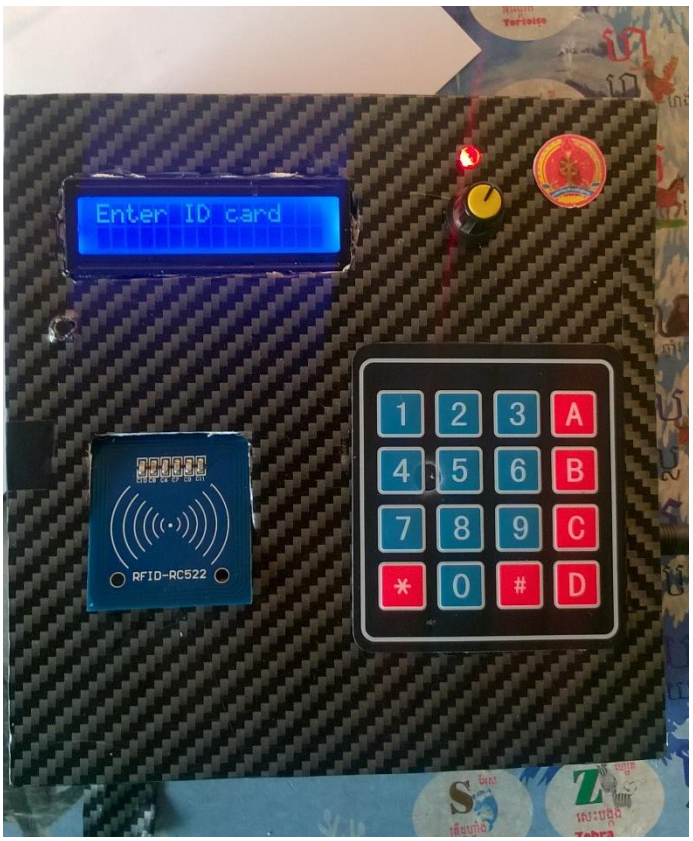
Royal University of Phnom Penh
Faculty of Engineering
Department of Telecommunication & Electronics

Report of RFID Authorized Access project with RC522 and ATMEGA328



Course of Electronics System Design, Semester 1 of Year 4
Instructor: CHAN Tola
Student names: AN Panharith
SIET Sophort
MEAN Sareth
SAM Sarom
CHIM Socheat
and PHAN Theara

[This page was intentionally left blank]

Contents

[This page was intentionally left blank]

Abstract

Objective: The main purpose was to create a RFID authorized access with two features of verifications which are RFID tag and passcode. The information of accesses was preferred to display on a 16x2 LCD.

Main components:

- RC522 as the RFID reader.
- 13.56MHz RFID cards as the reader's tags
- ATMEGA328 as the microcontroller.
- 16x2 LCD
- Keypad module

Methodology and Experiment:

- Planning for power supply, the circuit schematic and the flow chart of the code.
- Drawing the circuit in ISIS 7 Professional.
- Drawing the sigma of the circuit components in ARES Professional.
- Creating the physical circuit board.
- Packing into one product

Result: We successfully implemented a RFID verification product which we would like to call RFID Authorized Access (RFID AA) with two features of verifications RFID tag and passcode.

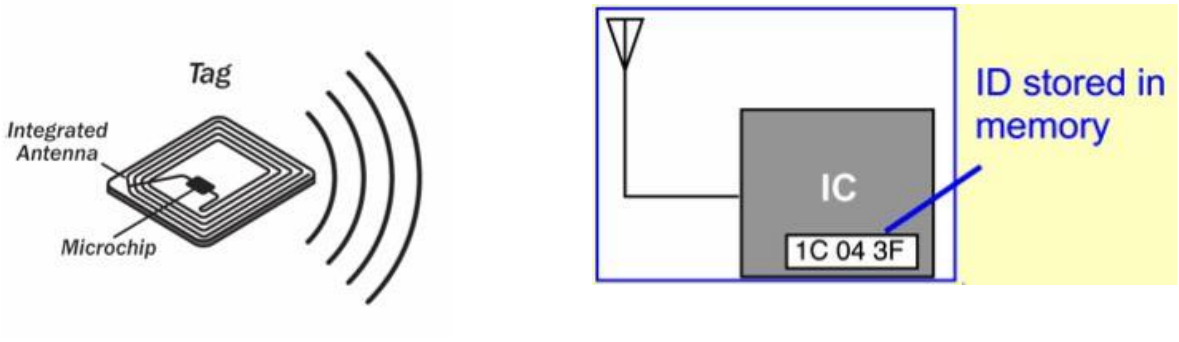
Conclusion: RFID AA offers two features of authorized verifications described above. It contributes to engineers who wish to integrate this product into other applications for example doors' locking/unlocking.

Keywords: RFID: reader, 13.56MHz card, serial-number, passive and active tags.
ATMEGA328, MFRC522 IC, micro-controller, authorized-access.

[This page was intentionally left blank]

I. Introduction

Radio frequency identification (RFID) wirelessly identifies and tracks the uniqueness of its tags by using electromagnetic fields in a short range of scanning. RFID tagging plays role as a normal ID system but uses radio frequency devices to verify the tags. Each RFID made of a microchip, memory, and an integrated antenna as show in figure 1. The main purpose of RFID tag's memory is to store the tag's ID. The main function of RFID system is to read each unique information which is commonly serial-number. It means that the RFID reader reads and identifies RFID tags in a distance of scanning.



(a) **(b)**

Figure 1: (a) microchip and the antenna of RFID tag. (b) the memory of the RFID tag.

There are two types of RFID tag which are active type and passive types. In this project we used passive tags. A passive tag uses radio energy which is transmitted from the reader as a supplying assistance device. However, it operates in a very short distance according to the propagation frequency for example low frequency (LF) tags, high frequency (HF) tag, ultrahigh-frequency (UHF) tags. Regularly, RFID passive tags use LF, HV and UHF band.

Band	**Regulations**	**Range**	**Data speed**
120–150 kHz (LF)	Unregulated	10 cm	Low
13.56 MHz (HF)	ISM band worldwide	10 cm–1 m	Low to moderate
433 MHz (UHF)	Moderate	1–100 m	Moderate
865-868 MHz (Europe)	ISM band	1–12 m	Moderate to high
902-928 MHz (North America) UHF			
2450-5800 MHz (microwave)	ISM band	1–2 m	High
3.1–10 GHz (microwave)	Ultra-wide band	Up to 200 m	High

Table 1: Common frequency band used by RFID tags

RFID tags are uniquely identified by serial number. By recognizing the serial number of the RFID tags, the microcontroller is able to verify the authorized access.

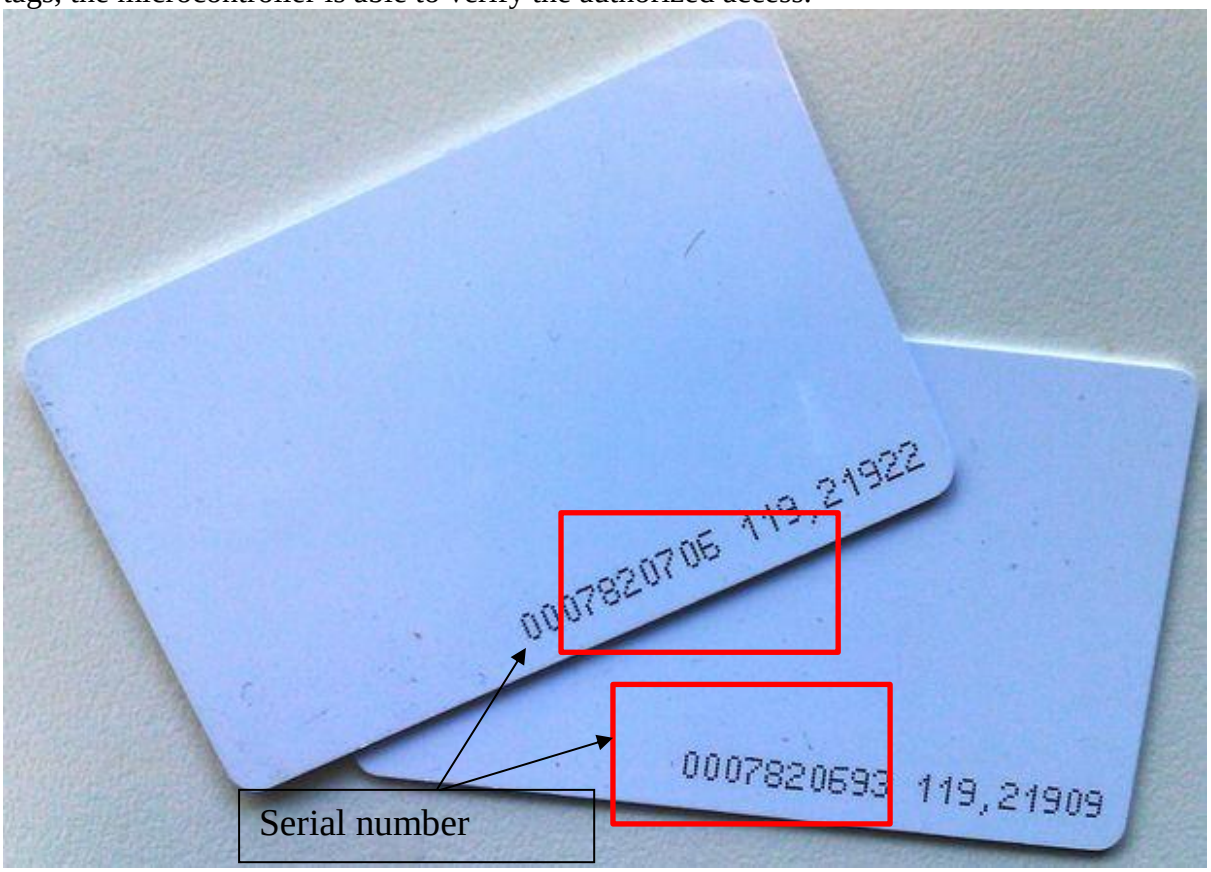


Figure 2: RFID tags and its own unique serial numbers

Note: In this project, the serial number is not shown on the RFID tag.

In this project, we used MFRC522 RFID reader Arduino based. MF RC522 is a read/write integrated card chip of 13.56MHz. The MF RC522 use of advanced modulation and demodulation concept completely integrated in the 13.56MHz all kinds of passive contactless communication methods and protocols. 14443A compatible transponder signal. The digital part handles the ISO14443A frames and error detection. In addition, support Quick CRYPTO1 encryption algorithm, the term verification MIFARE series. MFRC522 support MIFARE series of high-speed non-contact communication, two-way data transfer rates up to 424kbit / s. As 13.56MHz highly integrated card reader series chip new family, the MF RC522 MF RC500 MF RC530 there are many similarities, but also have many of the characteristics and differences. Communication between it and the host SPI mode, helps to reduce the connection, reduce PCB board volume and reduce costs.

Specification:

1) Module Name: MF522-ED
2) Working current：13—26mA/ DC 3.3V
3) Standby current：10-13mA/DC 3.3V
4) sleeping current：<80uA
5) peak current：<30mA
6) resistance: 614Ω
7) Working frequency：13.56MHz
8) Card reading distance ：0～60mm（mifare1 card）

9) Protocol：SPI
10) data communication speed：Maximum 10Mbit/s
11) Card types supported：mifare1 S50、mifare1 S70、mifare UltraLight、mifare Pro mifare Desfire
12) Dimension：40mm×60mm
13) Environment
14) Working temperature：-20—80 degree
15) Storage temperature：-40—85 degree
16) Humidity：relevant humidity 5%—95%
17) Max SPI speed: 10Mbit/s

II. Components and software

A. Components and tools

- ATMEGA328
- LCD 16x2
- RFID-RC522
- Potentiometer 20K
- 1 button
- Resistors: 220, 250,330 2K 10K with ohm as the unit
- Capacitor (1000uF,470uF and 22nF)
- 1 LED
- Regulator 7805
- T-block
- 16cm x 16 cm Copper board
- 2 oscillators (16.000)
- 12volt to 5volt DC adapter
- Wire

B. Software

- ISIS 7 professional (Protues)
- ARES professional
- Arduino IDE

III. Methodology

A. How it physically works

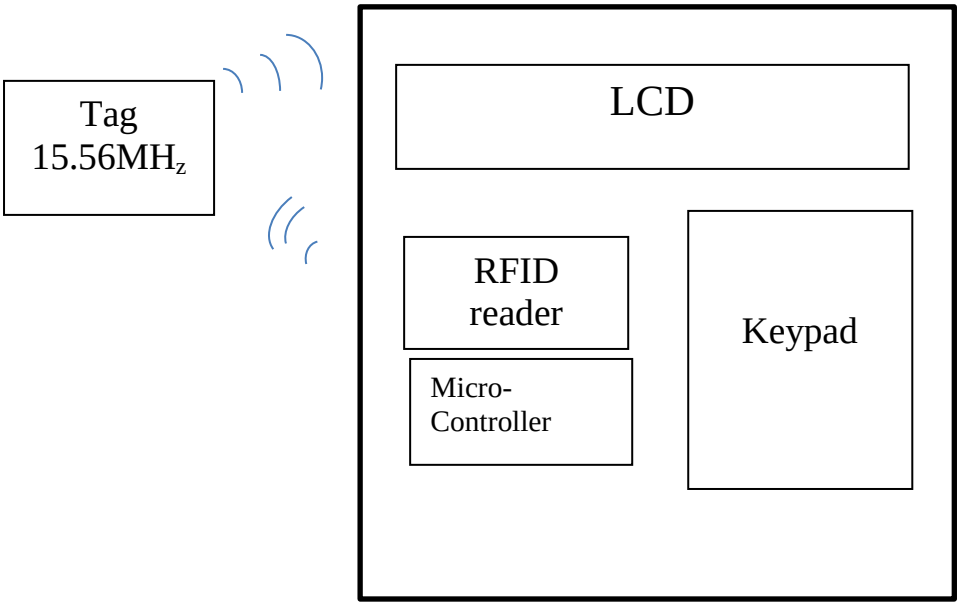


Figure 3: Brief summary of how this project works

Authorized access application with RFID
Step 1-Scanning: The circuit is always functioning and waiting for reading any RFID tags. Within the maximum distance of 60 milli-meter, the RFID is able to do scanning.
Step 2-Activating the tag: Since 13.56MHz RFID tags are passive tags. The RFID reader does scanning by transmit radio energy to activate the tag. Once the tag is activated, it transmits back the serial number to the RFID reader.
Step 3-Reading serial number: RFID reader receives the serial number and forward to the micro-controller.
Step 4-Showing information: The micro-controller verifies the serial number to the user's information. The verification result will be show on the LCD. If the tag is correct, all important information except for the passcode will be displayed on the LCD
Step 5- Requesting passcode: when the correct tag is read, the micro-controller will request passcode from the user.
Step 6- authorized access: authorized access occurs as long as the passcode is correctly input.

B. Power supply schematic

Unlike any Arduino hardware-based projects which the 3 volts and 5 volts power supplies, the most suitable condition for most electronic devices, are automatically provided. The power supplies of this application were a little bit more complicated.

We had a 12volts DC adapter, the main supply of the circuit, which we needed to share to the main physical components such as RC522, LCD, ATMEGA328. LCD needs 5volts power supply. We can use IC-

7805, the 1.5A DC converter with 5volts output from 12 volts input. ATMEGA328 needs 5volt of DC power to operate. RC522 with a build-in 614ohms resistance needs maximum power supply of 3.3 volts.

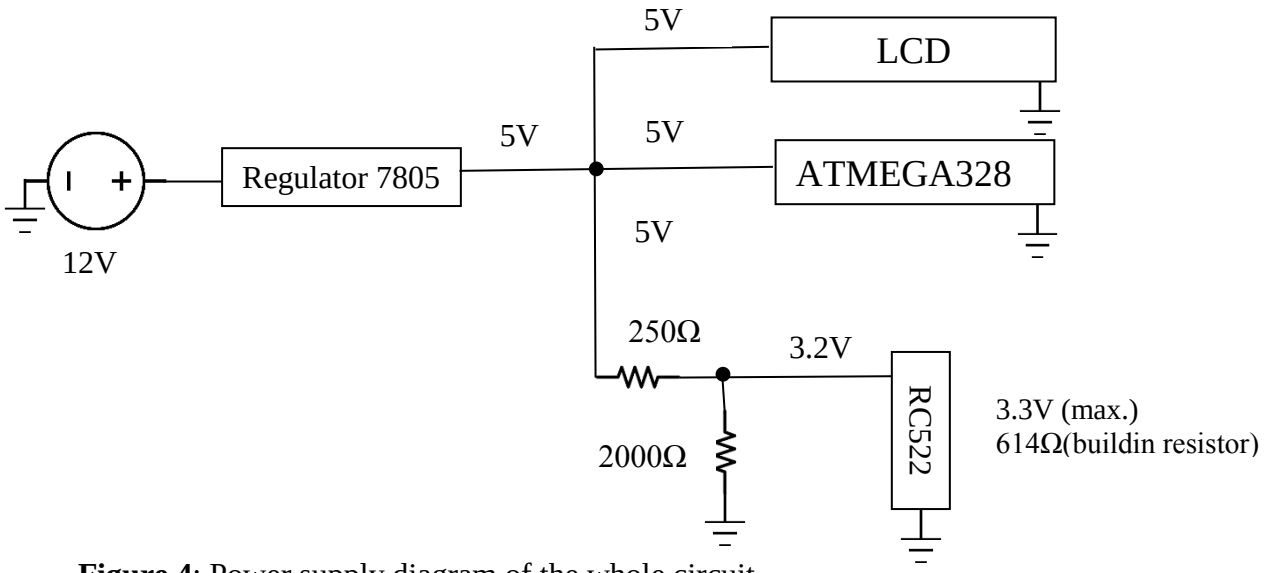
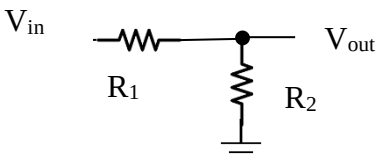


Figure 4: Power supply diagram of the whole circuit

How to divide from 5V to 3.3V at maximum power supply for RC522



$$V_{out} = V_{in} \cdot \frac{R_2}{R_1 + R_2}$$

Figure 5: diagram of voltage dividing using pure a couple of resistors

V_{in} = 5V
R_1 = 250Ω
R_2 = 470 Ω is the equivalent resistor between 2000 Ω resistor and 614Ω RC522's building resistor

Thus, $Vout = 5 \times \left[\frac{470}{250+470}\right] = 3.2V$

3.2volt of voltage supply is good enough for RC522 operation.

C. Flow chart of the code

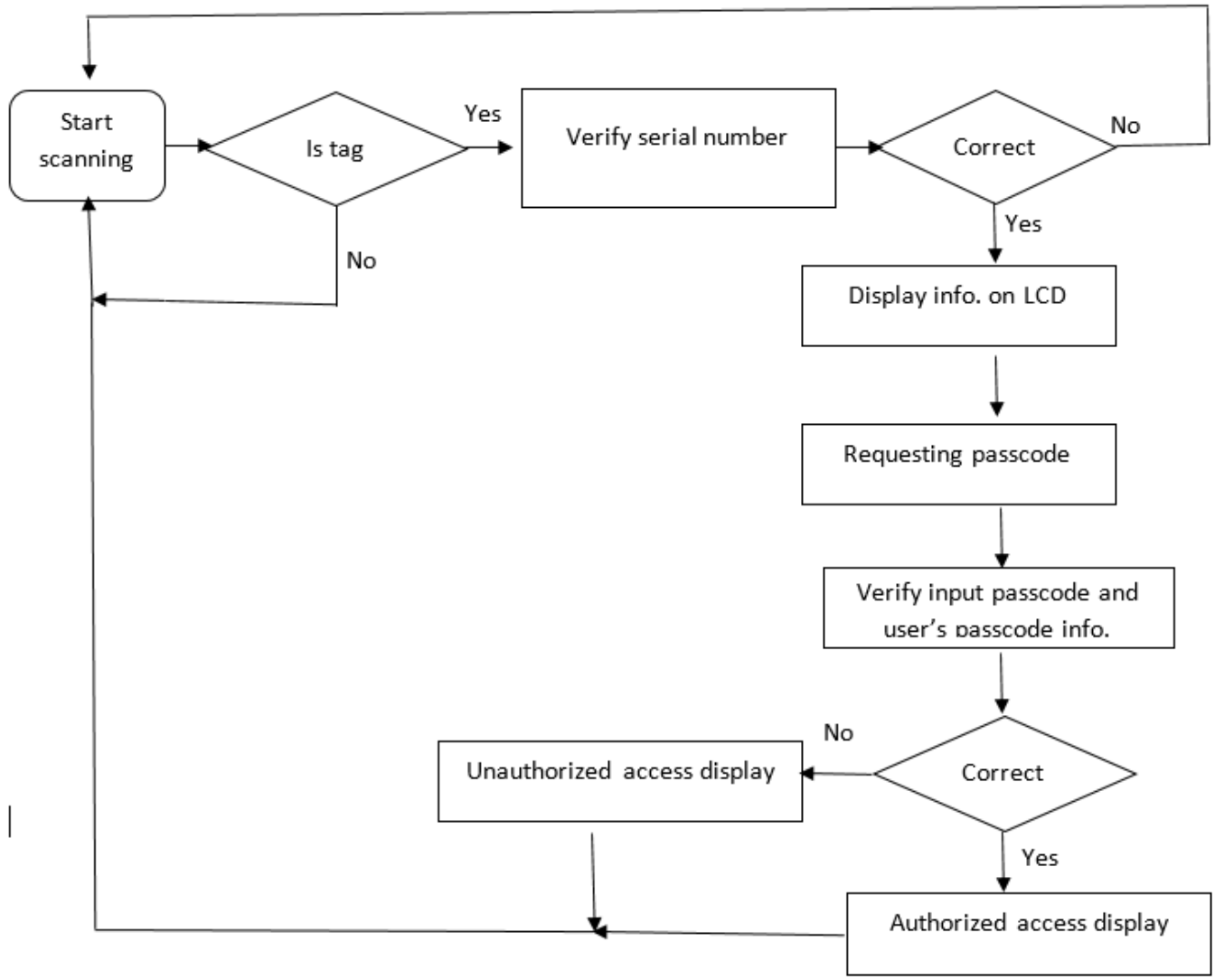


Figure 5: Flow chart of the application code

IV. Experiment

A. Schematic design

From the circuit schematic above, we could draw the circuit by using **ISIS 7 Professional**, electronics emulator supported by Windows platform. This figure below shows the circuit drawing in ISIS 7 Professional.

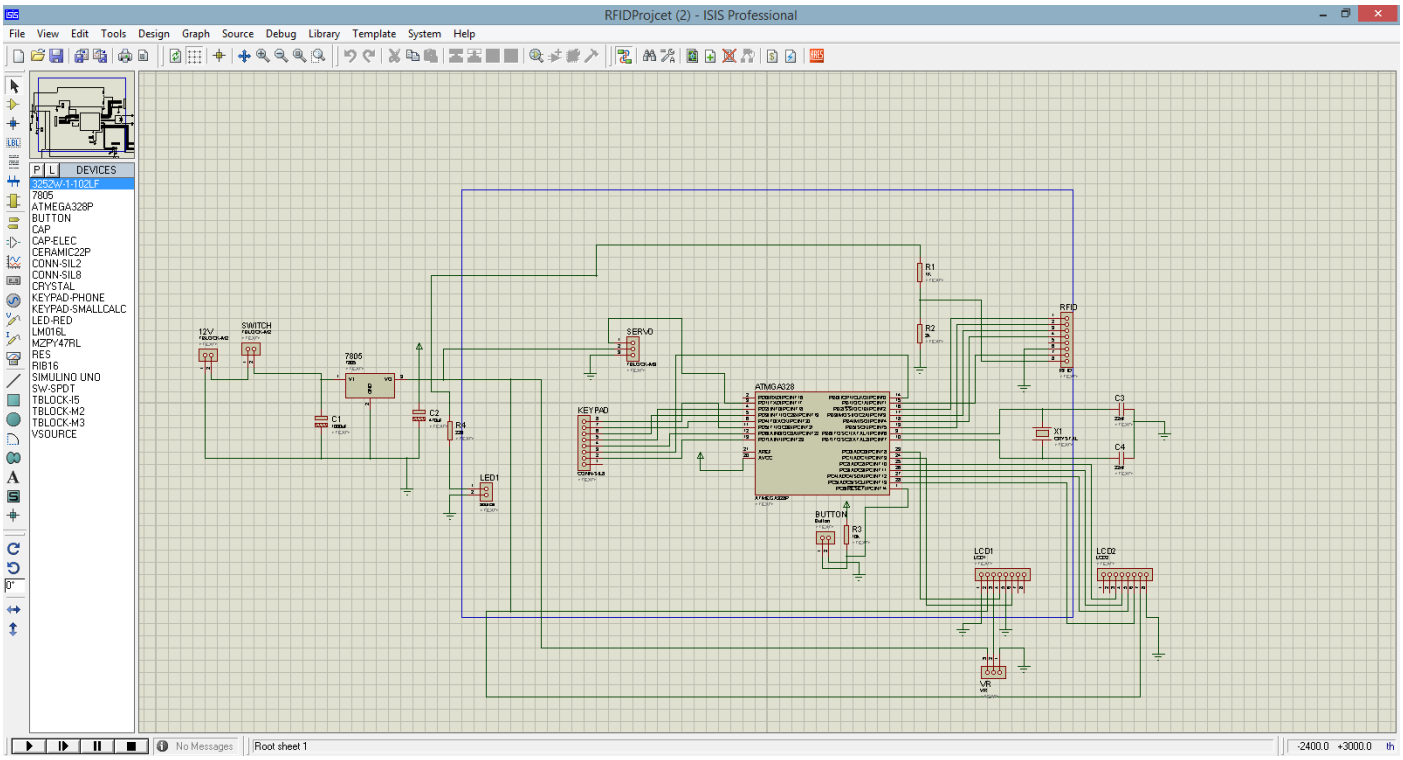


Figure 6: Circuit schematic design in ISIS 7 Professional

B. PCB layout design

After the drawing was finished in ISIS 7 Professional, the printed circuit board(PCB) layout of the circuit could be designed in ARES Professional, an associate software with ISIS 7 Professional. It is shown in the figure below.

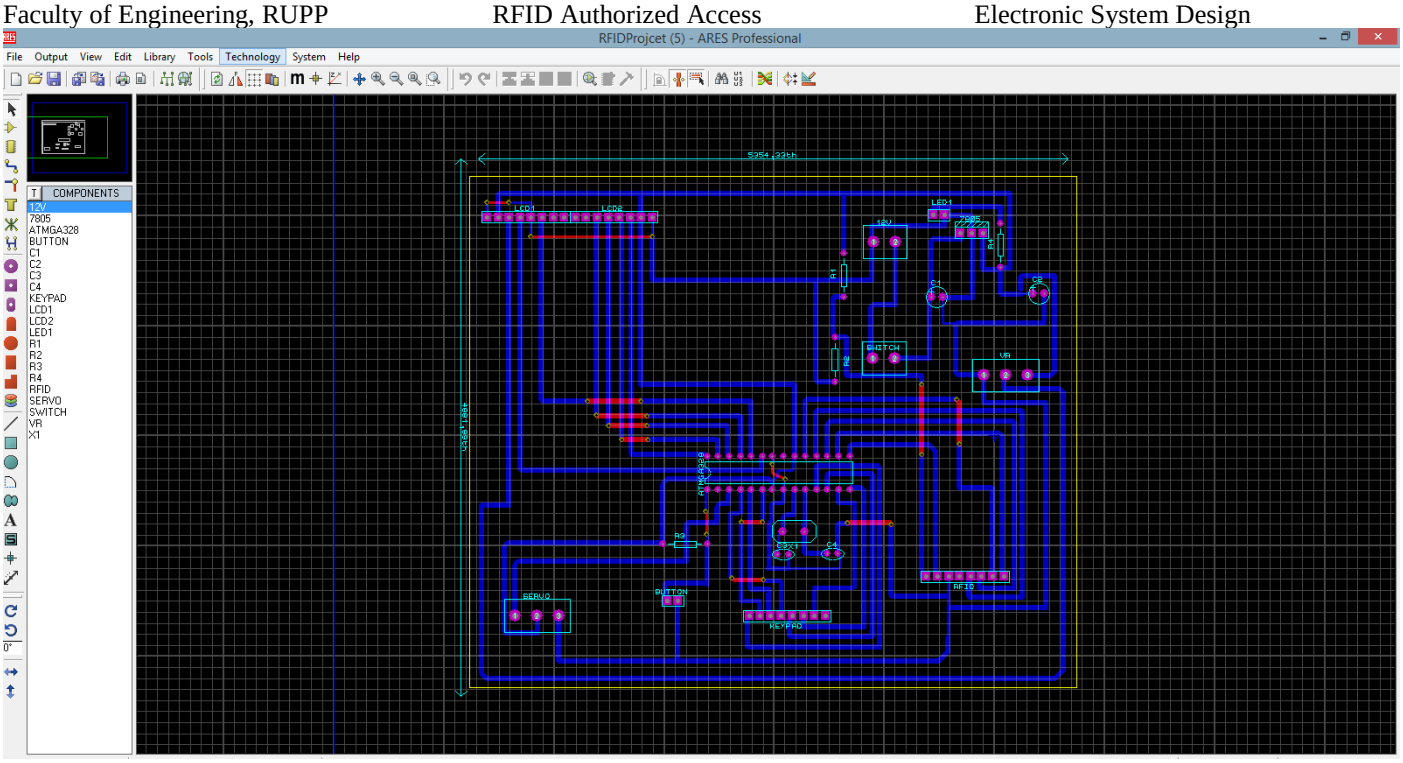


Figure 7: PCB layout design in ARES Professional.

C. Creating the physical PCB

After designing both the schematic and PCB layout, we were able to making the real PCB at home with simple tools. The website www.maker.pro provides a very good explanation of how to make physical PCB at home. The summary steps are shown below.

Step 1: Designing and printing the PCB layout
Step 2: Preparing your copper clad board
Step 3: Transfer of toner onto the board
Step 4: Washing the Board and Completely Removing the Paper
Step 5: The Etching Method
Step 6: Washing the board
Step 7: Remove the Toner, and You Made a PCB
Step 8: Drilling the board
Step 9: Inserting hardware components
Step 10: Soldering the components on the board

D. Coding in Arduino IDE

Since the ATMEGA328 is based on Arduino, thus Arduino IDE is the best way for writing instructions, with C programming language base, for this micro-chip. However, we need to import more libraries listed below to complete the code.

i. Keypad.h to code with keypad module
ii. RFID.h to instruct the RC522 reader
iii. LiquidCrystal.h for instructions of displaying function on the 16x2 LCD

Here is the complete source code of the project.

```
#include <Keypad.h>
#include <RFID.h>
#include <LiquidCrystal.h>
#include <SPI.h>
#include <Servo.h> // reserved

Servo theServo;  // create servo object to control a servo
int pos = 0;    // variable to store the servo position
bool start = true;
void Servo_access()
{
  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
       // in steps of 1 degree
       theServo.write(pos);              // tell servo to go to position in variable 'pos'
       delay(15);                       // waits 15ms for the servo to reach the position
      }
      delay(2000);
      for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
        theServo.write(pos);              // tell servo to go to position in variable 'pos'
        delay(15);                       // waits 15ms for the servo to reach the position
      }

}
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// System DataStructure
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

void PrintInsertCard ( LiquidCrystal lcd)
{
   lcd.begin(16,2);
   lcd.setCursor(0,0);
   lcd.print("Insert card   ");
   lcd.setCursor(0,1);
   lcd.print("               ");

}
void PrintText( LiquidCrystal lcd , String s)
{

   lcd.begin(16,2);
   lcd.setCursor(0,0);
   lcd.print("S");
   lcd.setCursor(0,1);
   lcd.print("               ");
}
```

```
Stuff s1,s2,s3;
  Stuff Company[3]={};


//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++
// RFID feature with LCD display
#define SS_PIN 10
#define RST_PIN 9
RFID rfid(SS_PIN,RST_PIN);

LiquidCrystal lcd (A0, A1, A2, A3, A4, A5); // pins of the LCD. (RS, E, D4, D5, D6, D7)
//String serNum;
int cards[][5] = {
  {53,33,105,208,173},
  {34,95,131,16,238}
};

bool onceAgain = false;
int correct = 0;
int memberID = 10000;
// declare some variable for the Input Password feature
const byte ROWS = 4; //four rows
const byte COLS = 4; //four columns

int count =1;
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};

byte rowPins[ROWS] = {5,4, 3,2}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {6,8, 7};

Keypad customKeypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS);
String pass = "";

void PrintHastag(LiquidCrystal lcd)
{

    lcd.clear();
    lcd.setCursor(0, 0);
      String Hastag = "";
      for (int k=0;k<count;++k)
      {
        Hastag +="#";
      }
      lcd.print(Hastag);
      count ++;
```

```
}

//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++

// Keypad feature
void InputPassword(String P)
{

  char customKey = customKeypad.getKey();
  if (customKey){
    lcd.clear();
    if (customKey == '#') // input
    {

       // compare pass to s.thePass
       if (pass == P)
       {

         lcd.clear();
          lcd.begin(16,2);
         lcd.setCursor(0,0);
         lcd.print("Access Granted");
         Servo_access(); // Access Granted => activate the servo
       }
       else
       {
         lcd.clear();
          lcd.begin(16,2);
         lcd.setCursor(0,0);
         lcd.print("Access Denied");
         delay(2000);

       }
       lcd.clear();
       pass="";
       count =1;
       onceAgain = true;
       return;
    }
    else if (customKey=='*')// clear password
      {
       lcd.clear();
       pass = "";
       count = 1;
    }
    else
      {
       pass = String (pass +customKey);
       // display # on LCD
        PrintHastag(lcd);
```

```
    }
  }// end of if key
}
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
void setup() {
     theServo.attach(1);  // attaches the servo on pin 1 to the servo object
     Serial.begin(9600);
     SPI.begin();
     rfid.init();
     AddStuff();
     lcd.print("Enter ID card   ");

}

void loop() {

  if(rfid.isCard())
     {
          Serial.print("card scanned");

        if(rfid.readCardSerial())
        {
           for(int x = 0; x < sizeof(cards); x++)
           {
             for(int i = 0; i < sizeof(rfid.serNum); i++ )
             {
                if(rfid.serNum[i] == cards[x][i])
                {
                  correct ++;
                }// end of (rfid.serNum[i] == cards[x][i])
                if (correct == 5)
                {
                        lcd.begin(16,2);
                        lcd.setCursor(0,0);
                        lcd.println("Card ID:       ");
                        lcd.setCursor(0,1);
                        memberID =x;
                        Company[x].PrintStuff(Company[x],lcd);
                        delay(500);
                        lcd.clear();
                        correct =0;

                        lcd.setCursor(0, 0);
                        lcd.print("Password");
                        lcd.setCursor(0, 1);
                        lcd.print("# Enter * Clear");
                           while (1)
                           {
                                InputPassword(Company[x].thePass);
                                if (onceAgain == true)
```

```
                                {
                                  onceAgain = false;
                                  break;
                                }
                            }
                        PrintEnterCard(lcd);
                        rfid.halt();
                        return;
                    }// end of correct ==5
                }// end of for(int i = 0; i < sizeof(rfid.serNum); i++ )
            }// end of for(int x = 0; x < sizeof(cards); x++)

                // otherwise it's wrong card
                lcd.begin(16,2);
                lcd.setCursor(0,0);
                lcd.print(" Not Allowed");
                delay(1000);
                lcd.clear();

        }
                PrintEnterCard(lcd);
        }// end of if(rfid.isCard())

    rfid.halt();

}// end of loop()
```

E. Uploading code to ATMEGA328

After finishing coding, now let us upload code to the micro-controller. From www.arduino.cc, we can find a very good resource of how to upload code to this micro-chip.

V. Result and Demo

A. PCB of the project

These following figures show result the project. The circuit needs 12volt power supply.

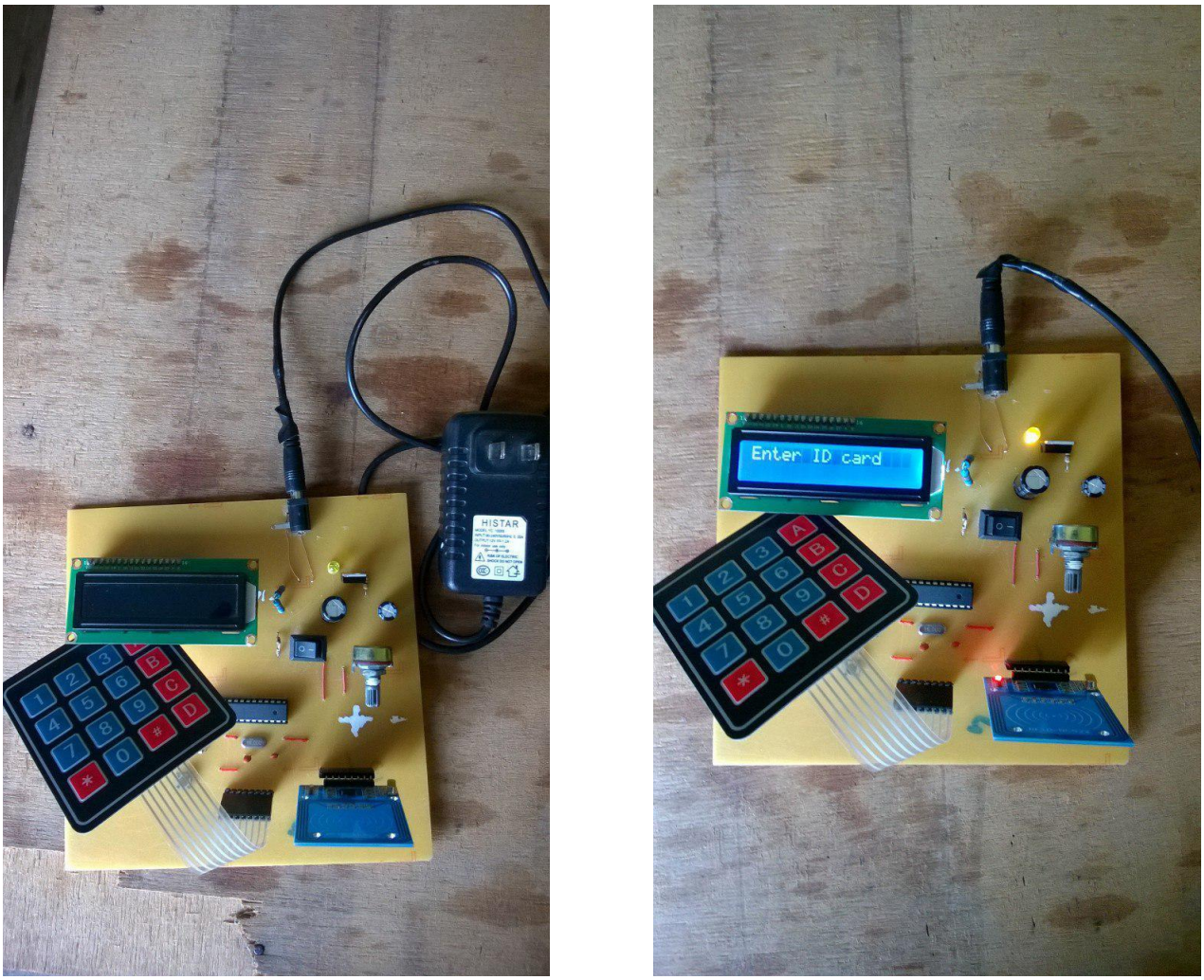


Figure 8: (a) the circuit board without case (b) the circuit board without case but with power supply

B. Final product of the project

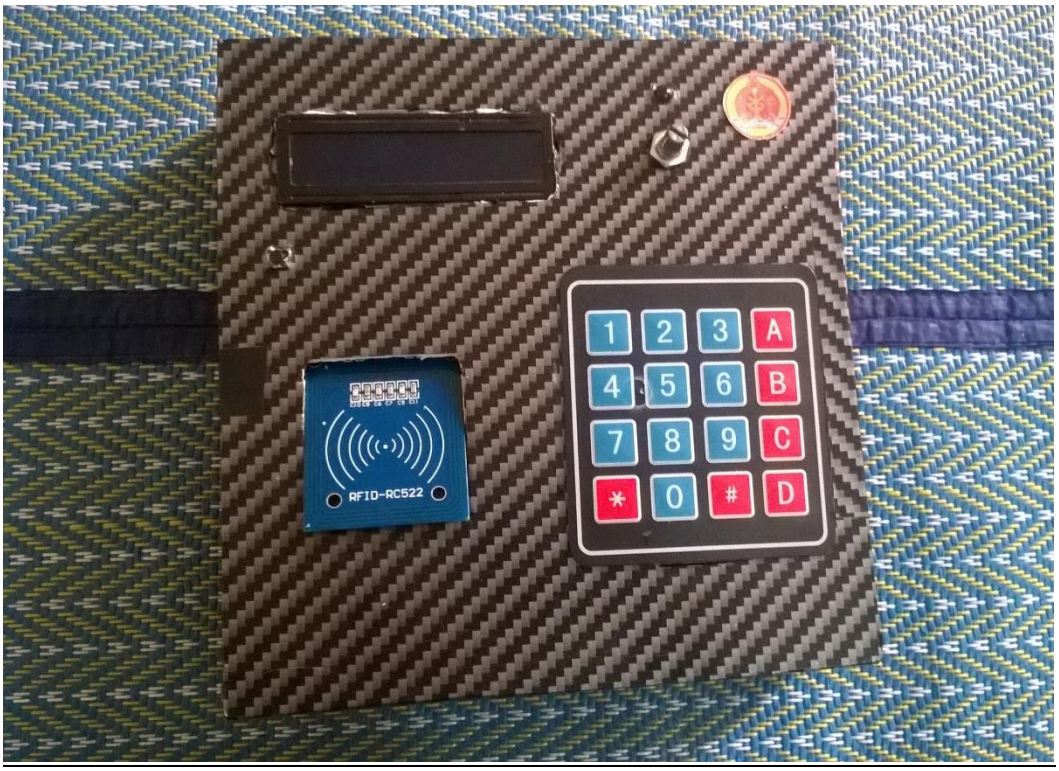


Figure 9: Final product of this project

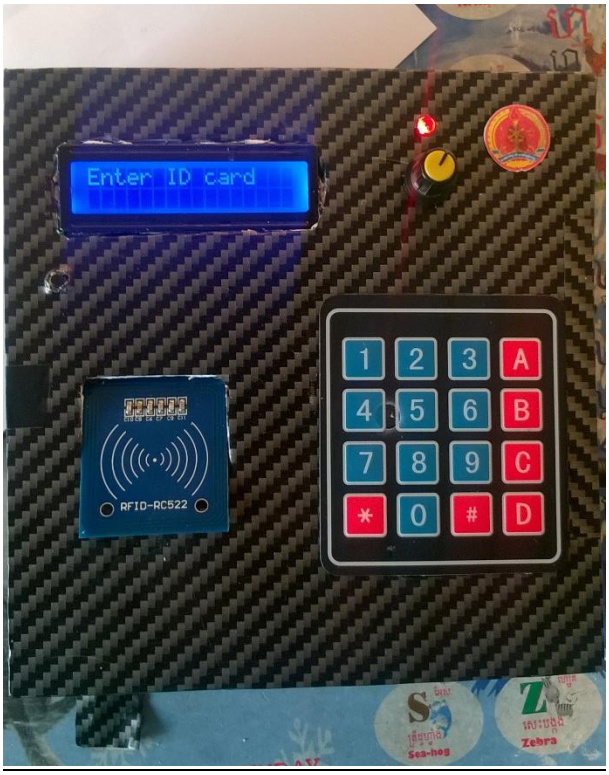


Figure 10: the product at operation

C. Demo

For further information of the product's operation, please go to this link below in Youtube.

RFID Authorized Access demo: https://www.youtube.com/watch?v=YfOiHcA_ld0

VII. Conclusion

RFID AA offers two features of authorized verifications described above. It contributes to engineers who wish to integrate this product into other applications for example doors' locking/unlocking.
This product is able to integrate with applications such as:

i. Record attendance of staffs or students etc.
ii. Home automation.
iii. Door automatic lock/unlock
iv. Smart wallet
v. Materials management

References

[1] How RFID Works and How to Make an Arduino based RFID Door Lock. Available at http://howtomechatronics.com/tutorials/arduino/rfid-works-make-arduino-based-rfid-door-lock/
[2] Product data sheet COMPANY PUBLIC, MFRC522 Standard performance MIFARE and NTAG frontend, Rev. 3.9, 27 April 2016
[3] RFID. Available at https://en.wikipedia.org/wiki/Radio-frequency_identification http://www.instructables.com/id/Arduino-RFID-Reader-MFRC522-Turorial/
[4] 13.56MHZ RFID/NFC CARD. Available at https://www.adafruit.com/product/359
[5] How to make a PCB. Available online at https://maker.pro/education/how-to-design-pcbs
[6] From Arduino to a Microcontroller on a Breadboard. Available online at https://www.arduino.cc/en/Tutorial/ArduinoToBreadboard
[7] Voltage divider. Available online at https://learn.sparkfun.com/tutorials/voltage-dividers