North South University

Department of Electrical and Computer Engineering

Course Title: CSE331 – Microprocessor Interfacing and Embedded System

Semester: Spring 2020

Team Project

Group: 5

Faculty Advisor:

Dr. Dihan Md. Nuruddin Hasan

Assistant Professor

Department of Electrical and Computer Engineering

North South University

ACKNOWLEDGMENT

We would like start off by thanking the Almighty for giving us the strength to complete the project and the report. We are also deeply grateful to all of our teachers who have supported us and provided us the knowledge to carry out this project.

It is imperative to show our appreciation for our honorable faculty member Dr. Dihan Md. Nuruddin Hasan who has given us proper guidelines to prepare the project. His undivided attention and advices have helped us to achieve this milestone. We are also very grateful to the department of Electrical and Computer Engineering of North South University for providing us with a course like CSE 331 where we learn about microprocessor and implement our knowledge by working on a project like this.

We thank our friends and family for their moral support throughout the project.

ABSTRACT

This report deals with the implementation of the following encryption table using microcontroller. We used single pole, double throw switch to configure the high and low conditions. LEDs were used to represent the corresponding output statuses.

Input				Output			
I3	**I2**	**I1**	**I0**	**O3**	**O2**	**O1**	**O0**
0	0	0	0	0	1	0	1
1	0	0	0	1	1	0	0
0	1	0	0	1	1	1	0
1	1	0	0	0	0	0	1
0	0	1	0	0	1	1	1
1	0	1	0	0	0	1	0
0	1	1	0	0	1	1	0
1	1	1	0	0	1	0	0
0	0	0	1	1	1	0	0
1	0	0	1	0	0	0	0
0	1	0	1	0	1	1	0
1	1	0	1	1	0	1	1
0	0	1	1	0	0	0	0

1	0	1	1	0	0	1	1
0	1	1	1	1	0	1	1
1	1	1	1	0	0	1	1

Project Description:

Encryption is a method of converting information into a secret code that requires a “key” or password to decode. This process is used by many websites that transmit credit card and bank account numbers to prevent identity theft and fraud.

In this project we input a 4-bit data and the microcontroller converts it to a 4-bit encrypted data. The output corresponds to the input given in the table. 0’s are considered low and 1’s are considered high. Single pole and double throw switch are used to configure high and low conditions of the inputs. LEDs are used to denote the output. If the LED lights up its high (1) and if it is turned off then it’s low (0).

Software Used:

1. Arduino IDE: The Arduino Integrated Development Environment (IDE) is a cross-platform application for Windows, macOS and Linux. It supports the languages C and C++ with special rules of code structuring. The program written with Arduino IDE is known as sketch. Arduino IDE is an open-source software that is mainly used for writing and compiling the code into the Arduino Module. It is an official Arduino software. The IDE environment mainly has the editor where the program is written and the compiler which compiles and uploads the file onto the Arduino model.

2. Proteus: Proteus is a proprietary software tool that is used for electronic design automation. It is mainly used to create schematics and electronic prints along with simulation of electronic designs.

Flow Chart:

Figure 1: Simple Flow Chart of the Encryption Process

Method of Derivation:

In order to find out the required algorithm we use K-map to derive the Boolean expressions. K-maps or Karnaugh maps are used to simplify Boolean algebraic functions. There are two forms of K-map- (i) Sum of Products (SOP) and (ii) Product of Sums (POS). Either can be used according to the problem. K-map is a table like representation where the grids are filled with 0's and 1's according to the given problem and solved by making groups. For our project we are using SOP to find out when the output is high (1).

Output 1:

CD \ AB	00	01	11	10
00	1	0	1	0
01	1	0	0	0
11	0	1	1	1
10	0	0	1	0

Equation: $I_0'I_2{\cdot}I_3' + I_1'I_2I_3 + I_0I_1I_3 + I_0I_1I_2$

Output 2:

CD \ AB	00	01	11	10
00	0	0	0	1
01	1	1	0	1
11	0	1	1	1
10	0	0	1	1

Equation: $I_2I_3' + I_0I_2 + I_0'I_1I_2' + I_1I_2'I_3$

Output 3:

CD \ AB	00	01	11	10
00	1	1	0	1
01	1	0	1	1
11	0	0	0	0
10	1	0	0	1

Equation: $I_0'I_3' + I_1'I_3' + I_0'I_1'I_3' + I_0'I_1I_2$

Output 4:

CD \ AB	00	01	11	10
00	0	1	0	1
01	0	0	0	0
11	0	0	0	1
10	1	0	1	0

Equation: $I_0'I_1'I_2'I_3 + I_0'I_1'I_2I_3' + I_0I_1'I_2'I_3' + I_0I_1'I_2I_3 + I_0I_1I_2I_3'$

Result:

The obtained Boolean expressions gives us the following results in C/C++ language-

Output 1: (!I_0 && !I_2 && !I_3') || (!I_1 && I_2 && I_3) || (I0 && I1 && I3) || (I0 && I1 && I2)

Output 2: (I_2 && !I_3) || (I_0 && I_2) || (!I_0 && I_1 && !I_2) || (I_1 && !I_2 && I_3)

Output 3: (!I_0 && !I_3) || (!I_1 && !I_3) || (!I_0 && !I_1 && !I_3) || (!I_0 && I_1 && I_2)

Output 4: (!I_0 && !I_1 && !I_2 && I_3) || (!I_0 && !I_1 && I_2 && I_3) || (I_0 && !I_1 && I_2 && I_3) || (I_0 && I_1 && I_2 && !I_3)

Circuit Diagram:

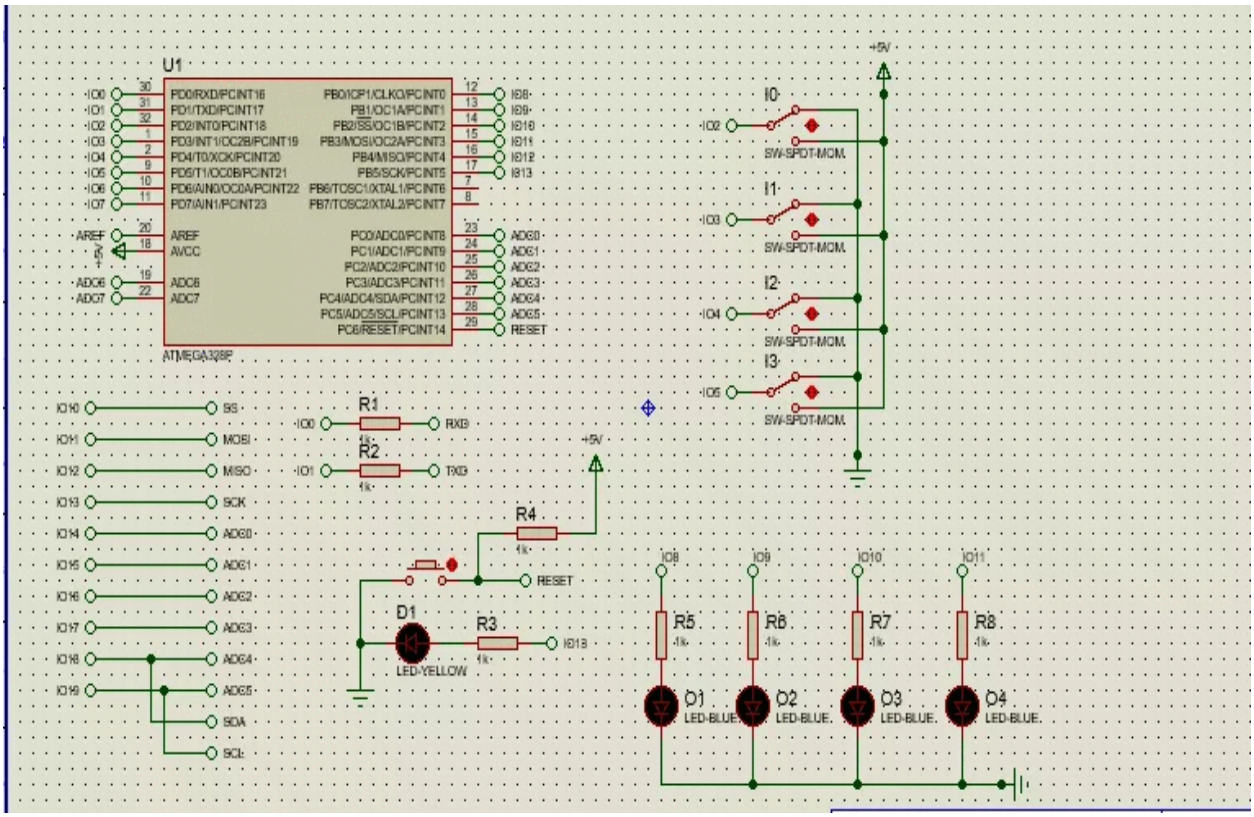

Figure 2: Schematic Drawing in Proteus

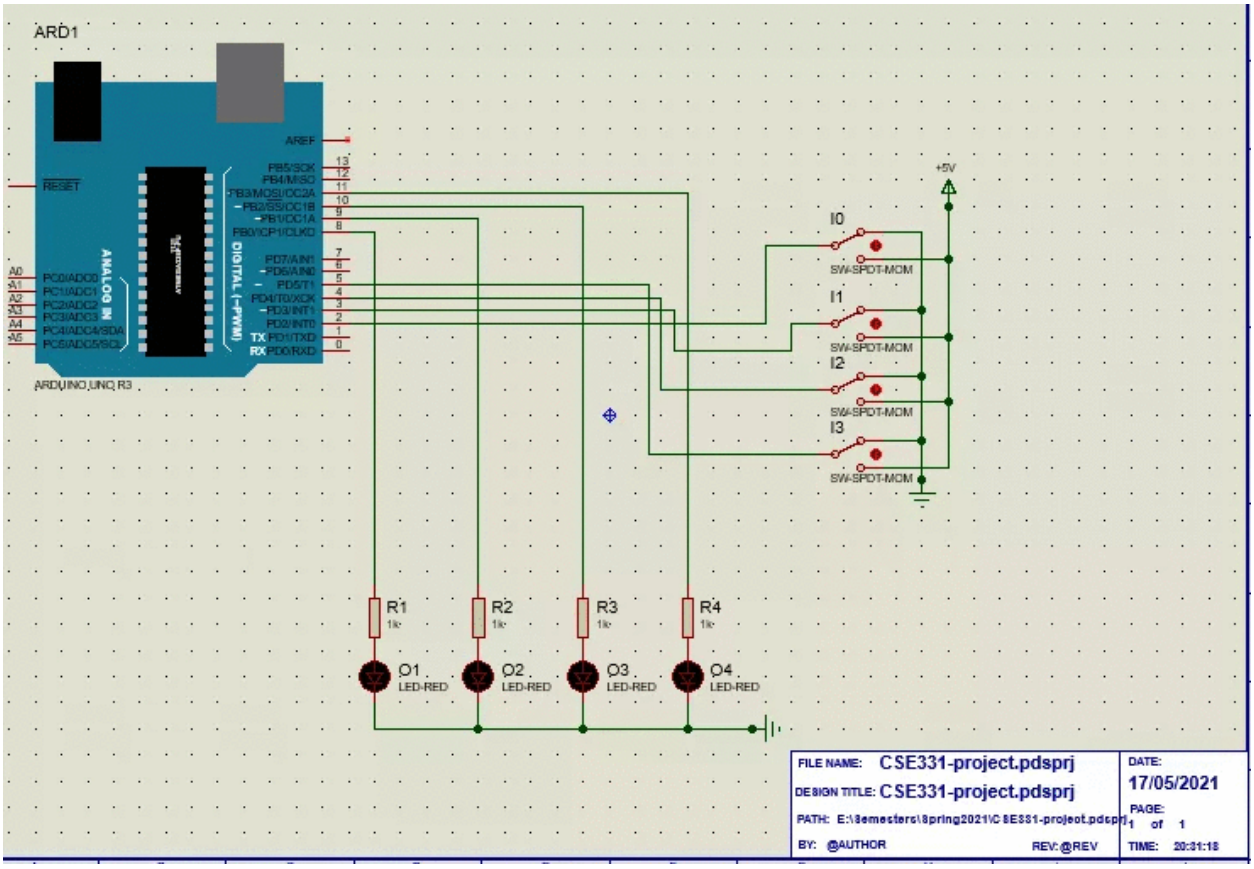

Figure 3: Arduino Uno top view of the Schematics

Code:

Compilation of Code: We used the latest version of Arduino IDE to write our embedded C code and generate the hex file.

EMBEDDED C CODE:

```
const int StateI0 = 2;

const int StateI1 = 3;

const int StateI2 = 4;

const int StateI3 = 5;
```

```
const int O1 = 8;

const int O2 = 9;

const int O3 = 10;

const int O4 = 11;


int I0 = 0;

int I1 = 0;

int I2 = 0;

int I3 = 0;


void setup() {

  pinMode(StateI0, INPUT);

  pinMode(StateI1, INPUT);

  pinMode(StateI2, INPUT);

  pinMode(StateI3, INPUT);

  pinMode(O1, OUTPUT);

  pinMode(O2, OUTPUT);

  pinMode(O3, OUTPUT);

  pinMode(O4, OUTPUT);

}
```

```
void loop() {
 I0 = digitalRead(StateI0);
 I1 = digitalRead(StateI1);
 I2 = digitalRead(StateI2);
 I3 = digitalRead(StateI3);

  if( (!I0 && !I2 && !I3) || (!I1 && I2 && I3) || (I0 && I1 && I3) || (I0 && I1 && I2) ){
    digitalWrite(O1, HIGH);
  }
  else{
    digitalWrite(O1, LOW);
  }

  if( (I2 && !I3) || (I0 && I2) || (!I0 && I1 && !I2) || (I1 && !I2 && I3) ){
    digitalWrite(O2, HIGH);
  }
  else{
    digitalWrite(O2, LOW);
  }

  if( (!I0 && !I3) || (!I1 && !3) || ( !I0 && !I1 && !I3) || ( !I0 && I1 && I2) ){
```

```
    digitalWrite(O3, HIGH);
  }
  else{
    digitalWrite(O3, LOW);
  }

  if( ( !I0 && !I1 && !I2 && I3) || ( !I0 && !I1 && I2 && !I3) || ( I0 && !I1 && !I2 && !I3) || (I0
&& !I1&& I2 && I3) || ( I0 && I1 && I2 && !I3) ){
    digitalWrite(O4, HIGH);
  }
  else{
    digitalWrite(O4, LOW);
  }

}
```

File Edit Sketch Tools Help

cse331

```
const int StateI0 = 2;
const int StateI1 = 3;
const int StateI2 = 4;
const int StateI3 = 5;

const int O1 = 8;
const int O2 = 9;
const int O3 = 10;
const int O4 = 11;

int I0 = 0;
int I1 = 0;
int I2 = 0;
int I3 = 0;

void setup() {
  pinMode(StateI0, INPUT);
  pinMode(StateI1, INPUT);
  pinMode(StateI2, INPUT);
  pinMode(StateI3, INPUT);
  pinMode(O1, OUTPUT);
  pinMode(O2, OUTPUT);
  pinMode(O3, OUTPUT);
  pinMode(O4, OUTPUT);
}

void loop() {
 I0 = digitalRead(StateI0);
```

```
void loop() {
 I0 = digitalRead(StateI0);
 I1 = digitalRead(StateI1);
 I2 = digitalRead(StateI2);
 I3 = digitalRead(StateI3);

  if( (!I0 && !I2 && !I3) || (!I1 && I2 && I3) || (I0 && I1 && I3) || (I0 && I1 && I2) ){
    digitalWrite(01, HIGH);
  }
  else{
    digitalWrite(01, LOW);
  }

  if( (I2 && !I3) || (I0 && I2) || (!I0 && I1 && !I2) || (I1 && !I2 && I3) ){
    digitalWrite(02, HIGH);
  }
  else{
    digitalWrite(02, LOW);
  }

  if( (!I0 && !I3) || (!I1 && !3) || ( !I0 && !I1 && !I3) || ( !I0 && I1 && I2) ){
    digitalWrite(03, HIGH);
  }
  else{
    digitalWrite(03, LOW);
  }

  if( ( !I0 && !I1 && !I2 && I3) || ( !I0 && !I1 && I2 && !I3) || ( I0 && !I1 && !I2 && !I3) || (I0 && !I1&& I2 && I3) || ( I0 && I1 && I2 && !I3) ){
    digitalWrite(04, HIGH);
```

```
Compiling sketch...
Linking everything together...
"C:\\Program Files\\WindowsApps\\ArduinoLLC.ArduinoIDE_1.8.49.0_x86__mdqgnx93n4wtt\\hardware\\tools\\avr/bin/avr-gcc" -w -Os -g -flto -fuse-linker-plugin -
```

Figure 4: Code in Arduino IDE

Click this to view "Hex File"

Simulation:

We used Proteus version 8.12 professional to carry out our simulation of arduino uno. We simply imported the Atmega328p board in the workspace and connected the needed peripherals by importing them in similar manner. After that we flashed the Hex file onto the board and carried out our simulation.

A screenshot taken during simulation of our project:

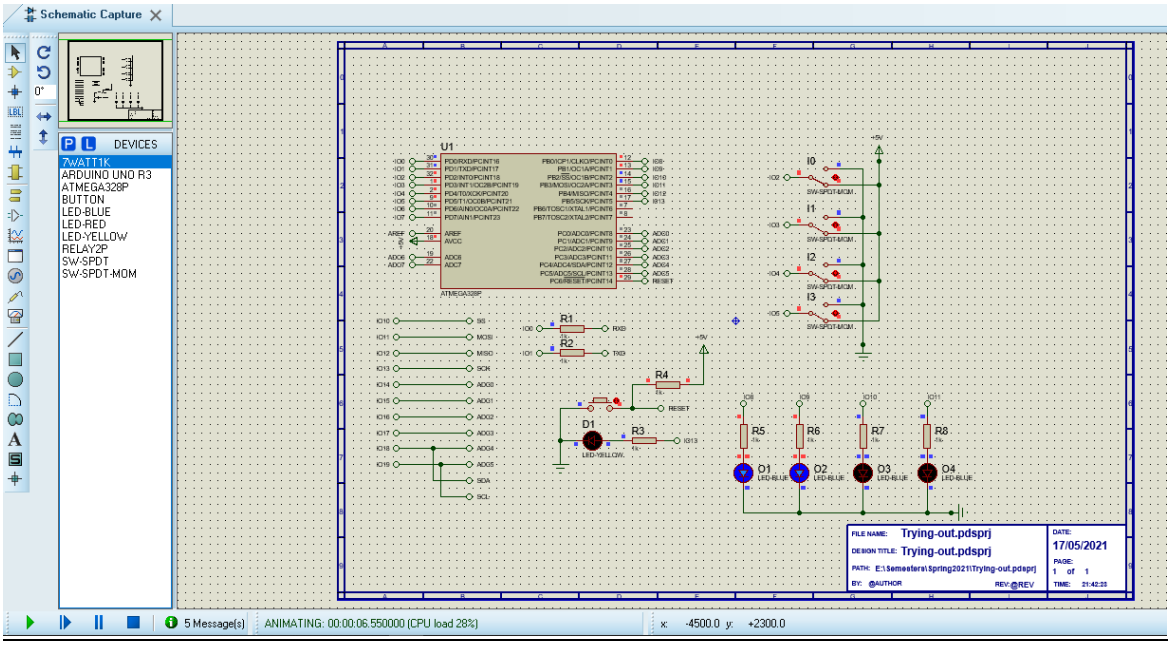

Figure 5: Simulation in Proteus 8.12

Questions:

1. What is the clock frequency of the microcontroller used?

Answer: The microcontroller used is Arduino Uno R3 and it's clock speed is 16Mhz.

2. What is the data bus width of the microcontroller used?

Answer: Arduino Uno is built on AVR architecture so it has a 8 bit data bus.

3. What is the size of the hex file generated? Attach the hex code.

Answer: The size of generated hex file is 3.38 KB.

4. Can the project be implemented using interrupt?

Answer: Yes, but in that case the input and output will be frizzed, and no new data can be process.

5. Is the main routine required to be an infinite loop?

Answer: Yes, because the circuit takes 4 bit input data and encrypts them to 4 bit output data. So the program needs to run continuously in a loop. Without loop the program will be stuck when one time a set of data is given, and could not be used again. That why it required an infinite loop.

6. Is there any difference between level triggered and edge triggered operation for the given project?

Answer: In level triggering the circuit will become active when the gating or clock pulse is on a particular level. In edge triggering the circuit becomes active at negative or positive edge of the clock signal So that the edge trigger won't work with this project.

7. Is the project referring encryption or decryption from input to output?

Answer: Yes, The project is working perfectly for encryption or decryption from input to output as shown in the truth table.