Τεκμηρίωση Δεύτερου Θέματος εξαμηνιαίας εργασίας μαθήματος «Προγραμματισμός Ενσωματωμένων Συστημάτων σε Περιβάλλοντα Edge»:

Εαρινό Εξάμηνο 2021 – 2022

Αλέξανδρος Βεντούρας 21106

Χρήστος Τζέλης 21115

A.Verilog:

Περιγραφή υλοποίησης & στιγμιότυπα εκτέλεσης:

Η υλοποίηση του module σε Verilog έγινε με χρήση του edaplayground. Συγκεκριμένα, κατά τα πρότυπα των προγραμμάτων που διδάχθηκαν στα εργαστήρια του μαθήματος, δημιουργήθηκαν τα δύο προγράμματα design.sv και testbench.sv.

Σε πρώτη φάση, στο πρόγραμμα του σχεδιασμού, δημιουργήθηκε ένα module το οποίο καθορίζει την συμπεριφορά του συναγερμού, ορίζοντας οκτώ εισόδους (για τους τρεις αισθητήρες motion1, motion2 και reed, καθώς και για τα πέντε κανάλια εισόδου του κωδικού του χρήστη) και δύο εξόδους (alarm και active).

Έπειτα, με τη χρήση της εντολής always, ορίζεται πως όποτε αλλάζει η τιμή σε ένα από τα οκτώ κανάλια εισόδου, θα ελέγχεται η πεντάδα των bits της εισόδου του χρήστη.

- Εάν αυτή είναι η 00001, (η οποία, επειδή τα ψηφία είναι αντιστοιχισμένα με αύξουσα σημαντικότητα, αντιστοιχεί στον αριθμό 32), τότε η τιμή του active τίθεται σε 1, και εάν κάποια από τις εισόδους των αισθητήρων είναι 1 (εδώ βάσει σύμβασης ορίζουμε αυτήν την τιμή ως αυτήν που δείχνει παραβίαση), τότε ενεργοποιείται και ο συναγερμός, αλλιώς παραμένει ανενεργός. Εάν το alarm είναι ήδη ενεργοποιημένο, τότε ο συναγερμός

παραμένει ενεργός χάρη στο τελευταίο σκέλος της συνθήκης του if στη γραμμή 12.

- Αν πάλι η είσοδος του χρήστη είναι 00100 (η οποία ως παλίνδρομη ακολουθία αριθμών όπως και αν τη διατάξουμε παραμένει αντιστοιχιζόμενη με τον αριθμό 4), τότε απλώς απενεργοποιούνται τόσο ο συναγερμός, όσο και η όπλιση.
- Σε περίπτωση που ο χρήστης εισάγει κάτι διάφορο των παραπάνω, τότε τυπώνεται ένα μήνυμα σφάλματος.

Ο κώδικας του εν λόγω προγράμματος είναι ο παρακάτω:

```
module alarm (m1, m2, r, k0, k1, k2, k3, k4, active, alarm);//Ver
  output active, alarm;
  input k0, k1, k2, k3, k4;
  input m1, m2, r;
  reg active, alarm;
  always @(k0 or k1 or k2 or k3 or k4 or m1 or m2 or r)
  begin
    case ({k0, k1, k2, k3, k4})
      5'b00001:
        begin
          active = 1;
          if((m1 == 1) | (m2 == 1) | (r == 1) | (alarm == 1))
            alarm = 1;
          else
            alarm = 0;
        end
      5'b00100:
        begin
          active = 0; alarm = 0;
        end
      default:
        $display("keyboard signal other than 4 or 32!!!");
    endcase
  end
endmodule
```

Σε δεύτερη φάση, στο πρόγραμμα ελέγχου, χρησιμοποιείται ένα στιγμιότυπο του συναγερμού αυτού, στο οποίο εισάγονται διαδοχικά διάφορες τιμές, προκειμένου

```
[2022-06-04 14:22:22 EDT] iverilog '-Wall' design.sv testbench.sv  && unbuffer vvp a.out
               time    m1    m2     r    k0    k1    k2    k3    k4  active  alarm
                  0     0     0     0     0     0     0     0     1       1      0
                 10     0     1     1     0     0     0     0     1       1      1
                 20     0     1     1     0     0     1     0     0       0      0
                 30     1     0     1     0     0     1     0     0       0      0
                 40     0     0     0     0     0     1     0     0       0      0
                 50     0     0     0     0     0     0     0     1       1      0
                 60     0     1     1     0     0     0     0     1       1      1
                 70     0     0     0     0     0     0     0     1       1      1
                 80     0     0     0     0     0     1     0     0       0      0
Done
```

να ελεγχθεί η ορθότητα του προγράμματος σχεδιασμού. Η αλληλουχία αυτών των τιμών φαίνεται στο παρακάτω στιγμιότυπο οθόνης, το οποίο επεξηγούμε αμέσως μετά.

Αρχικά, γίνεται η αρχικοποίηση των τιμών, έτσι ώστε όλοι οι αισθητήρες να έχουν την τιμή 0, και το σύστημα να είναι οπλισμένο (t = 0).

Έπειτα, θέτουμε την τιμή 1 σε δύο από τους αισθητήρες, και πράγματι βλέπουμε ότι ενεργοποιείται ο συναγερμός (t = 10).

Στη συνέχεια, αφοπλίζουμε, και το σύστημα αντιδρά αναμενόμενα, θέτοντας τα alarm και alarm σε 0 (t = 20).

Ακολούθως, αλλάζουμε τις τιμές κάποιων αισθητήρων, αλλά αφού το σύστημα είναι αφοπλισμένο, δεν αντιδρά (t = 30).

Αμέσως μετά, θέτουμε την τιμή 0 σε όλους τους αισθητήρες (t = 40), και οπλίζουμε (t = 50).

Έπειτα, θέτουμε πάλι την τιμή 1 σε δύο από τους αισθητήρες, και πράγματι βλέπουμε ότι ενεργοποιείται εκ νέου ο συναγερμός (t = 60).

Αυτή τη φορά όμως, δεν αφοπλίζουμε άμεσα, αλλά θέτουμε την τιμή 0 σε όλους τους αισθητήρες, όμως επειδή ακριβώς δεν έχουμε αφοπλίσει, το σύστημα συνεχίζει να έχει την τιμή 1 στο alarm (t = 70).

Η αφόπλιση γίνεται την επόμενη χρονική στιγμή, όπου πράγματι βλέπουμε ότι ο συναγερμός σιωπά.

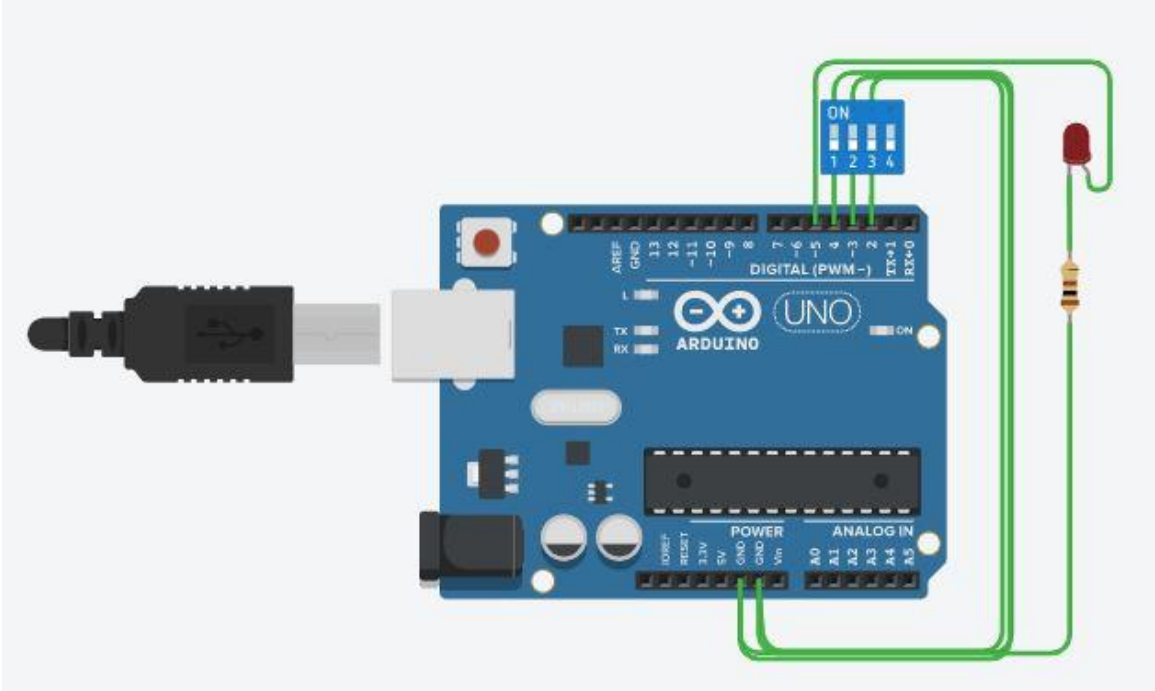
Ο κώδικας του δεύτερου προγράμματος είναι ο παρακάτω:

```
module top();
  reg m1, m2, r, k0, k1, k2, k3, k4;
  wire active, alarm;
  alarm my_alarm(m1, m2, r, k0, k1, k2, k3, k4, active, alarm);
  initial
  begin
    $display("\t\ttime\tm1\tm2\tr\tk0\tk1\tk2\tk3\tk4   active
alarm");
    $monitor($time
,"\t%b\t%b\t%b\t%b\t%b\t%b\t%b\t%b\t%b\t%b",m1,m2,r,k0,k1,k2,
k3,k4,active,alarm);
//     Arm is 10000, disarm is 00100
    {m1,m2,r}=3'b000;
    {k4,k3,k2,k1,k0}=5'b10000;
    #10
    {m1,m2,r}=3'b011;
    #10
    {k4,k3,k2,k1,k0}=5'b00100;
    #10
    {m1,m2,r}=3'b101;
    #10
    {m1,m2,r}=3'b000;
    #10
    {k4,k3,k2,k1,k0}=5'b10000;
    #10
    {m1,m2,r}=3'b011;
    #10
    {m1,m2,r}=3'b000;
    #10
    {k4,k3,k2,k1,k0}=5'b00100;
    #10 $finish;
  end
endmodule
```

B. Arduino:

Περιγραφή υλοποίησης & στιγμιότυπα εκτέλεσης:

Η υλοποίηση του συστήματος σε Arduino έγινε με τη βοήθεια του εξομοιωτή Tinkercad. Συγκεκριμένα, υλοποιήθηκε ένα κύκλωμα όπως αυτό της παρακάτω εικόνας:



Το κύκλωμα περιλαμβάνει μία συσκευή τεσσάρων διακοπτών, εκ των οποίων οι τρεις πρώτοι αναπαριστούν τους αντίστοιχους αισθητήρες του συστήματος συναγερμού. Οι τρεις αυτοί διακόπτες συνδέονται με αντίστοιχο αριθμό θυρών εισόδου δεδομένων, και όλες είναι συνδεδεμένες με τη γείωση. Εκτός αυτού, προστέθηκε ένας LED λαμπτήρας, προκειμένου να είναι περισσότερο εμφανής η λειτουργία του συναγερμού.

Στις επόμενες παραγράφους, γίνεται ανάλυση του κώδικα που χειρίζεται τη λειτουργία του παραπάνω κυκλώματος.

Αρχικά, ορίζεται ότι ο συναγερμός είναι απενεργοποιημένος, και ότι το σύστημα είναι αφοπλισμένο. Επίσης με μία εντολή προεπεξεργαστή καθορίζουμε ότι η είσοδος 5 εφεξής θα ονομάζεται LED (ουσιαστικά, με αυτήν την εντολή ορίζουμε ότι ο προεπεξεργαστής θα αντικαταστήσει το alias LED με την τιμή 5).

Στη συνέχεια, στη συνάρτηση setup() ορίζουμε τις εισόδους των διακοπτών, καθώς και των λαμπτήρων LED. Επίσης ορίζουμε το baud rate για την είσοδο και έξοδο δεδομένων από και προς τον χρήστη μέσω της κονσόλας

Έπειτα, στη συνάρτηση loop() ορίζουμε τις μεταβλητές που θα συνδέονται με την είσοδο του χρήστη και με τους αισθητήρες, και κατόπιν με τη βοήθεια ενός if statement, κάθε φορά που υπάρχει νέα είσοδος χρήστη από το serial monitor, ελέγχεται αν είναι ίση με 32 ή με 4.

Αν είναι ίση με 32, τότε σε περίπτωση που όλες οι τιμές των αισθητήρων είναι ίσες με 1 (εδώ με 1 ορίζουμε την τιμή όπου δεν νοείται παραβίαση), τότε το σύστημα οπλίζει, αλλιώς τυπώνεται ένα μήνυμα αποτυχίας, με την παρότρυνση προς τον χρήστη προκειμένου να κλείσει όλους τους διακόπτες.

Αν πάλι η είσοδος είναι ίση με 4, τότε το σύστημα αφοπλίζει και ο συναγερμός τίθεται εκτός λειτουργίας. Επίσης σβήνουν και οι δύο λαμπτήρες LED.

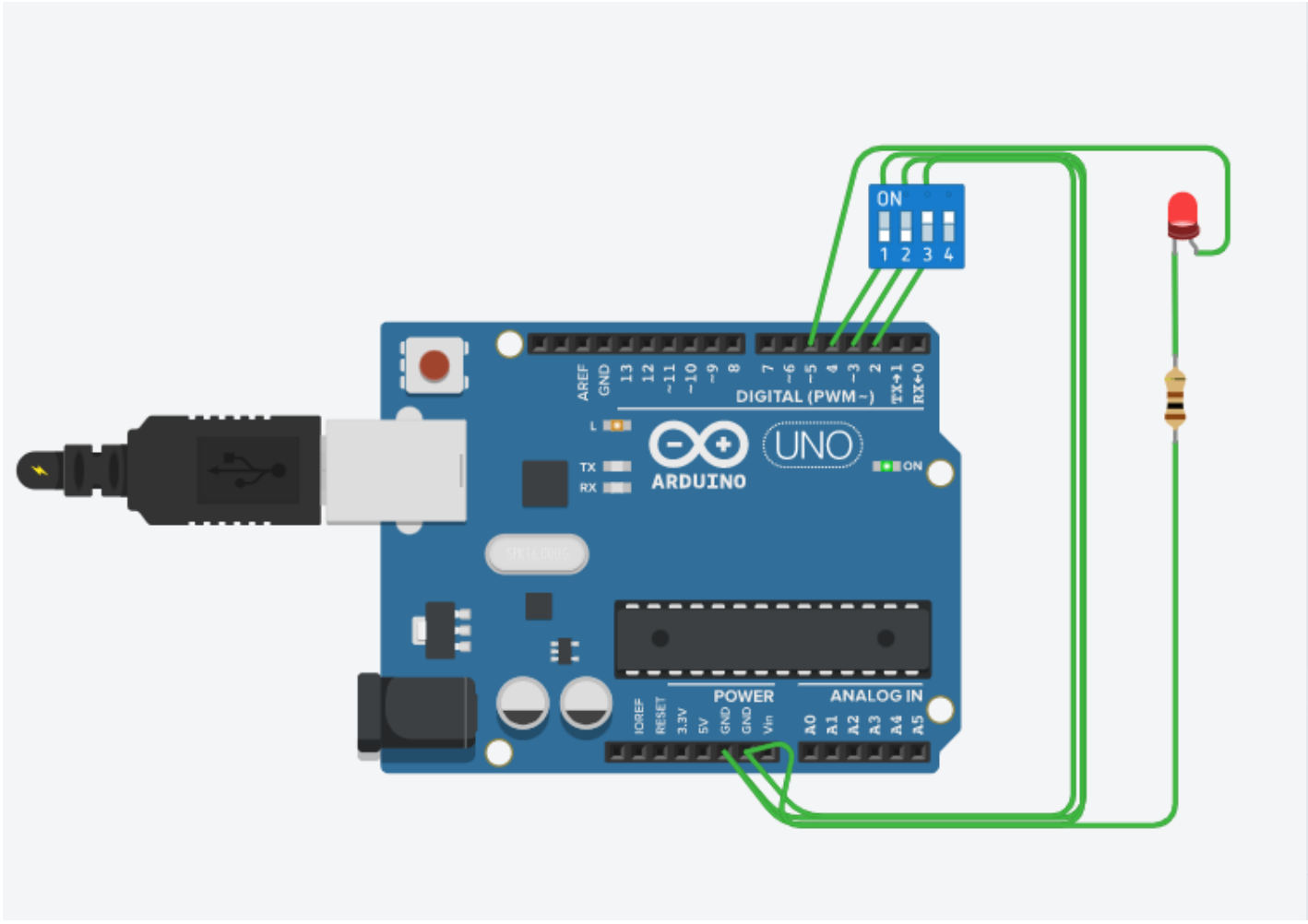
Κατόπιν, υπάρχει μία άλλη εντολή if, η οποία σε κάθε κύκλο μηχανής ελέγχει εάν το σύστημα είναι οπλισμένο, και εάν είναι, ελέγχει περαιτέρω για το εάν έχει ενεργοποιηθεί κάποιος αισθητήρας, ή εάν έχει προηγουμένως ενεργοποιηθεί ο συναγερμός. Σε περίπτωση που ισχύει κάτι από αυτά, ενεργοποιείται ο συναγερμός, τυπώνονται σχετικά μηνύματα στην οθόνη, και αρχίζουν να αναβοσβήνουν τα LED.

Ο παραπάνω περιγραφόμενος κώδικας απεικονίζεται στο screenshot της επόμενης σελίδας. Επίσης στην μεθεπόμενη σελίδα υπάρχει και ένα στιγμιότυπο εκτέλεσης της υλοποίησης στο Arduino, τη στιγμή που βρίσκεται σε φάση που έχει οπλίσει ο συναγερμός (φαίνεται το σχετικό μήνυμα ARM), έχει παραβιαστεί μία από τις εισόδους (η 3^{η}), και έχει ενεργοποιηθεί ο συναγερμός.

```
bool alarm = false;
bool armed = false;
#define LED 5
void setup()
{
  pinMode(2, INPUT_PULLUP);
  pinMode(3, INPUT_PULLUP);
  pinMode(4, INPUT_PULLUP);
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(LED, OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  int ch;
  int m1 = digitalRead(2);
  int m2 = digitalRead(3);
  int r = digitalRead(4);
  if(Serial.available()) {
    ch = Serial.parseInt();
    if(ch == 32) {
      if(m1 == 1 & m2 == 1 & r == 1) {
        Serial.println("ARM");
        armed = true;
      } else {
        Serial.println("ARM FAILURE, SET OFF ALL SWITCHES.");
      }
    } else if(ch == 4) {
      armed = false;
      alarm = false;
      Serial.println("DISARM");
      digitalWrite(LED_BUILTIN, LOW);
      digitalWrite(LED, LOW);
    }
  }
  if(armed) {
    if(m1 == 0 | m2 == 0 | r == 0 | alarm) {
      alarm = true;
      Serial.println("BURGLARS!!!!");
      digitalWrite(LED_BUILTIN, HIGH);
      digitalWrite(LED, HIGH);
      delay(250);
      digitalWrite(LED_BUILTIN, LOW);
      digitalWrite(LED, LOW);
      delay(250);

    }
  }
}
```




```
1  bool
2  bool
3  #defi
4  void
5  {
6    pinM
7    pinM
8    pinM
9    pinM
10   pinM
11   Seri
12 }
13
14 void
15 {
16   int
17   int
18   int
19   int
20   if(
21     cl
22     i
23
24
25
26
27
```

Serial M

```
ARM
BURGLARS!!!!
BURGLARS!!!!
BURGLARS!!!!
BURGLARS!!!!
BURGLARS!!!!
BURGLARS!!!!
BURGLARS!!!!
```