση εργασίας και ομάδας

Κωνσταντίνος Παρασκευόπουλος AM 1072608, 5^ο Έτος Σπουδών

2 Περιγραφή εφαρμογής-προβλήματος

Στις μέρες μας ένα από τα σημαντικότερα προβλήματα που εμφανίζονται στην καθημερινότητα είναι η αδυναμία προστασίας των οικιών από διάφορες ενδογενείς καταστροφές, όπως κάποια διαρροή νερού εξαιτίας αστοχίας των σωληνώσεων, κάποια διαρροή επικίνδυνων αερίων, πολλές φορές εξαιτίας κάποιου κακού χειρισμού, η ακόμα και κάποια πυρκαγιά. Οι επιπτώσεις αυτών των καταστροφών μπορεί να είναι καταστροφικές ακόμη και αν υπάρχει κάποιος άνθρωπος που θα τις εντοπίσει και θα αντιδράσει εγκαίρως, πόσο μάλλον όταν δεν υπάρχει κανένας εντός της οικίας, όπου οι καταστροφές θα έχουν ακόμα μεγαλύτερο αντίκτυπο. Σκοπός του συστήματος το οποίο θα σχεδιάσουμε είναι να αυτοματοποιεί τον έλεγχο, και την καταστολή τέτοιων καταστροφών με ταχύτητα σημαντικά μεγαλύτερη από την μέση ταχύτητα αντίδρασης ενός ανθρώπου και επίσης να ειδοποιεί τον χρήστη για την ενεργοποίησή του ώστε αυτός να λάβει τα απαραίτητα μέτρα στη συνέχεια.

3 Απαιτήσεις χρήστη, λειτουργιών

Οι κύριες απαιτήσεις του συστήματος είναι η γρήγορη και έγκαιρη απόκριση καθώς και το μικρό κόστος, αφού τέτοια συστήματα υπάρχουν ήδη στην αγορά, ωστόσο έχουν πολύ υψηλό κόστος με συνέπεια να μην απευθύνονται στον μέσο καταναλωτή. Σχετικά με την απόκριση, θα πρέπει αυτή να είναι σχετικά γρήγορη, υπολογίζοντας όμως και το γεγονός ότι όσο αυξάνεται ο ρυθμός δειγματοληψίας, αυξάνεται και η πιθανότητα για false-positive αποτελέσματα (π.χ. δεν θέλουμε το σύστημα να μπερδεύει το άναμμα ενός αναπτήρα για 2-3 δευτερόλεπτα για πυρκαγιά και να ενεργοποιεί το σύστημα πυρόσβεσης). Επίσης η απόκριση θα πρέπει να είναι όσο το δυνατόν πιο έγκαιρή και βεβαιασμένη περισσότερο προς false-positive αποτελέσματα, καθώς είναι προτιμότερη μία άστοχη ενεργοποίηση του συναγερμού εξαιτίας κάποιου ακραίου εξωτερικού παράγοντα παρά την μη ενεργοποίησή του ενώ συντρέχει κάποιος κίνδυνος, χωρίς αυτό να συμβαίνει ότι κάτι τέτοιο είναι αποδεκτό καθώς αυτό μειώνει την αξιοπιστία της συσκευής. Το σύστημα που θα σχεδιάσουμε θα πρέπει ακόμη να έχει την δυνατότητα να ενημερώνει τον χρήστη μέσω SMS για την ενεργοποίηση του συναγερμού. Τέλος η συσκευή θα πρέπει να έχει back-up τροφοδοσία μέσω μπαταρίας η οποία θα εξασφαλίζει την λειτουργία της για κάποιο χρονικό διάστημα ακόμη και στην περίπτωση διακοπής ρεύματος.

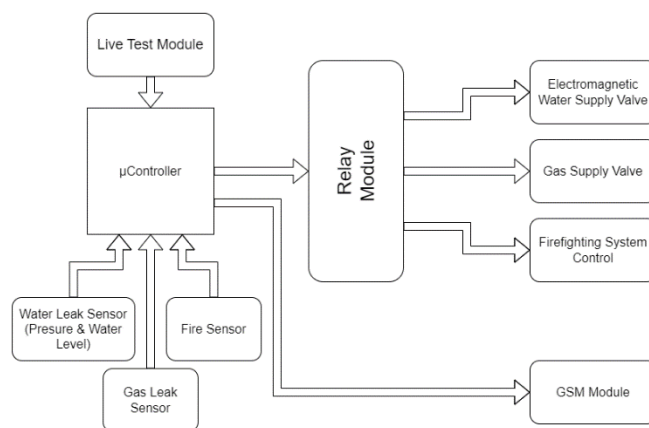
4 Προδιαγραφές χρήστη, λειτουργιών

- Δειγματοληψία συσκευής: ανά 10 sec
- Κόστος Παραγωγής: $\leq 100\text{€}$
- Ακρίβεια αισθητήρων: 0-2 δεκαδικά ψηφία
- Αυτονομία: minimum 5 ώρες
- Πρωτόκολλο επικοινωνίας με χρήστη: SMS (μέσω GSM Module, καθώς η λειτουργία του παραπάνω module δεν εξαρτάται από την παροχή ή όχι internet η οποία κατά την διάρκεια μιας πυρκαγιάς για παράδειγμα μπορεί να μην είναι δυνατή)

- Θεωρώ ότι το σύστημα πυρόσβεσης εγκαθίσταται ξεχωριστά από τον χρήστη, όπως και όλοι οι αισθητήρες και ηλεκτρο-βαλβίδες που συνοδεύουν το σύστημα
- Είσοδοι για έλεγχο ορθής λειτουργίας του συστήματος

5 Προσέγγιση αρχιτεκτονικής λογισμικού

Παρακάτω παραθέτω το διάγραμμα της προσέγγισης της αρχιτεκτονικής του υλικού του συστήματος. Με λίγα λόγια μπορεί κανείς να πει πως το σύστημα βασίζεται σε έναν μικροελεγκτή ο οποίος δειγματοληπτεί μια ομάδα αισθητήρων και ανάλογα με τα αποτελέσματα ελέγχει ένα σετ από Relay τα οποία είναι με τη σειρά τους υπεύθυνα για την λειτουργία των μηχανισμών αντιμετώπισης των αντίστοιχων καταστροφών. Τέλος υπάρχει ένα module το οποίο είναι υπεύθυνο για την δοκιμή-έλεγχο ορθής λειτουργίας του συστήματος και θα αποτελείται κατά πάσα πιθανότητα από κάποιο switch το οποίο θα πραγματοποιεί έλεγχο μέσω του software



6 Προσέγγιση αρχιτεκτονικής υλικού

Το λογισμικό που θα εκτελεί ο μικροελεγκτής θα πρέπει να δειγματοληπτεί τις τιμές των αισθητήρων και να ελέγχει εάν βρίσκονται σε λογικά όρια(-τιμές για αισθητήρες που επιστρέφουν ψηφιακές τιμές 0 ή 1), σε αντίθετη περίπτωση θα πρέπει να οδηγεί τις εξόδους του αντίστοιχα στις τιμές που θα ενεργοποιούν τα Relay τα οποία με τη σειρά τους θα ελέγχουν τους μηχανισμούς αντιμετώπισης. Επίσης κατά την ενεργοποίηση της εισόδου ελέγχου ορθής λειτουργίας του κυκλώματος θα πρέπει να προσομοιάζεται η ενεργοποίηση κάθε ενός από τους αισθητήρες χωρίς όμως να ενεργοποιείται ο μηχανισμός αντιμετώπισης, καθώς αυτό θα ήταν μη αναγκαία χρήση διαθέσιμων πόρων. Τέλος το Software του συστήματος είναι υπεύθυνο για την σύνταξη και αποστολή μηνύματος SMS στον χρήστη που θα τον ειδοποιεί για οποιαδήποτε ενεργοποίηση κάποιου αισθητήρα (θα πρέπει να στέλνει SMS και κατά την διάρκεια του testing)

7 Αρχιτεκτονική Υλικού

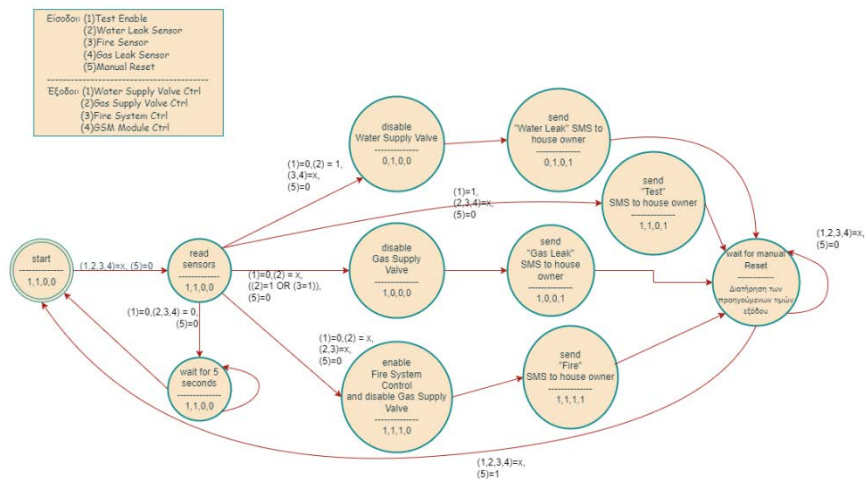
Για το υλικό του συστήματος επιλέχθηκε ο microcontroller *AT-Mega328*, ο οποίος ανήκει στην οικογένεια των 8-Bit μικροελεγκτών της Atmel και είναι η βάση του *Arduino Uno* γύρω από το οποίο έχει σχεδιαστεί συνολικά το σύστημα. Ο κυριότερος λόγος που επιλέχθηκε η συγκεκριμένη πλατφόρμα ανάπτυξης υλικού είναι το χαμηλό της κόστος καθώς κάθε μονάδα κοστίζει 27,60 \$ ενώ καλύπτει τις ανάγκες του συστήματος σε εξαιρετικά ικανοποιητικό επίπεδο. Επίσης επιλέχθηκε η version της πλατφόρμας με το αφαιρούμενο chip δίνοντας έτσι τη δυνατότητα αλλαγής μόνο του chip και όχι ολόκληρης της πλακέτας σε περίπτωση καταστροφής του αρχικού chip. Με αυτό τον τρόπο εξασφαλίζεται το χαμηλό κόστος επισκευής και υποστήριξης του συστήματος, καθώς το ανταλλακτικό chip κοστίζει μόλις 6,50 \$ (Μόλις 25% της τιμής αντικατάστασης ολόκληρης της πλακέτας). Επιπλέον ακόμα ένα προτέρημα της παραπάνω πλατφόρμας είναι συνοδεύεται από την open-source «σουίτα» προγραμματισμού του Arduino η οποία διευκολύνει αρκετά τον προγραμματισμό της συσκευής. Τέλος το ίδιο το Arduino UNO είναι open-source project συνεπώς μπορεί να μειωθεί και το ίδιο το κόστος της πλακέτας

διότι μπορεί να σχεδιαστεί αρκετά εύκολα κάποιο είδος «κλώνου» με αρκετά χαμηλότερο κόστος. Για το σύστημά μας χρησιμοποιούνται 2 από τα Digital pins για το διάβασμα των τιμών των αισθητήρων 3 Digital Pins για τον έλεγχο των Relays ελέγχου των υποσυστημάτων προστασίας, 1 Analog Pin για τον έλεγχο της στάθμης νερού και τέλος 2 pins για την αποστολή και λήψη δεδομένων ανάμεσα στο Arduino και το GSM Module το οποίο είναι υπεύθυνο για την αποστολή SMS στον χρήστη όταν το σύστημα εντοπίσει κάποια «καταστροφή». Για GSM Module χρησιμοποιήθηκε το *SIM900 GSM Shield* από το οποίο χρησιμοποιείται η δυνατότητα αποστολής SMS μέσω δικτύου GSM με τη χρήση κάρτας SIM 2G. Τέλος έχει σχεδιαστεί σύστημα διαχείρισης τροφοδοσίας με σκοπό την συνύπαρξη τροφοδοσίας από ηλεκτρική πρίζα και τροφοδοσίας από μπαταρία για την εξασφάλιση αυτονομίας του συστήματος σε περίπτωση διακοπής ρεύματος.

8 Αρχιτεκτονική Λογισμικού

Το λογισμικό είναι υπεύθυνο στο σύστημα για την ανάγνωση των τιμών των αισθητήρων (έχει προβλεφθεί η επιλογή όσο το δυνατόν περισσότερων αισθητήρων με ψηφιακές εξόδους, μονάχα ο αισθητήρας στάθμης νερού έχει αναλογική έξοδο για το διάβασμα της οποίας χρησιμοποιείται 1 εκ των 6 αναλογικών pin του Arduino). Επίσης μέσω λογισμικού γίνεται ο έλεγχος των 12V Relays τα οποία είναι υπεύθυνα για τον έλεγχο των βαλβίδων παροχής νερού και αερίου και της αντλίας νερού για πυρόσβεση. Η δειγματοληψία των αισθητήρων γίνεται ανά 10 δευτερόλεπτα. Τέλος το Software του συστήματος είναι υπεύθυνο για τον έλεγχο του GSM Module και την αποστολή SMS όταν αυτό κριθεί απαραίτητο. Για τον έλεγχο του GSM Module χρησιμοποιείται η βιβλιοθήκη SoftwareSerial.h. Παρακάτω παρατίθεται ένα FSM το οποίο εξηγεί την λειτουργία του Software του συστήματος. Για την απλοποίηση του FSM παραλείφθηκαν οι περιπτώσεις ενεργοποίησης παραπάνω από ενός αισθητήρα, ωστόσο αυτό προβλέπεται στο τελικό Software εκτελώντας σειριακή ανάγνωση των τιμών των αισθητήρων και στη συνέχεια εκτελώντας τις αντίστοιχες ενέργειες. Σε περίπτωση ενεργοποίησης ενός αισθητήρα το σύστημα μπαίνει σε mode αντιμετώπισης κινδύνου και δεν

βγαίνει από αυτό μέχρι να γίνει manual Reset του συστήματος (μέσω του κουμπιού Reset της πλακέτας του Arduino UNO)



Ο Κώδικας του συστήματος είναι:

```

/*****Libraries*****/
#include <SoftwareSerial.h> //Library for the GSM Module
#include <Wire.h>

/*****Const Pin Variables*****/

const int RELAY_ENABLE_FIRE = 10; //Relay Digital Pin
const int RELAY_ENABLE_WATER = 11; //Relay Digital Pin
const int RELAY_ENABLE_GAS = 12; //Relay Digital Pin
const int WATER_SENSOR = A5; //Water sensor Analog Pin
const int TEST_ENABLE = 3; //Test Enable Pin
const int FIRE_SENSOR_D = 4; //IR Sensor Digital Pin
//const int FIRE_SENSOR_A = A0; //IR Sensor Analog Pin
const int GAS_SENSOR_D = 5; //Gas Sensor Digital
//const int GAS_SENSOR_A = A1; //Gas Sensor Analog

/*****Flags*****/

```

```

bool SMS_SENT_FIRE = 0;
bool SMS_SENT_WATER = 0;
bool SMS_SENT_GAS = 0;

/*****Constructors*****/

//Create software serial object to communicate with SIM900
SoftwareSerial mySerial(7, 8); //SIM900 Tx & Rx is connected to Digital Pins #7 & #8
//Adafruit_VEML6075 uv = Adafruit_VEML6075(); //Call Constructor for the UV Sensor

/*****Global Vars*****/

int FS = 0; //Init Fire Detect Variable
int WLS = 0; //Init Water Detect Variable
int GLS = 0; //Init Gas Detect Variable
int TEST_EN;

/*****Main Program*****/
void setup() {

    /*PinMode Declarations*/
    pinMode (FIRE_SENSOR_D, INPUT);
    pinMode (GAS_SENSOR_D, INPUT);
    pinMode (TEST_ENABLE, INPUT);

    digitalWrite(RELAY_ENABLE_WATER, LOW); //Initial States of Valves
    digitalWrite(RELAY_ENABLE_FIRE, LOW);
    digitalWrite(RELAY_ENABLE_GAS, LOW);

    FS = 0; //Init Fire Detect Variable
    WLS = 0; //Init Water Detect Variable
    GLS = 0; //Init Gas Detect Variable

    Serial.begin(9600);

```

```

//Begin serial communication with Arduino and SIM900
mySerial.begin(9600);
Serial.println("Initializing...");
delay(1000);

}

void loop() {

    TEST_EN = digitalRead(TEST_ENABLE);
    FS = digitalRead(FIRE_SENSOR_D);
    GLS = digitalRead(GAS_SENSOR_D);
    if(analogRead(WATER_SENSOR)>100) WLS=HIGH;
    else WLS=LOW;

    if(TEST_EN){ //If Test Enable is HIGH then perform System Check Routine
        System_Check_Routine();
    }

    if(FS == HIGH){ //Check for Fire
        if(!SMS_SENT_FIRE){
            if(!TEST_EN){
                open_Valve_Fire(); //Enable Fire Extinguishing Pump
                close_Valve_Gas(); //Close Flammable Gases Valve
            }
            SMS_fire(); //Send Fire Detection SMS
            SMS_SENT_FIRE = true;//Enable Flag that assures that all the actions have been completed to avoid being done again
        }
    }

    if(WLS == HIGH){ //Check for Water Leak
        if(!SMS_SENT_WATER){
            if(!TEST_EN){
                close_Valve_Water(); //Enable Fire Extinguishing Pump
            }
            SMS_water_leak();
        }
    }
}

```



```

    SMS_SENT_WATER = true;
  }
}

if(GLS == HIGH){ //Check for Gas Leak
  if(!SMS_SENT_GAS){
    if(!TEST_EN){
      close_Valve_Gas(); //Enable Fire Extinguishing Pump
    }
    SMS_gas_leak();
    SMS_SENT_GAS = true;
  }
}

delay(10000); //Wait 10 seconds
}

void updateSerial()
{
  delay(500);
  while (Serial.available())
  {
    mySerial.write(Serial.read()); //Forward what Serial received to Software Serial Port
  }
  while(mySerial.available())
  {
    Serial.write(mySerial.read()); //Forward what Software Serial received to Serial Port
  }
}

void SMS_fire() {

  mySerial.println("AT"); //Handshaking with SIM900
  updateSerial();

  mySerial.println("AT+CMGF=1"); // Configuring TEXT mode
  updateSerial();

  mySerial.println("AT+CMGS=\"+306941413370\""); //change ZZ with country code and
  xxxxxxxxxxxx with phone number to sms

```

```

    updateSerial();
    mySerial.print("Fire Detected"); //text content
    updateSerial();
    mySerial.write(26); /*Sends SMS to the specified phone number. After this AT command
any text message followed by 'Ctrl+z' character is treated as SMS. 'Ctrl+z' is actually a
26th non-printing character that is described as 'substitute' in the ASCII table. There-
fore, we need to send 26 (0x1A) at the end of the command.*/
}

void SMS_water_leak() {
    mySerial.println("AT"); //Handshaking with SIM900
    updateSerial();

    mySerial.println("AT+CMGF=1"); // Configuring TEXT mode
    updateSerial();
    mySerial.println("AT+CMGS=\"+306941413370\"");//change ZZ with country code and
xxxxxxxxxxxx with phone number to sms
    updateSerial();
    mySerial.print("Water Leak Detected"); //text content
    updateSerial();
    mySerial.write(26); /*Sends SMS to the specified phone number. After this AT command
any text message followed by 'Ctrl+z' character is treated as SMS. 'Ctrl+z' is actually a
26th non-printing character that is described as 'substitute' in the ASCII table. There-
fore, we need to send 26 (0x1A) at the end of the command.*/
}

void SMS_gas_leak() {
    mySerial.println("AT"); //Handshaking with SIM900
    updateSerial();

    mySerial.println("AT+CMGF=1"); // Configuring TEXT mode
    updateSerial();
    mySerial.println("AT+CMGS=\"+306941413370\"");//change ZZ with country code and
xxxxxxxxxxxx with phone number to sms
    updateSerial();
    mySerial.print("Gas Leak Detected"); //text content
    updateSerial();

```

```

    mySerial.write(26); /*Sends SMS to the specified phone number. After this AT command
any text message followed by 'Ctrl+z' character is treated as SMS. 'Ctrl+z' is actually a
26th non-printing character that is described as 'substitute' in the ASCII table. There-
fore, we need to send 26 (0x1A) at the end of the command.*/
}

void close_Valve_Water(){
    digitalWrite(RELAY_ENABLE_WATER, HIGH);
}

void open_Valve_Fire(){
    digitalWrite(RELAY_ENABLE_FIRE, HIGH);
}

void close_Valve_Gas(){
    digitalWrite(RELAY_ENABLE_GAS, HIGH);
}

void System_Check_Routine(){
    FS = HIGH; //Simulate Fire Alarm
    WLS = HIGH; //Simulate Water Leak Alarm
    GLS = HIGH; //Simulate Gas Leak Alarm

    int check = FS & WLS & GLS; //All Values must be 1 to pass the test / This also tests
    ALU for faults in the AND Operation

    mySerial.println("AT"); //Handshaking with SIM900
    updateSerial();

    mySerial.println("AT+CMGF=1"); // Configuring TEXT mode
    updateSerial();
    mySerial.println("AT+CMGS=\"+306941413370\""); //change ZZ with country code and
    xxxxxxxxxxx with phone number to sms
    updateSerial();
    if(check){
        mySerial.print("Check Routine Initialization...\n\nTest Passed!"); //Test text con-
        tent for Test Success
    }
}

```

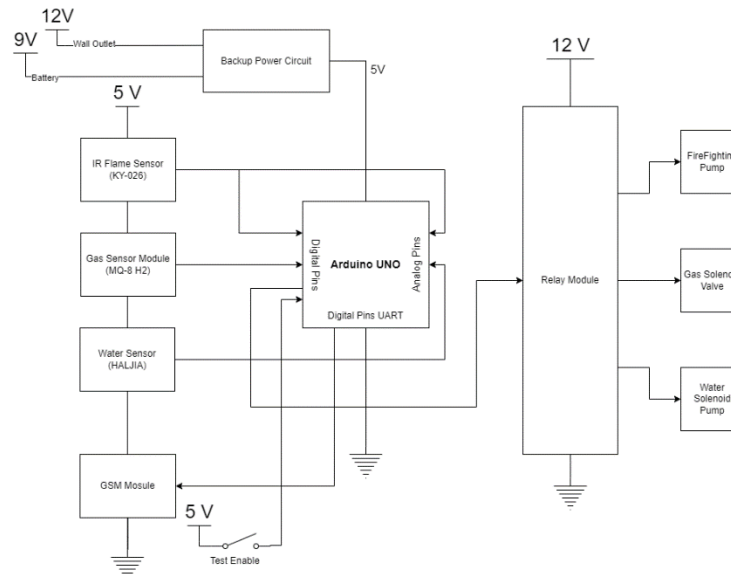
```

}
else {
    mySerial.print("\n\n-----\n\n");
    mySerial.print("Check Routine Initialization...\n\nTest Failed!"); //Test text con-
tent for Test Fail
    mySerial.print("\nFire: ");
    mySerial.print(FS);
    mySerial.print("\nGas Leak: ");
    mySerial.print(GLS);
    mySerial.print("\nWater Leak: ");
    mySerial.print(WLS);
    mySerial.print("\n\n-----\n\n");
}
updateSerial();
mySerial.write(26);
}

```

9 Σχεδιασμός Συνολικού Συστήματος

Το συνολικό σύστημα από άποψη υλικού φαίνεται στο σχήμα παρακάτω.



Για το σύστημα από άποψη λογισμικού παρατίθεται παραπάνω το FSM στο οποίο βασίστηκε ο σχεδιασμός του Software.

10 Σχεδιασμός Υποσυστημάτων

Το σύστημα μπορεί θεωρητικά να χωριστεί σε 5 υποσυστήματα τα οποία θα αναλύσω παρακάτω.

10.1 Υποσύστημα Προστασίας από φωτιά

Για το υποσύστημα προστασίας από φωτιά χρησιμοποιήθηκε ο αισθητήρας υπερύθρων KY-026. Ο αισθητήρας αυτός επιλέχθηκε εξαιτίας του χαμηλού του κόστους καθώς και της δυνατότητας να παρέχει ψηφιακή έξοδο σε περίπτωση ανίχνευσης φωτιάς. Η ευαισθησία του αισθητήρα μπορεί να ρυθμιστεί μέσω ροοστάτη και ρυθμίζεται ανάμεσα στα 1 έως 4 μέτρα με βάση το αντίστοιχο datasheet. Σε περίπτωση ανίχνευσης πηγής φωτιάς ενεργοποιείται μια αντλία 12V χαμηλής ροής-υψηλής πίεσης η οποία ελέγχεται από το αντίστοιχο 12V Relay και συνδέεται με νεπόζιτο νερού ξεχωριστό από την κύρια παροχή νερού της οικίας. Η αντλία μετά την ενεργοποίησή της δε σταματά να λειτουργεί μέχρι να γίνει manual Reset στο σύστημα

10.2 Υποσύστημα προστασίας από διαρροή νερού

Για το υποσύστημα προστασίας από διαρροή νερού χρησιμοποιήθηκε ο αισθητήρας στάθμης νερού HALJIA. Ο αισθητήρας αυτός επιλέχθηκε εξαιτίας του χαμηλού του κόστους, ωστόσο οι περισσότεροι αισθητήρες τέτοιου τύπου έχουν πάρα πολλές ομοιότητες και σχετικά χαμηλό κόστος. Ο αισθητήρας μπορεί να λειτουργεί σε επίπεδα υγρασίας 10%-90% σύμφωνα με το αντίστοιχο Datasheet. Επίσης παράγει αναλογική έξοδο ανάλογα με το επίπεδο της στάθμης νερού. Για τον εντοπισμό της διαρροής νερού ενεργοποιείται το αντίστοιχο σήμα κινδύνου στον μικροελεγκτή όταν αναγνωστεί η μικρότερη δυνατή τιμή στάθμης νερού από τον αισθητήρα. Όταν ενεργοποιηθεί το σήμα εντοπισμού διαρροής νερού κλείνει μια ηλεκτρομαγνητική βαλβίδα νερού αποκόπτοντας έτσι την κεντρική παροχή νερού (Solenoid Valve ST-DA 1/4" brass EPDM 0-13bar 12V DC). Η βαλβίδα ελέγχεται και αυτή από ένα 12V Relay και αφού απενεργοποιηθεί (κλείσει) δεν ενεργοποιείται (ανοίγει) μέχρι να γίνει manual Reset στο σύστημα από το χρήστη. Η βαλβίδα είναι normally closed συνεπώς κατά την ενεργοποίηση του σήματος διαρροής το αντίστοιχο σήμα ελέγχου του Relay θα πρέπει να έχει την τιμή 0.

10.3 Υποσύστημα προστασίας από διαρροή αερίου

Για το σύστημα προστασίας από διαρροή αερίου χρησιμοποιήθηκε ο αισθητήρας αερίων που περιέχουν Υδρογόνο (εύφλεκτα αέρια) MQ-8 H2. Ο αισθητήρας αυτός επιλέχθηκε λόγω του χαμηλού του κόστους και της δυνατότητας του για ψηφιακή έξοδο. Ο χρόνος εντοπισμού διαρροής αερίου υπό σωστές συνθήκες και σωστή εγκατάσταση είναι μικρότερος των 20s από την έναρξη της διαρροής. Αντίστοιχα με το σύστημα προστασίας από διαρροή νερού χρησιμοποιείται βαλβίδα η οποία μπλοκάρει την κεντρική παροχή αερίου στην οικία (Solenoid Valve ST-DA 1/4" brass EPDM 0-13bar 12V DC)

10.4 Υποσύστημα επικοινωνίας με το χρήστη μέσω SMS

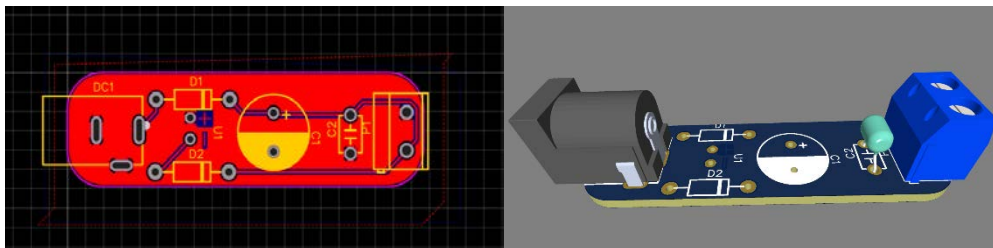
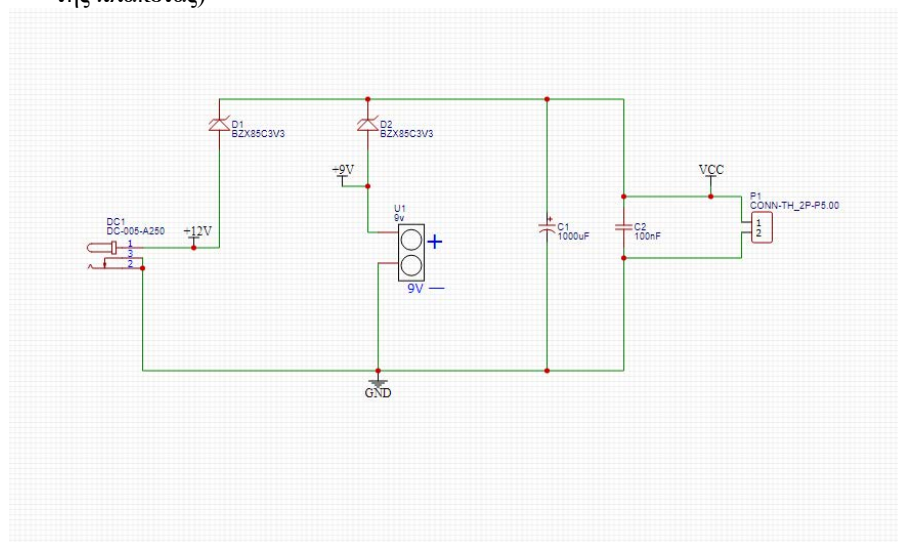
Για την επικοινωνία με το χρήστη μέσω SMS χρησιμοποιήθηκε το SIM900 GSM Shield το οποίο συνδέεται με το Arduino μέσω των Software UART pins της πλακέτας (pins 7 και 8). Για τον έλεγχο του Module χρησιμοποιούνται εντολές AT όπως οι:

- AT: Υπεύθυνη για τη χειραψία με το chip SIM900
- AT+CMGF=1: Διαμόρφωση TEXT mode
- AT+CMGS="\r\nZZxxxxxxxxxx\r\n": Επιλογή αριθμού τηλεφώνου χρήστη

10.5 Υποσύστημα διαχείρισης τροφοδοσίας

Για τον σχεδιασμό του συστήματος διαχείρισης της τροφοδοσίας χρησιμοποιήθηκε το πρόγραμμα σχεδιασμού PCB EasyEDA και σχεδιάστηκε πλακέτα η

οποία έχει τη δυνατότητα να υποστηρίξει τροφοδοσία 12V από ηλεκτρική πρίζα και τροφοδοσία επίσης από μπαταρία 9V. Σχέδια της πλακέτας παρατίθενται παρακάτω (1. Σχηματικό 2. Σχέδιο PCB 3. Τρισδιάστατη Αναπαράσταση της πλακέτας)



11 Σχεδιασμός Συστατικών Συστήματος/Υποσυστημάτων

Για την επιλογή των κατάλληλων Components σχεδιάστηκαν οι παρακάτω πίνακες οι οποίοι χρησιμοποιήθηκαν για τη σύγκριση μεταξύ των ευρημάτων.

Board	Κόστος	Processor	Clock Speed	Flash Memory	RAM	Analog Inputs	GPIO	Form Factor (in.)
Arduino Uno - R3	\$27,60	ATmega328P	16MHz	32KB	2KB	6	20	2.7 x 2.1
Arduino Uno - R3 SMD	\$26,30	ATmega328P	16MHz	32KB	2KB	6	20	2.7 x 2.1
Arduino Pro Mini - 3V/8MHz	\$10,95	ATmega328P	8MHz	32KB	2KB	8	22	1.3 x 1.7
Arduino Mega 2560 R3	\$48,40	ATmega2560	16MHz	256KB	8KB	16	54	4.0 x 2.1
Arduino Nano	\$23,45	ATmega328P	16MHz	32KB	2KB	6	20	0.7 x 1.8

Εικόνα 1: *μControllers Comparison*

Type of Sensor	Κόστος	Name	Responsivity Time (ms)	Sensitivity Distance	Comments
Ultraviolet (UV) Flame Detectors	\$5,95	VEML6075	50	~15 m	Good for Houses with Big Rooms, but in general it is a bit Overkill
Infrared (IR) Flame Detectors	\$3,99	KY-028	3000	1-3 m	Creates Some False Alarms + Great for Home Use, Cheap
Ultraviolet/Infrared (UV/IR) Flame Detectors	\$9,94	VEML6075 + KY-026	min 50, max 1000	1-35 m	Good for Houses with Big Rooms and Small Spaces, definitely Overkill

Εικόνα 2: *Fire Sensors Comparison*

Type of Sensor	Κόστος	Name	Responsivity Time (s)	Comments
Gas Sensor Detection Module	\$6.47	SP13	<20	Detects Propane
Gas Sensor Module	\$4.11	MQ-8 H ₂	<20	Detects Hydrogen-containing Gas (Combustible Gases)
Gas Sensor Module	\$12.99	MQ-9 CO	<20	Detects Flammable CO

Εικόνα 3: Gas Leak Sensors Comparison

Type of Sensor	Κόστος	Name	Water Detection Area	Operating Humidity	Comments
Water Leak (or Level) Sensor	\$4.99	HAL13A	80mm X 16mm (640mm ²)	10%-90%	Great Overall

Εικόνα 4: Water Leak Sensors Comparison

12 Θεωρητική Προσέγγιση Υλοποίησης

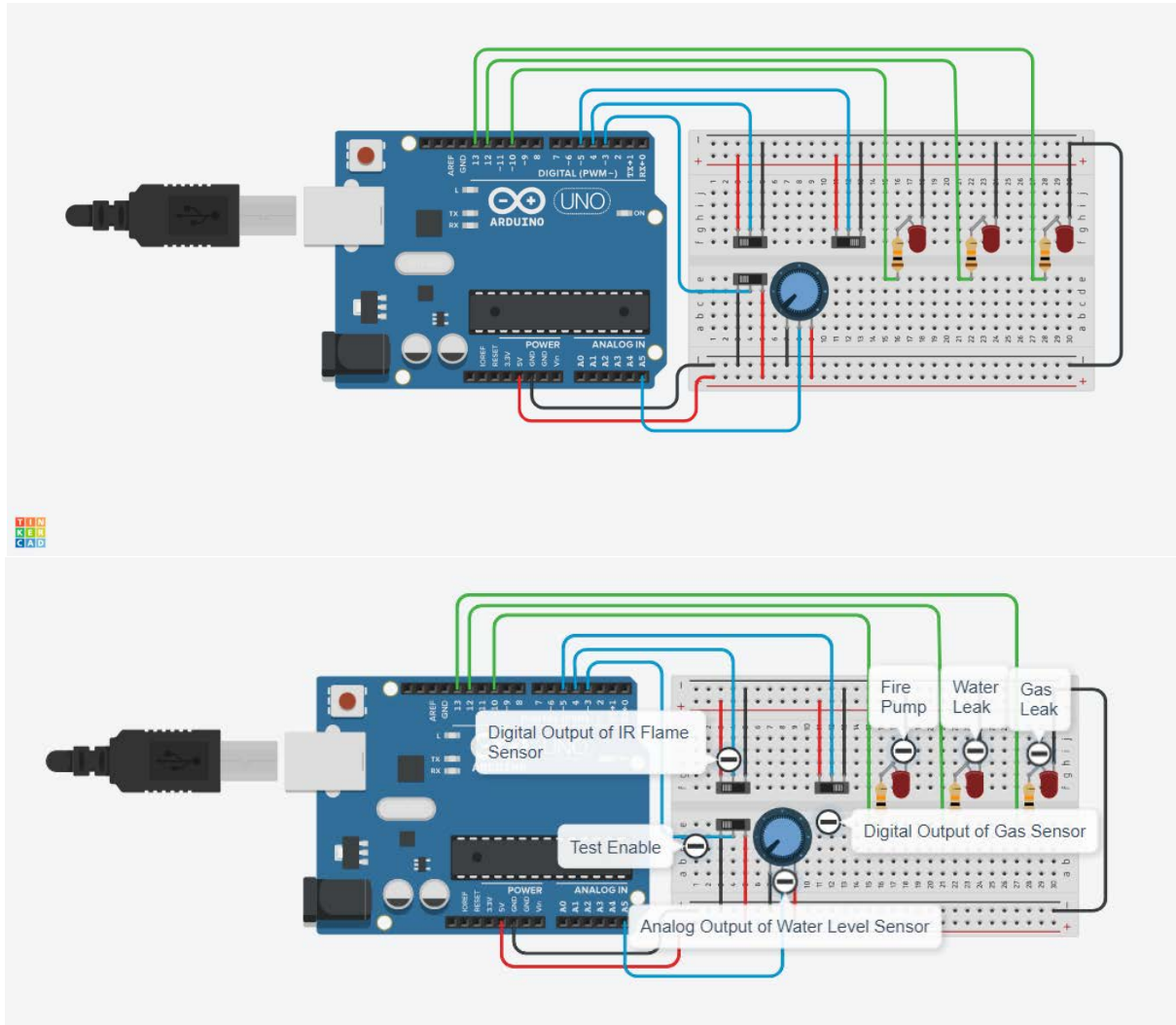
Για την θεωρητική προσέγγιση της υλοποίησης χρησιμοποιήθηκε η online πλατφόρμα σχεδίασης και εξομοίωσης κυκλωμάτων TinkerCAD της εταιρίας AUTODESK. Η πλατφόρμα αυτή δίνει την δυνατότητα για εξομοίωση κυκλωμάτων καθώς και την εκτέλεση κώδικα γραμμένου για το Arduino UNO. Η συγκεκριμένη πλατφόρμα χρησιμοποιείται κυρίως για εκπαιδευτικούς λόγους, συνεπώς δεν εκτελεί εντελώς πιστή εξομοίωση των κυκλωμάτων, ωστόσο η ακρίβεια της εξομοίωσης είναι εντός των αποδεκτών ορίων για το σύστημα που υλοποιούμε. Εξαιτίας των ελλείψεων της εφαρμογής σε Components, διάφορα στοιχεία του κυκλώματος αντικαταστάθηκαν με άλλα τα οποία έχουν την ίδια λογική συμπεριφορά. Συγκεκριμένα:

- Οι ψηφιακές έξοδοι των αισθητήρων (είσοδοι στο Arduino) αντικαταστάθηκαν με Switches (Διακόπτες) οι οποίοι έχουν τιμές 0 ή 1
- Οι αναλογική έξοδος (είσοδος στο Arduino) του αισθητήρα επιφάνειας νερού αντικαταστάθηκε από έναν Ποσοστάτη ο οποίος δίνει στην αναλογική είσοδο του Arduino τιμές από 0V έως 5V όπως θα λειτουργούσε δηλαδή και ο αντίστοιχος πραγματικός αισθητήρας.
- Οι ψηφιακές έξοδοι του Arduino που θα συνδέονταν με Relay τα οποία με τη σειρά τους θα είχαν τον έλεγχο των αντίστοιχων Βαλβίδων και της Αντλίας Νερού συνδέονται πλέον σε LEDs τα οποία επιδεικνύουν την ενεργοποίηση ή όχι των αντίστοιχων Relays που θα υπήρχαν στην πραγματική υλοποίηση.
- Τέλος εξαιτίας της έλλειψης αντιστοιχίας για το SIM900 GSM Shield που χρησιμοποιείται για την αποστολή SMS χρησιμοποιείται το Serial Interface του Arduino που μας επιτρέπει να διαβάζουμε τα μηνύματα κειμένου που θα στέλνονταν υπό κανονικές συνθήκες στο χρήστη.

Ακολουθεί σύνδεσμος που οδηγεί στην σελίδα της υλοποίησης του συστήματος.

<https://www.tinkercad.com/things/a2G4SbTMq65-home-protection-system>

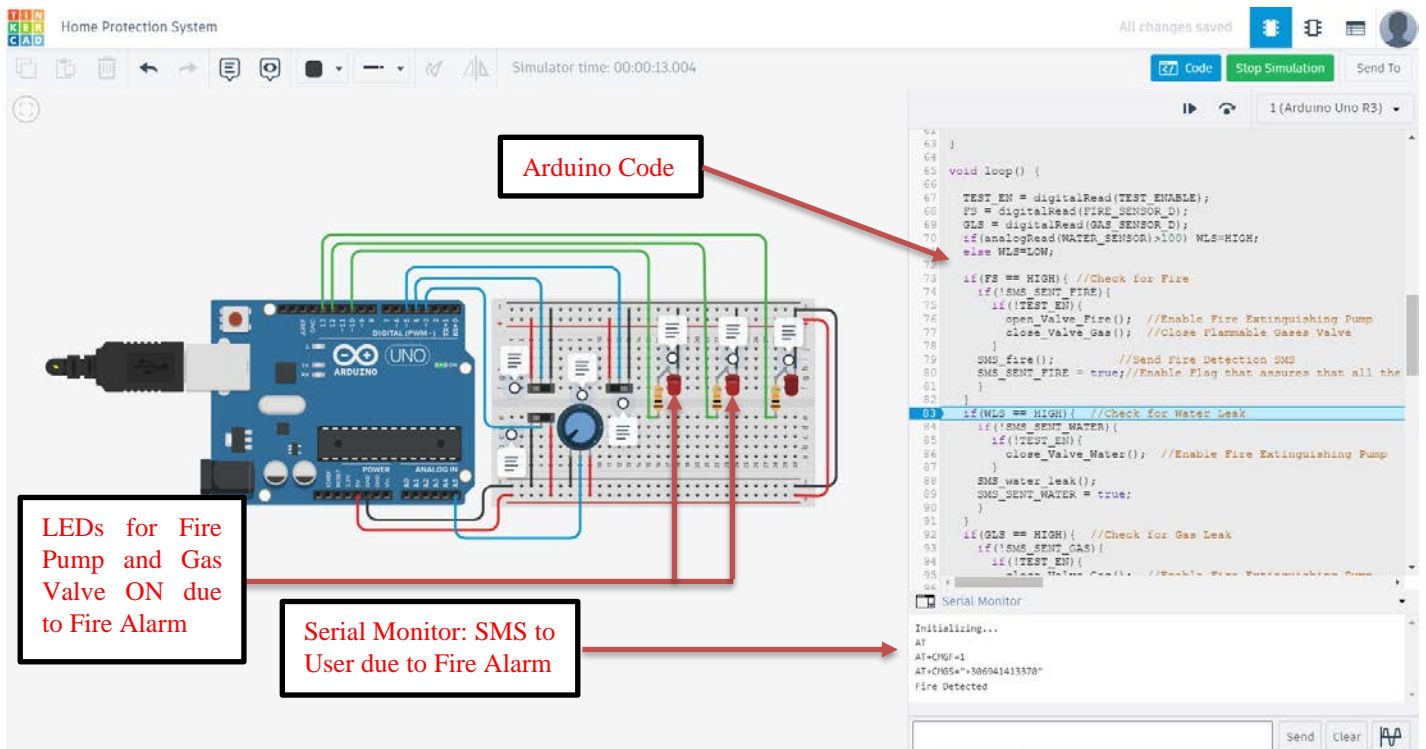
Ακολουθούν φωτογραφίες από την υλοποίηση του συστήματος στο TinkerCAD.



Εικόνα 5: Φωτογραφία υλοποίησης με και χωρίς επισημάνσεις

13 Εκτίμηση Πειραματικών Μετρήσεων

Για την εκτίμηση των Πειραματικών μετρήσεων χρησιμοποιήθηκαν 2 εργαλεία. Πρώτα απ' όλα και όπως αναφέρθηκε και στην ενότητα της Υλοποίησης, χρησιμοποιήθηκε το λογισμικό TinkerCAD με το οποίο έγινε εξομοίωση της λειτουργίας του κυκλώματος καθώς και μερικώς έλεγχος ορθής λειτουργίας των υποσυστημάτων και του μικροελεγκτή. Παρακάτω παρατίθεται στιγμιότυπο από την εξομοίωση της λειτουργίας του κυκλώματος. Για τον λογικό έ-



λεγχο των τιμών που θα έπρεπε να προκύψουν κατά την εκτέλεση του κώδικα χρησιμοποιήθηκε το λογισμικό Excel του Microsoft Office όπου με την χρήση συναρτήσεων προσομοιώθηκε η εκτέλεση του Firmware του συστήματος και μετρήθηκαν οι αντίστοιχες τιμές που θα έπρεπε θεωρητικά να προκύψουν σε συνδυασμό επίσης και με μελέτη των αντίστοιχων Datasheets.

=IF(B17>100;"HIGH";"LOW")

Εικόνα 6: Παράδειγμα χρήσης συνάρτησης στο Excel, που αντιστοιχεί στην ενεργοποίηση της μεταβλητής WLS (εντοπισμός διαρροής νερού) όταν η αναλογική είσοδος από τον αισθητήρα ξεπεράσει την τιμή 100 που αντιστοιχεί σε περίπου 1cm νερού

Input from Fire Sensor	IR Detection	FS boolean variable	RELAY_ENABLE_FIRE	RELAY_ENABLE_WATER	RELAY_ENABLE_GAS
0	<800nm	LOW	LOW	LOW	LOW
1	800nm - 1100nm	HIGH	HIGH	LOW	HIGH

Εικόνα 7: Μέτρηση θεωρητικών αποκρίσεων του συστήματος σε εισόδους από τον αισθητήρα φωτιάς

Input from Gas Sensor	H2 Gas Detection	GLS boolean variable	RELAY_ENABLE_FIRE	RELAY_ENABLE_WATER	RELAY_ENABLE_GAS
0	(<100 (H2 gas))	LOW	LOW	LOW	LOW
1	(100-1000ppm(H2 gas))	HIGH	LOW	LOW	HIGH

Εικόνα 9: Μέτρηση θεωρητικών αποκρίσεων του συστήματος σε εισόδους από τον αισθητήρα διαρροής αερίων

Input from Water Sensor	Water Level	WLS boolean variable	RELAY_ENABLE_FIRE	RELAY_ENABLE_WATER	RELAY_ENABLE_GAS
0	0 cm	LOW	LOW	LOW	LOW
50	0,5 cm	LOW	LOW	LOW	LOW
100	1 cm	LOW	LOW	LOW	LOW
150	1,5 cm	HIGH	LOW	HIGH	LOW
200	2 cm	HIGH	LOW	HIGH	LOW
250	2,5 cm	HIGH	LOW	HIGH	LOW
300	3 cm	HIGH	LOW	HIGH	LOW
350	3,5 cm	HIGH	LOW	HIGH	LOW
400	4 cm	HIGH	LOW	HIGH	LOW
450	4,5 cm	HIGH	LOW	HIGH	LOW
500	5 cm	HIGH	LOW	HIGH	LOW
550	5,5 cm	HIGH	LOW	HIGH	LOW

Εικόνα 8: Μέτρηση θεωρητικών αποκρίσεων του συστήματος σε εισόδους από τον αισθητήρα διαρροής νερού

14 Διαδικασίες Ελέγχου ορθής λειτουργίας

Για την υλοποίηση ελέγχου ορθής λειτουργίας κατά την λειτουργία του κυκλώματος έχει υλοποιηθεί η συνάρτηση *System_Check_Routine()* (Ο κώδικας της συνάρτησης βρίσκεται στο [κάτω μέρος](#) του κώδικα του *Firmware* του συστήματος και έχει επισημανθεί με σκούρο μαύρο *Background*) η οποία ενεργοποιείται όταν ο διακόπτης TEST_ENABLE ενεργοποιηθεί. Κατά την εκτέλεση της ρουτίνας ελέγχου τίθενται οι τρεις μεταβλητές «παρακολούθησης» των τιμών που επιστρέφουν οι αισθητήρες στην λογική τιμή 1 (HIGH) ώστε να «προσομοιωθεί» η ενεργοποίηση και των τριών αισθητήρων (ύπαρξη φωτιάς, διαρροής νερού και αερίου). Στη συνέχεια εκτελείται λογικό AND μεταξύ των 3 μεταβλητών και το αποτέλεσμα αποθηκεύεται στην μεταβλητή *check*. Με αυτό τον τρόπο ελέγχονται τυχόν stuck-at faults που μπορεί να υπάρχουν στο αρχείο καταχωρητών καθώς και επίσης η ορθή λειτουργία της λογικής πράξης AND. Τέλος αποστέλλεται SMS στον χρήστη με το αποτέλεσμα του test με ένδειξη επιτυχίας αν η μεταβλητή *check* έχει το αποτέλεσμα 1 και ένδειξη αποτυχίας και αναλυτικότερων αποτελεσμάτων διαφορετικά. Κατά την εκτέλεση του προγράμματος ελέγχου έχουν παρθεί μέτρα για την αποφυγή της ενεργοποίησης κάποιου από τα Relay για την αποφυγή των False Activation των υποσυστημάτων πυρόσβεσης και διακοπής των παροχών νερού και αερίου αντίστοιχα. Προφανώς αυτό δημιουργεί το πρόβλημα ότι δεν είναι εύκολος ο έλεγχος της ορθής λειτουργίας των Relay. Ο έλεγχος της ορθής λει-

τουργίας των Relay απαιτεί την απενεργοποίηση του διακόπτη TEST_ENABLE, την αποσύνδεση των Βαλβίδων και της αντλίας νερού και τον έλεγχο των Outputs των Relay. Ο χρήστης θα πρέπει να δημιουργήσει «χειροκίνητα» τις συνθήκες ύπαρξης κάποιας καταστροφής, για παράδειγμα με τη χρήση κάποιου αναπτήρα κοντά στον αισθητήρα φωτιάς, την έκχυση μικρής ποσότητας νερού πάνω στον αισθητήρα πλημμύρας, ή την παραγωγή αερίων από εξωτερική πηγή (π.χ. υγραέριο από κάποιο γκαζάκι ή από κάποιο αναπτήρα) κοντά στον αισθητήρα αερίων. Ο έλεγχος αυτός μπορεί να γίνει είτε με τη χρήση κάποιου πολυμέτρου, ή ακόμη και με την ακοή, καθώς τα Relay όταν αλλάζουν κατάσταση παράγουν έναν χαρακτηριστικό θόρυβο «κλικ».

15 Βελτιστοποιήσεις

Η προφανέστερη βελτιστοποίηση του συστήματος, θα ήταν η χρήση περισσότερων αισθητήρων για την κάλυψη μεγαλύτερων χώρων. Συγκεκριμένα θα μπορούσαν να χρησιμοποιηθούν αρκετοί αισθητήρες IR και ανίχνευσης αερίων, με την χρήση πυλών OR, καθώς παράγουν ψηφιακή έξοδο και δεν μας ενδιαφέρει ποιος ακριβώς αισθητήρας ενεργοποιήθηκε. Επίσης είναι δυνατή η χρήση μέχρι 6 αισθητήρων διαρροής νερού (όσες και οι αντίστοιχες αναλογικές εισοδοί του Arduino UNO), με μικρές μετατροπές στο Firmware. Μια ακόμη βελτιστοποίηση θα μπορούσε να είναι η επιλογή κάποιου μικροελεγκτή με γνώμονα τις πραγματικές ανάγκες του συστήματος και όχι με γνώμονα την ευκολία ανάπτυξης προγραμμάτων, το μικρό κόστος και την ευκολία συντήρησης. Κατά την μετάφραση του προγράμματος για τον προγραμματισμό του Firmware του συστήματος, ενημερωνόμαστε ότι αυτό καταλαμβάνει μόλις το 16% (5262 Bytes από 32256 Bytes) του χώρου αποθήκευσης προγράμματος, οι Global μεταβλητές μόλις το 28% (579 Bytes από 2048 Bytes). Ωστόσο για την χρήση κάποιου διαφορετικού μικροελεγκτή ο οποίος θα ήταν πιο κοντά στις πραγματικές ανάγκες του συστήματος θα έπρεπε να θυσιάστούν η συμβατότητα, η ευκολία ανάπτυξης και συντήρησης του Firmware και ίσως και το συνολικό κόστος, αφού θα έπρεπε να αναπτυχθεί επιπλέον μια αναπτυξιακή πλατφόρμα από το 0 για τον αντίστοιχο μικροελεγκτή πάνω στην οποία θα γινόταν αρχικά ο έλεγχος των λειτουργιών του συστήματος. Μια ακόμη βελτιστοποίηση θα ήταν η χρήση μιας εκτυπωμένης πλακέτας κυκλώματος PCB αντί του Breadboard που παρουσιάζεται στην θεωρητική υλοποίηση. Αυτό θα είχε αποτέλεσμα την μείωση των σφαλμάτων από τυχόν αποσυνδέσεις καλωδίων που μπορεί να συμβούν κατά την καθημερινή χρήση καθώς και την βελτίωση της ποιότητας του τελικού προϊόντος. Θα μπορούσε ακόμα να υλοποιηθεί το κομμάτι του Arduino UNO που είναι απαραίτητο για την λειτουργία του κυκλώματος πάνω στην κύρια πλακέτα του συστήματος με αποτέλεσμα το σύστημα να είναι πιο compact. Τέλος αντί ο έλεγχος για την ενεργοποίηση κάποιου αισθητήρα να γίνεται σειριακά και ανά 10 δευτερόλεπτα, θα μπορούσε να γίνει η χρήση Interrupts και όταν ανιχνευθεί η ενεργοποίηση κάποιου αισθητήρα να εκτελείται σχεδόν αμέσως η αντίστοιχη συνάρτηση αντιμετώπισης του αντί-

στοιχου κινδύνου, ωστόσο αυτό ίσως να προκαλούσε πρόβλημα υπερβολικής κατανάλωσης ενέργειας και κατά συνέπεια μείωση αυτονομίας σε περίπτωση διακοπής ρεύματος.

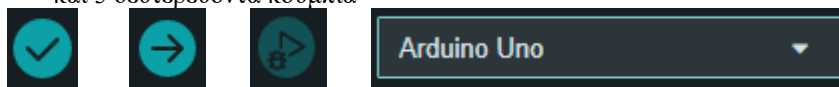
16 Παρουσίαση Εργαλείων-Εφαρμογών

Για την εργασία χρησιμοποιήθηκαν τα εξής εργαλεία και εφαρμογές:

- **Arduino IDE** για την συγγραφή του Firmware και τον προγραμματισμό του μικροελεγκτή του Arduino UNO
- **Autodesk TinkerCAD** για την εξομοίωση και τον έλεγχο των λειτουργιών του συστήματος
- **EasyEDA** για την σχεδίαση του κυκλώματος αυτονομίας και την σχεδίαση της αντίστοιχης πλακέτας PCB

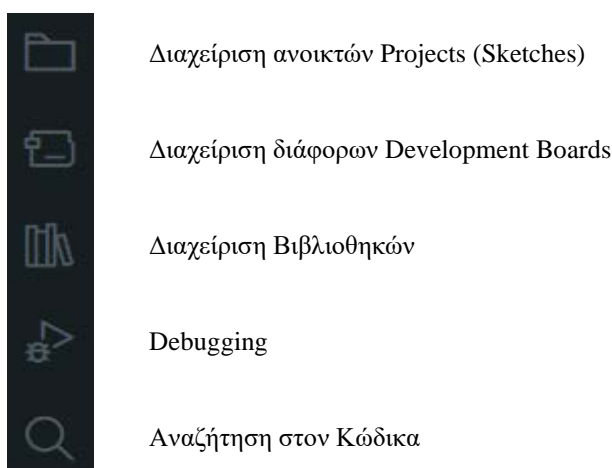
16.1 Arduino IDE

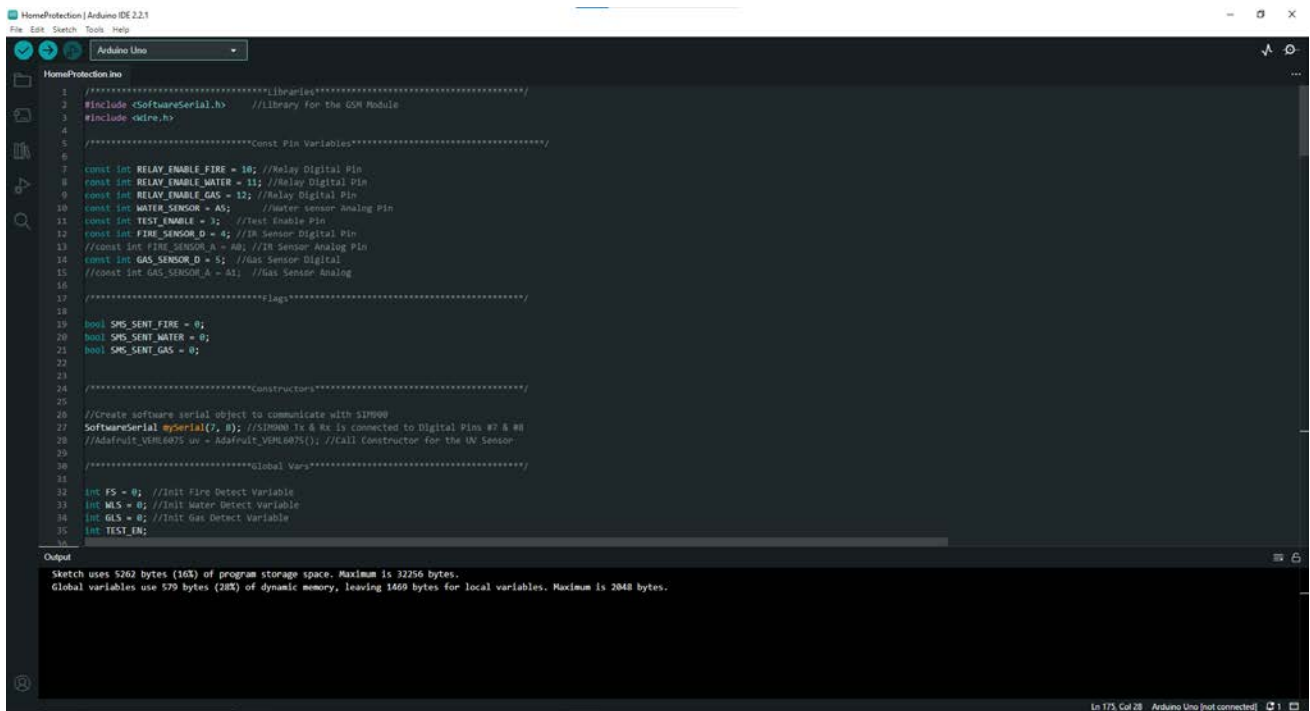
Το Arduino IDE είναι το Integrated Development Environment που παρέχεται από την εταιρία Arduino για την ανάπτυξη προγραμμάτων και τον προγραμματισμό των μικροελεγκτών που είναι συμβατοί με τα προϊόντα της εταιρίας. Το περιβάλλον είναι πολύ απλό στη χρήση καθώς αποτελείται από 3 βασικά και 5 δευτερεύοντα κουμπιά



Βασικά Κουμπιά για: (με σειρά από τα αριστερά προς τα δεξιά)

- 1) Verification/Compilation
- 2) Upload στον μController
- 3) Debugging
- 4) Επιλογή Development Board

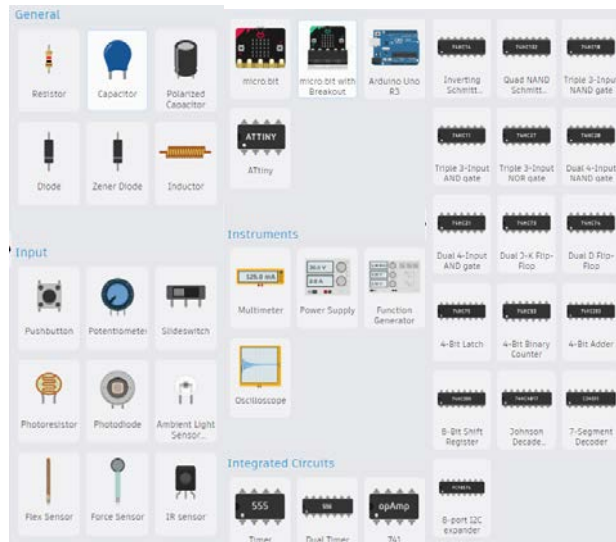




Εικόνα 10: Arduino IDE User Interface

16.2 Autodesk TinkerCAD

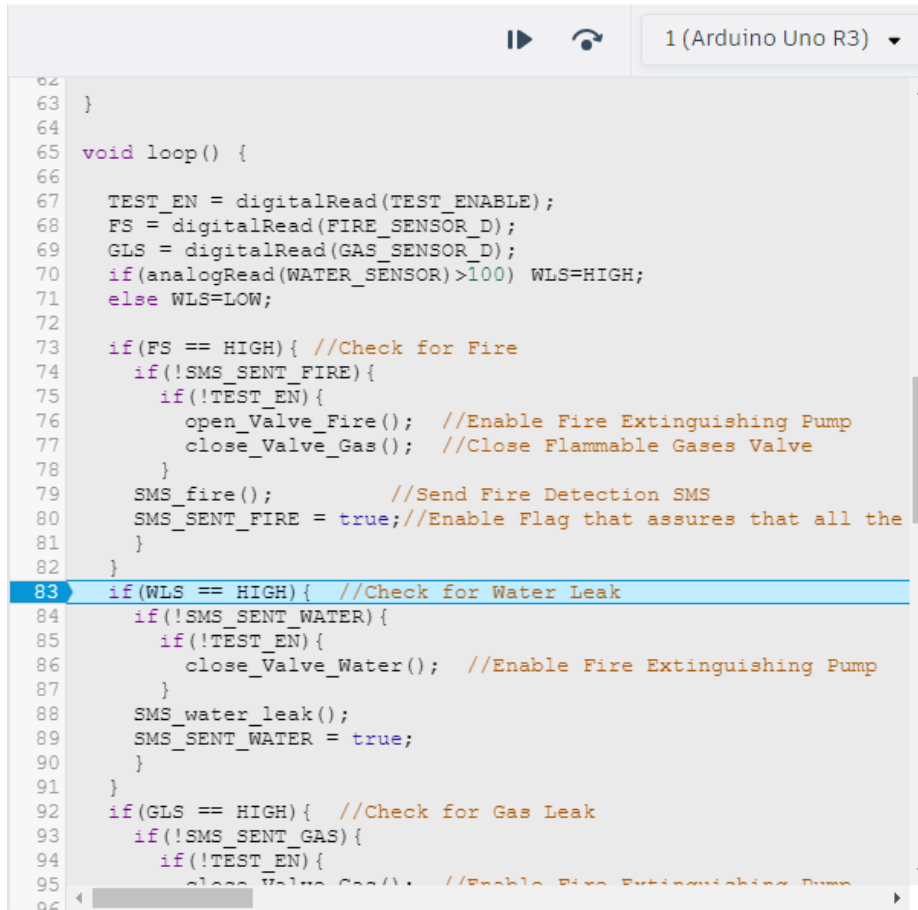
Το Autodesk TinkerCAD είναι ένα online πρόγραμμα σχεδίασης. Χρησιμοποιείται στην σχεδίαση 3-διάστατων αντικειμένων, την σχεδίαση και την εξομοίωση ηλεκτρικών κυκλωμάτων κ.α. Η χρήση του στον σχεδιασμό του συστήματος περιορίστηκε στον σχεδιασμό και εξομοίωση ηλεκτρικών κυκλωμάτων. Για τον σχεδιασμό των κυκλωμάτων διαθέτει μια μεγάλη βιβλιοθήκη από ηλεκτρικά Components όπως LED, Switches, Resistors, Capacitors, Motors, Servos, διάφορους αισθητήρες, όπως Αισθητήρες θερμότητας, υγρασίας, εγγύτητας κτλ και πιο σύνθετα ηλεκτρονικά, όπως Arduino Boards, micro:bit Boards και διάφορα άλλα Ολοκληρωμένα (ICs).



Εικόνα 11: Διάφορα Components που προσφέρονται από το TinkerCAD για τον σχεδιασμό και την εξομίωση κυκλωμάτων



Εικόνα 12: Arduino Serial Monitor για την ανάγνωση των μηνυμάτων που ανταλλάσσονται από το Arduino και τον υπολογιστή μέσω σειριακής επικοινωνίας

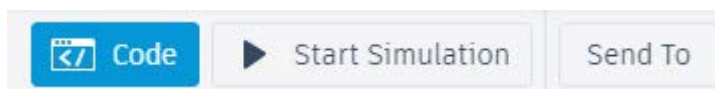


```

62 }
63 }
64
65 void loop() {
66
67     TEST_EN = digitalRead(TEST_ENABLE);
68     FS = digitalRead(FIRE_SENSOR_D);
69     GLS = digitalRead(GAS_SENSOR_D);
70     if(analogRead(WATER_SENSOR)>100) WLS=HIGH;
71     else WLS=LOW;
72
73     if(FS == HIGH){ //Check for Fire
74         if(!SMS_SENT_FIRE){
75             if(!TEST_EN){
76                 open_Valve_Fire(); //Enable Fire Extinguishing Pump
77                 close_Valve_Gas(); //Close Flammable Gases Valve
78             }
79             SMS_fire(); //Send Fire Detection SMS
80             SMS_SENT_FIRE = true; //Enable Flag that assures that all the
81         }
82     }
83     if(WLS == HIGH){ //Check for Water Leak
84         if(!SMS_SENT_WATER){
85             if(!TEST_EN){
86                 close_Valve_Water(); //Enable Fire Extinguishing Pump
87             }
88             SMS_water_leak();
89             SMS_SENT_WATER = true;
90         }
91     }
92     if(GLS == HIGH){ //Check for Gas Leak
93         if(!SMS_SENT_GAS){
94             if(!TEST_EN){
95                 close_Valve_Gas(); //Enable Fire Extinguishing Pump
96             }
97         }
98     }
99 }

```

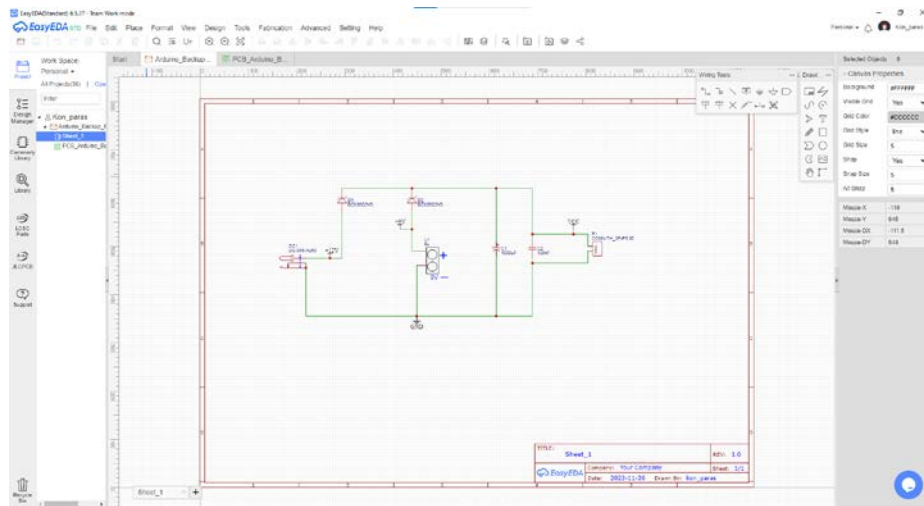
Εικόνα 13: Χώρος συγγραφής Firmware για τον προγραμματισμό του Arduino (ή οποιασδήποτε άλλης πλατφόρμας) που χρησιμοποιείται στο κύκλωμα



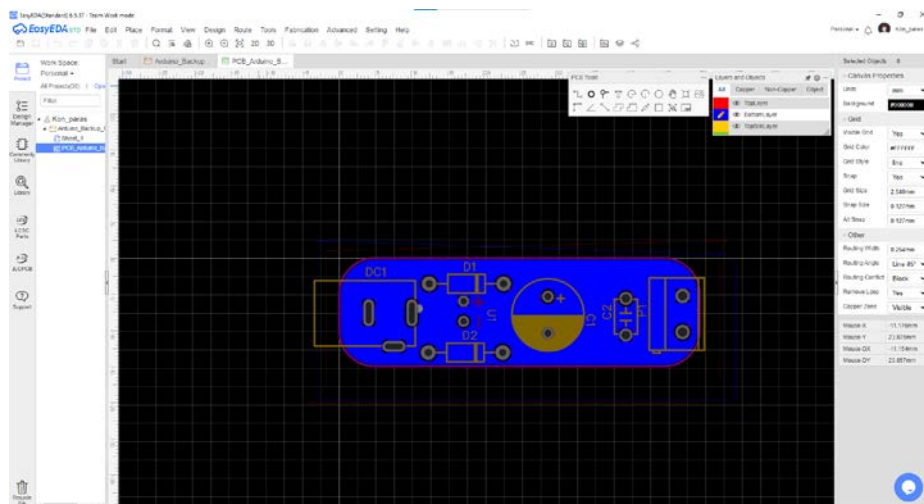
Εικόνα 14: Κουμπιά για την εμφάνιση του χώρου προγραμματισμού, έναρξη εξομοίωσης, και διαμοιρασμού του Project με την αντίστοιχη σειρά

16.3 EasyEDA

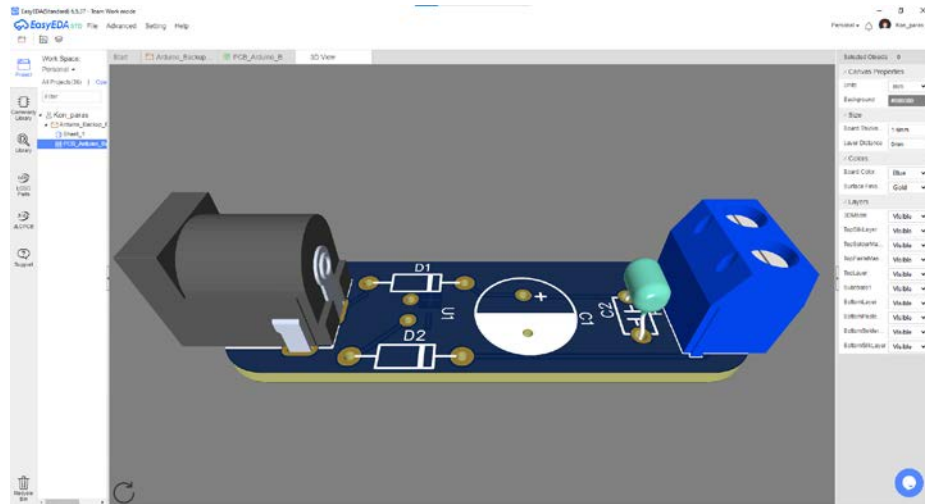
Το EasyEDA είναι ένα πρόγραμμα σχεδιασμού ηλεκτρικών κυκλωμάτων και εκτυπωμένων πλακετών. Διαθέτει μεγάλες βιβλιοθήκες Ηλεκτρονικών Components που περιέχουν στοιχεία σχετικά με την φυσική μορφή καθώς και 3D μοντέλα, τα οποία χρησιμεύουν στο σχεδιασμό της πλακέτας του εκάστοτε κυκλώματος καθώς και στο 3d Rendering αυτής μετά την ολοκλήρωση του σχεδιασμού.



Εικόνα 15: UI Σχεδίασης του σχηματικού του κυκλώματος



Εικόνα 16: UI Σχεδίασης της πλακέτας του κυκλώματος



Εικόνα 17: UI 3D View της πλακέτας

17 References

1. N N Mahzan: Design of an Arduino-based home fire alarm system with GSM module (<https://iopscience.iop.org/article/10.1088/1742-6596/1019/1/012079/pdf>) (2018)
2. Arduino Gas Detecting Alarm System (<https://www.instructables.com/Arduino-Gas-Detecting-Alarm-System/>)
3. SOS Leak Sensor with an Arduino (<https://bluerobotics.com/learn/sos-leak-sensor-with-an-arduino/>)
4. Arduino Comparison Guide (https://www.sparkfun.com/standard_arduino_comparison_guide)
5. Send Receive SMS & Call with SIM900 GSM Shield & Arduino (<https://lastminuteengineers.com/sim900-gsm-shield-arduino-tutorial/>)
6. KY-026 FLAME SENSOR MODULE (<https://arduinomodules.info/ky-026-flame-sensor-module/>)
7. ARDUINO WATER LEVEL SENSOR TUTORIAL (<https://www.thegeekpub.com/236571/arduino-water-level-sensor-tutorial/>)
8. Interfacing MQ-8 Hydrogen Gas Sensor Module with Arduino (<https://electropeak.com/learn/interfacing-mq-8-smoke-gas-sensor-module-with-arduino/>)
9. Arduino IDE (<https://docs.arduino.cc/software/ide-v2>)
10. EasyEDA Editor STD (<https://easyeda.com/editor>)
11. Official Guide to Tinkercad Circuits (https://assets.ctfassets.net/jl5ii4oqrddmc/4sMFqe3rDlbUymJt0I4yh/85a4487f7fe274e74c19870ae4679fc1/tinkercad-guides_circuits-Printable.pdf)
12. AUTODESK TincerCAD: (<https://www.tinkercad.com/>)