North South University

Department of Electrical & Computer Engineering

Project Report

CSE331

Section: 01

Group: 03

Submission Date: 30.12.2022

Submitted to: Dr. Dihan Md. Nuruddin Hasan (DMH)

Group Member

Md Ulfat Tahsin (1913057042)

Md Saeem Hossain Shanto (1912218042)

Sanjida Islam (1831038642)

Table of Contents

Abstract

We were given an encryption table for this project and had to construct it using a microcontroller. We were also instructed to set up the inputs for high and low circumstances using a single pole and double-throw switch. In addition, we were obliged to employ LEDs to symbolize the table's matching output.

The given encryption table:

Input				**Output**			
I_3	I_2	I_1	I_0	O_3	O_2	O_1	O_0
0	0	0	0	0	1	0	1
1	0	0	0	0	0	1	1
0	1	0	0	1	0	1	0
1	1	0	0	0	1	1	0
0	0	1	0	1	0	1	1
1	0	1	0	1	1	0	0
O	1	1	0	0	1	1	0
1	1	1	0	1	1	0	0
0	0	0	1	0	0	0	1

Input				Output			
1	0	0	1	0	0	1	1
0	1	0	1	1	0	1	1
1	1	0	1	1	1	0	0
0	0	1	1	1	0	1	0
1	0	1	1	0	1	1	1
0	1	1	1	0	0	1	0
1	1	1	1	1	1	1	0

General Description:

2.1 Arduino UNO

The Arduino UNO is the best board to get started with electronics and coding. If this is one's first experience tinkering with the platform, the UNO is the most robust board one can start playing with. The UNO is the most used and documented board of the whole Arduino family.

Arduino UNO is a microcontroller board based on the **ATmega328P**. It has 14 digital input/output pins (of which six can be used as PWM outputs), six analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started. You can tinker with your UNO without worrying too much about doing something wrong; in the worst-case scenario, you can replace the chip for a few dollars and start over again.

ATmega328P: The classic high-performance, low-power AVR microcontroller.

Replaceable chip: The ATmega328P can easily be replaced, as it is not soldered to the board.

EEPROM: The ATmega328P also features 1kb of EEPROM, a memory that is not erased when powered off.

Battery Connector: The Arduino UNO features a barrel plug connector that works great with a standard 9V battery.

It doesn't use standard USB for connection with a computer. Instead, it comes with a type B Micro USB. Flash memory is 16KB or 32KB that all depending on the Atmega board. Atmega168 comes with 16KB of flash memory, while Atmega328 comes with a flash memory of 32 KB. Flash memory is used for storing code. The 2KB of memory out of total flash memory is used for a bootloader. The SRAM memory of 2KB is present in Arduino Nano. It has an EEPROM memory of 1KB.

2.2 Arduino IDE

The software used for writing, compiling & uploading code to Arduino boards is called Arduino IDE (Integrated Development Environment). It is a cross-platform software that is available for every Operating System like Windows, Linux, and macOS. This software can be used with any Arduino board. We write our code in this compiler and then connect the microcontroller via USB to upload the code in it. Arduino IDE consists of different sections such as WindowBar, MenuBar, ShortcutButtons, Text Editor, and Output Panel.

2.3 Proteus 8 PRO

Proteus 8 Professional is software that can be used to draw schematics, PCB layout, code, and even simulate the schematic. The Proteus Design Suite is a software suite containing schematic, simulation as well as PCB designing. Schematic Capture in the Proteus Design Suite is used for both the simulation of designs and as the design phase of a PCB layout project. It is, therefore, a core component and is included in all product configurations.

We simulate our circuit in Proteus 8 Pro using Arduino Uno, resistors, single pole double throw switches, and LED lights.

2.4 Logisim

Logisim is a logic simulator that permits circuits to be designed and simulated using a graphical user interface.

It is free and open-source software compatible with multiple platforms like Windows, macOS, and Linux. The ease of Logisim comes from its interactive graphical user interface developed with the Java Swing GUI library. Logisim enables users to create circuits of any scale, including sub-circuits as well as complex structures. Color-coded wires impart clarity to the representation. Logisim lets a user simulate his logical circuits and edit circuits while the simulation is running. Therefore, a user can immediately rectify flaws without the need for separate edits and loading.

Equipment

We used a variety of software and hardware to finish the projects, those are

1. Logisim

2. Arduino Uno

3. Proteus 8 Pro

4. 4-bit SPDT switch

5. 4 x LED light

6. Jumper Wire

7. Breadboard

8. 4 x 10k ohm resistor

Method of Derivation

Because we already had the encryption table, the first step in the derivation process was to identify the right Boolean expressions for each of the outputs. Both the input and output are 4-bits long because we're encrypting 4-bit data. A separate Boolean expression must represent each bit in the output. Because there are four inputs, one for each input bit, we may use a four-variable K-map to obtain simplified Boolean expressions for each output bit.

4.1 Truth Table:

Input				**Output**			
I_3	I_2	I_1	I_0	O_3	O_2	O_1	O_0
0	0	0	0	0	1	0	1
1	0	0	0	0	0	1	1
0	1	0	0	1	0	1	0
1	1	0	0	0	1	1	0
0	0	1	0	1	0	1	1
1	0	1	0	1	1	0	0
O	1	1	0	0	1	1	0
1	1	1	0	1	1	0	0
0	0	0	1	0	0	0	1
1	0	0	1	0	0	1	1

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

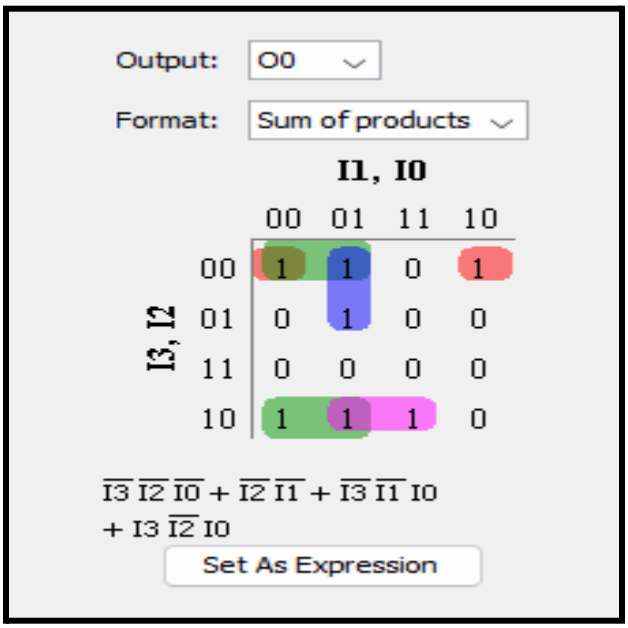
4.2 **Derived Result:**

We use the K-Map to derive the expression for the output.

KMAP:

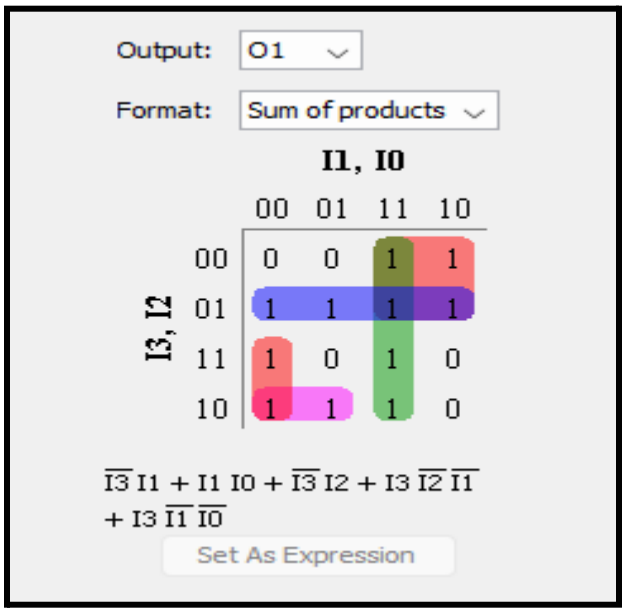
Below are the K-maps for the output of our encryption table, where A, B, C, and D correspond to inputs I_0, I_1, I_2, and I_3, respectively.

For output O_0:



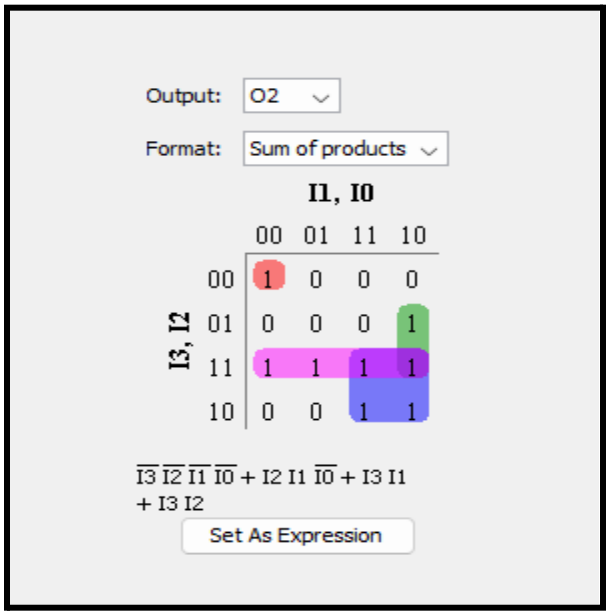
$O_0 = I_3'I_2'I_0' + I_2'I_1' + I_3\,I_2'\,I_0$

For output O_1:



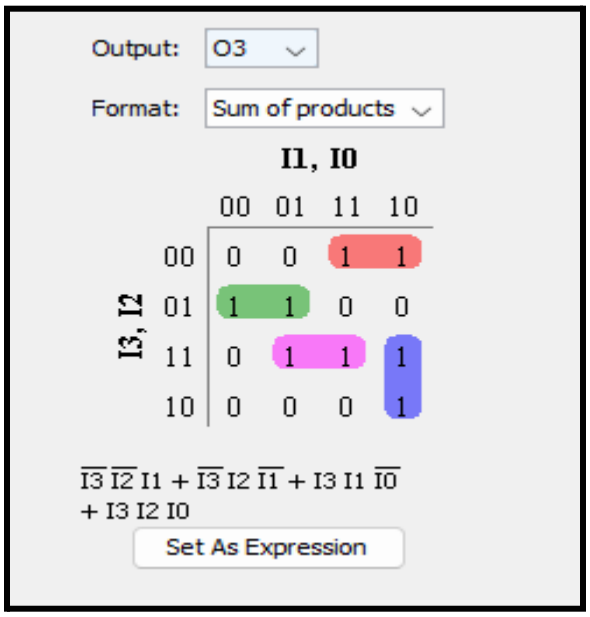
$O_1 = I_3' I_1 + I_1I_0 + I_3I_2 + I_3 I_2' I_1' + I_3 I_1' I_0'$

For output O_2:



$O_2 = I_3'I_2'I_1'I_0 + I_2I_1I_0' + I_3I_1 + I_3I_2$

For output O_3:



$O_3 = I_3'I_2'I_1 + I_3'I_2I_1' + I_3I_1I_0' + I_3I_2I_0$

After using the KMAP we found the following expressions:

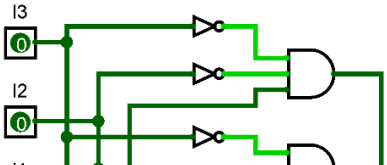
O_0 $= I_3'I_2'I_0' + I_2'I_1' + I_3\,I_2'\,I_0$

O_1 $= I_3'\,I_1 + I_1I_0 + I_3I_2 + I_3\,I_2'\,I_1' + I_3\,I_1'\,I_0'$

O_2 $= I_3'I_2'I_1'I_0 + I_2I_1I_0' + I_3I_1 + I_3I_2$

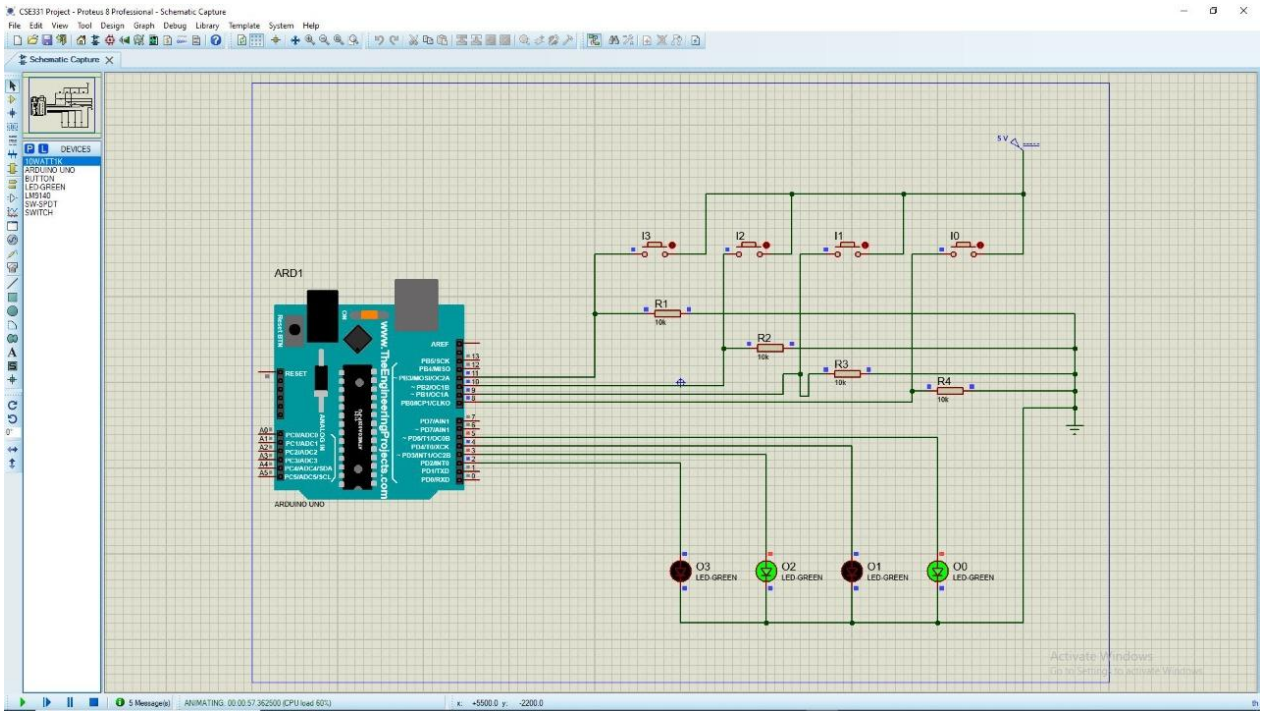
O_3 $= I_3'I_2'I_1 + I_3'I_2I_1' + I_3I_1I_0' + I_3I_2I_0$

Logical Circuit Diagram



Circuit Diagram with Values of Electrical Components

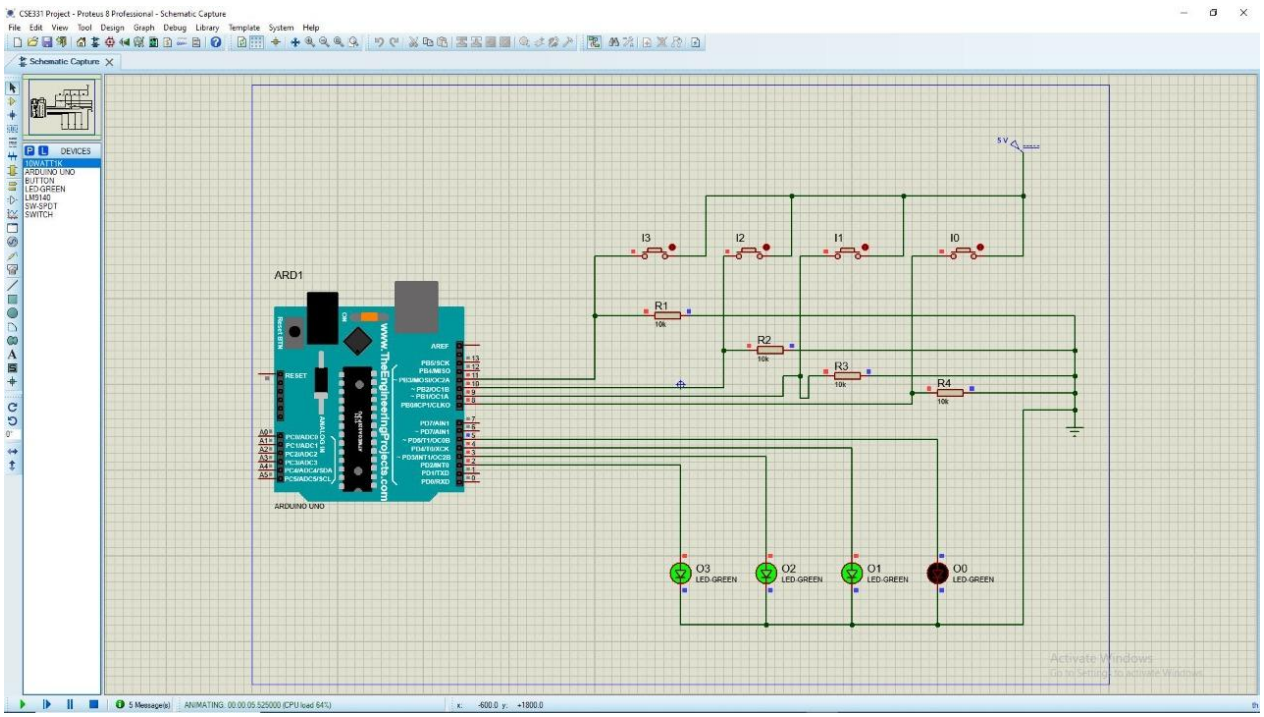
5.1 Figure 1:



Here we can see the circuit in Proteus 8,

for the inputs are $\mathbf{I_3=0, I_2=0, I_1=0, I_0=0}$ the LED outputs are $\mathbf{O_3=0, O_2=1, O_1=0, O_0=1}$.

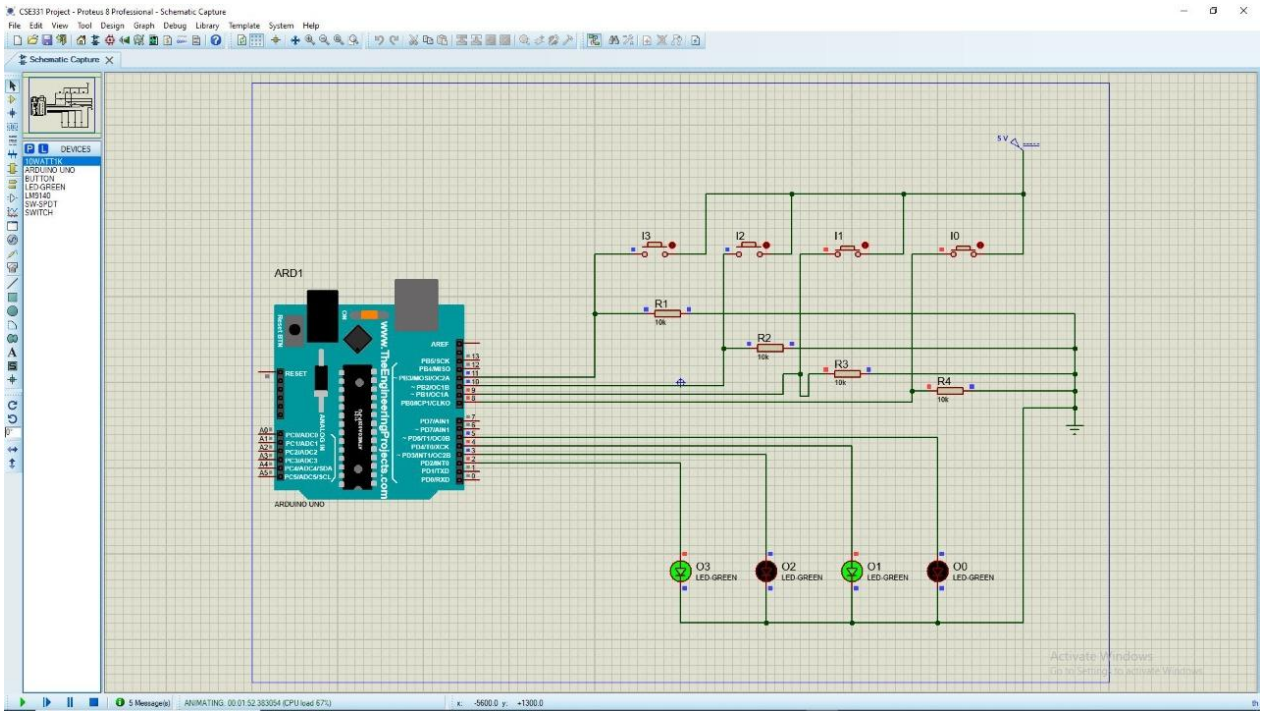
5.2 Figure 2:



Here we can see the circuit in Proteus 8,

for the inputs are $\mathbf{I_3=1, I_2=1, I_1=1, I_0=1}$ the LED outputs are $\mathbf{O_3=1, O_2=1, O_1=1, O_0=0}$.

5.3 Figure 3:



Here we can see the circuit in Proteus 8,

for the inputs are **I_3=0, I_2=0, I_1=1, I_0=1** the LED outputs are **O_3=1, O_2=0, O_1=1, O_0=0**.

Circuit Operation Principles:

We get our output functions using the K-map and the equations are,

$\mathbf{O_0}= I_3'I_2'I_0' + I_2'I_1' + I_3 I_2' I_0$

$\mathbf{O_1} = I_3' I_1 + I_1I_0 + I_3I_2 + I_3 I_2' I_1' + I_3 I_1' I_0'$

$\mathbf{O_2} = I_3'I_2'I_1'I_0 + I_2I_1I_0' + I_3I_1 + I_3I_2$

$\mathbf{O_3}= I_3'I_2'I_1 + I_3'I_2I_1' + I_3I_1I_0' + I_3I_2I_0$

The next step is to build the circuit using logic gates and implement them on logisim. Then we used Proteus 8 pro to build the hardware circuit diagram, the components used in proteus are **Arduino Uno**, this is the microcontroller we usedin the simulation.

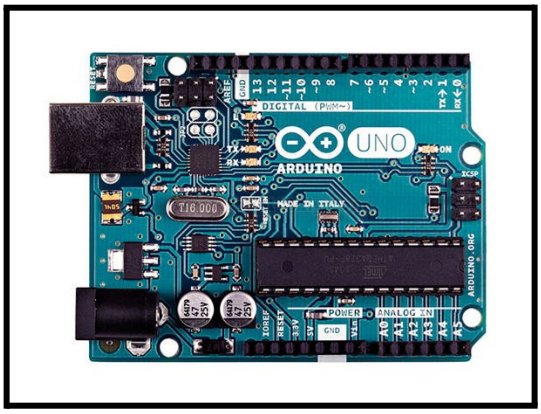
W 

Figure: Arduino UNO

Push Switch:

A push button switch is a mechanical device that uses physical button pressing to activate an internal switching mechanism to regulate an electrical circuit. Depending on the needs of the design, they are available in a number of forms, dimensions, and arrangements. The push switch we used has two pins. One pin is connected to the Arduino output pin and the left one is connected to the other switches. Each switch is connected to the other and individually connected to Arduino input pins 2,3,4,5, respectively.

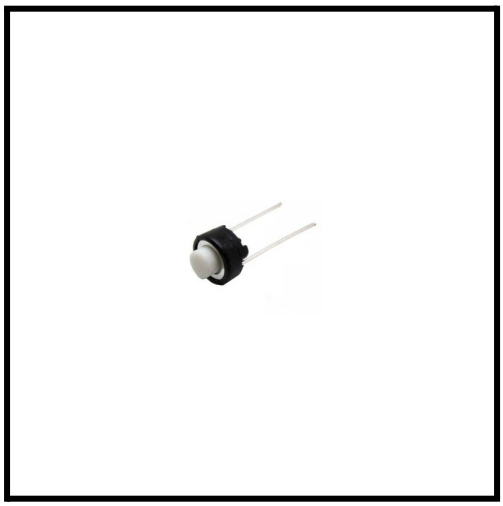


Figure: Switch

Resistors:

We used 4 10K -ohm resistors to connect with the switches and connect with the LED outputs.

Figure: Resistor

LEDs:

We connected 4 LED outputs O_0, O_1, O_2, O_3 to Arduino Uno pin no D11, D10, D9, D8, respectively. After building the circuit we uploaded the Arduino code to the Nano and simulated

the project. We uploaded a video on Drive to demonstrate the Proteus simulation of the circuit.

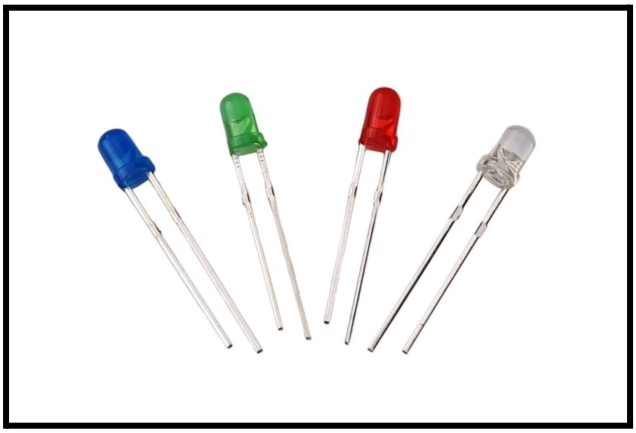
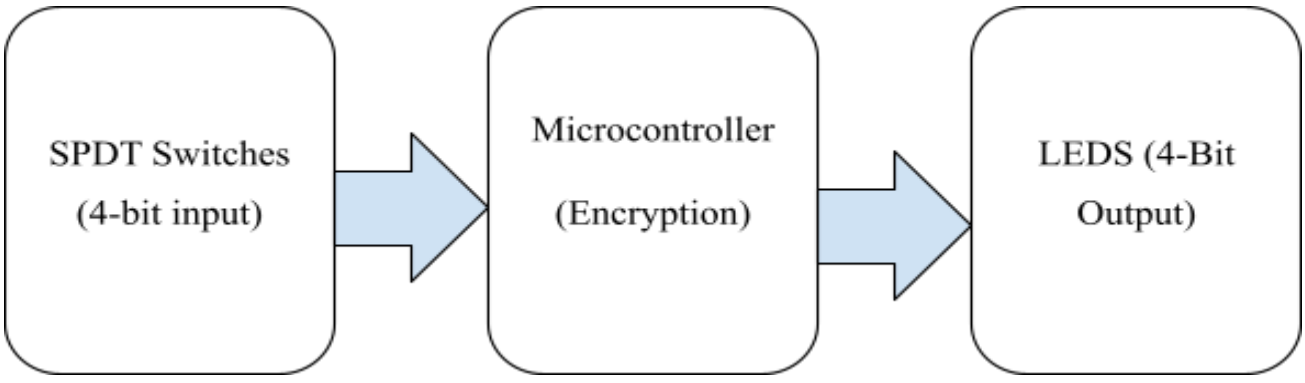


Figure: LEDs

Flow Diagram



Arduino Program

Arduino CODE:

```
int in_0 = 8;

int in_1 = 9;
int in_2 = 10;
int in_3 = 11;
int out_0 = 5;
int out_1 = 4;
int out_2 = 3;
int out_3 = 2;
int A = 0;
int B = 0;
int C = 0;
int D = 0;

void setup() {
      pinMode(in_0, INPUT);
      pinMode(in_1, INPUT);
      pinMode(in_2, INPUT);
      pinMode(in_3, INPUT);
      pinMode(out_0, OUTPUT);
      pinMode(out_1, OUTPUT);
      pinMode(out_2, OUTPUT);
      pinMode(out_3, OUTPUT);
}

void loop() {
      A = digitalRead(in_0);
      B = digitalRead(in_1);
      C = digitalRead(in_2);
      D = digitalRead(in_3);

      if ((!D && !C && !A) || (!C && !B) || (!D && !B && A) || (D
      && !C && A)){
            digitalWrite(out_0, HIGH);
      } else {
            digitalWrite(out_0, LOW);
      }

      if ((!D && B) || (B && A) || (!D && C) || (D && !C && !B)
      || (D && !B && !A)){
            digitalWrite(out_1, HIGH);
```

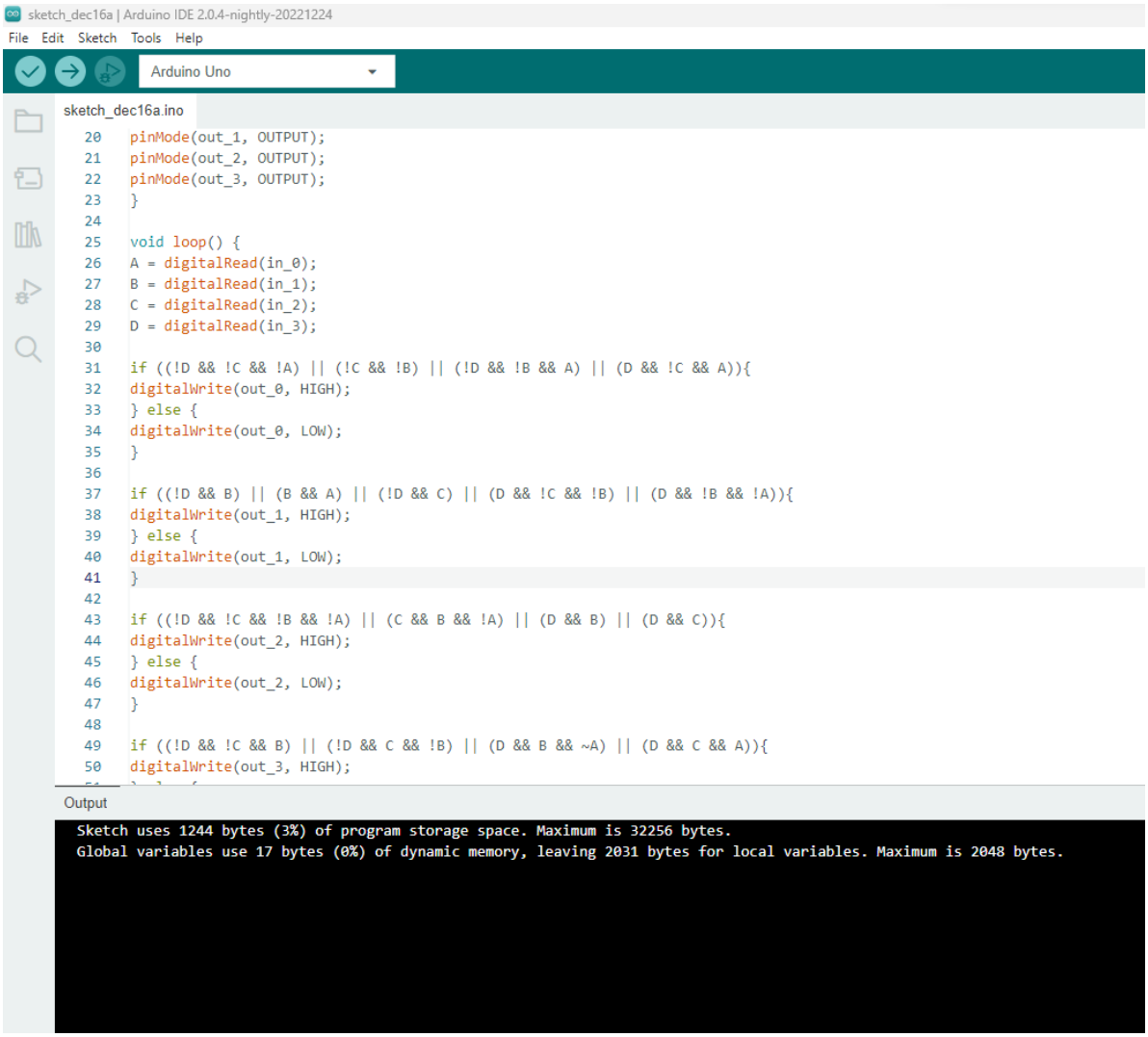
```
        } else {
              digitalWrite(out_1, LOW);
        }

        if ((!D && !C && !B && !A) || (C && B && !A) || (D && B) ||
        (D && C)){
              digitalWrite(out_2, HIGH);
        } else {
              digitalWrite(out_2, LOW);
        }

        if ((!D && !C && B) || (!D && C && !B) || (D && B && ~A) ||
        (D && C && A)){
              digitalWrite(out_3, HIGH);
        } else {
              digitalWrite(out_3, LOW);
        }

}
```

Compiling Arduino Code



Question and Answer

Question 1: What is the clock frequency of the microcontroller used?

Answer: We used Arduino UNO, and the clock frequency of the microcontroller is 16MHz.

Question 2: What is the data bus width of the microcontroller used?

Answer: The data bus width of Arduino UNO is 8-bit.

Question 3: What is the size of your hex file generated? Attach the hex codes in your report.

Answer: The hex file size is 3.43KB. The hex file is attached below:

```
:100000000C9461000C9473000C9473000C947300B6
:100010000C9473000C9473000C9473000C94730094
:100020000C9473000C9473000C9473000C94730084
:100030000C9473000C9473000C9473000C94730074
:100040000C9426010C9473000C9473000C947300B0
:100050000C9473000C9473000C9473000C94730054
:100060000C9473000C947300000000000240027001F
:100070002A0000000000250028002B0000000000DE
:10008000230026002900040404040404040202DA
:10009000020202020303030303030301020408102007
:1000A000408001020408102001020408102000012
:1000B000000800020100000304070000000000027
:1000C000000011241FBECFEFD8E0DEBFCDBF21E07E
:1000D000A0E0B1E001C01D92A131B207E1F70E949A
:1000E00070010C946B020C940000833081F028F4B2
:1000F000813099F08230A9F008958730A9F08830D6
:10010000C9F08430B1F4809180008F7D03C080916C
:1001100080008F7780938000089584B58F7784BDA9
:10012000089584B58F7DFBCF8091B0008F77809349
:10013000B00008958091B0008F7DF9CFCF93DF9309
:10014000282F30E0F901E255FF4F8491F901E6567E
:10015000FF4FD491F901EA57FF4FC491CC23A1F08E
:1001600081110E947500EC2FF0E0EE0FFF1FE458A4
:10017000FF4FA591B491EC91ED2381E090E009F45B
:1001800080E0DF91CF91089580E090E0FACF1F9357
:10019000CF93DF93282F30E0F901E255FF4F849190
:1001A000F901E656FF4FD491F901EA57FF4FC49188
:1001B000CC23A9F0162F81110E947500EC2FF0E0DE
:1001C000EE0FFF1FEE58FF4FA591B4918FB7F89433
:1001D000EC91111108C0D095DE23DC938FBFDF9125
:1001E000CF911F910895DE2BF8CFCF93DF9390E04E
:1001F000FC01E656FF4F24918A579F4FFC018491E2
:100200008823D1F090E0880F991FFC01E859FF4F37
:10021000A591B491FC01EE58FF4FC591D4916111A5
```

```
:100220000EC09FB7F8948C91E22FE0958E238C93AB
:100230002881E223E8839FBFDF91CF9108958FB794
:10024000F894EC91E22BEC938FBFF6CF1F920F92B4
:100250000FB60F9211242F933F938F939F93AF93D9
:10026000BF9380910D0190910E01A0910F01B0916B
:1002700010013091OC0123E0230F2D3758F5019622
:10028000A11DB11D20930C0180930D0190930E01CF
:10029000A0930F01B0931001809108019091090182
:1002A000A0910A01B0910B010196A11DB11D80938F
:1002B000080190930901A0930A01B0930B01BF912B
:1002C000AF919F918F913F912F910F900FBE0F9003
:1002D0001F90189526E8230F0296A11DB11DD2CFBD
:1002E000789484B5826084BD84B5816084BD85B511
:1002F000826085BD85B5816085BD80916E0081601D
:10030000809360010928100809181008260809 3C2
:1003100081008091810081608093810080918000C4
:1003200081608093800080 1B10084608093B100EF
:100330008091B00081608093B00080917A008460E9
:1003400080937A0080917A00826080937A00809115
:100350007A00816080937A0080917A00806880932F
:100360007A001092C10060E088E00E94F50060E031
:1003700089E00E94F50060E08AE00E94F50060E0FC
:100380008BE00E94F50061E085E00E94F50061E0ED
:1003900084E00E94F50061E083E00E94F50061E0E6
:1003A00082E00E94F50080E0A82E80E0B82E88E070
:1003B0000E949E008C01909307018093060189E0C2
:1003C0000E949E00EC0190930501809304018AE055
:1003D0000E949E007C019093030180930 2018BE0B8
:1003E0000E949E006C0190930101809300 1892B73
:1003F00049F4E114F10461F40115110571F020973D
:1004000029F40BC0E114F10409F450C060E006C007
:100410002097E1F701151105C9F361E085E00E941D
:10042000C700C114D10499F4209719F4E114F10420
:1004300099F061E084E00E94C700C114D10471F416
```

```
:10044000E114F104B9F52097B9F160E00BC02097F1
:10045000B9F10115110569F760E0ECCFE114F10481
:10046000B9F161E083E00E94C700CD2899F4EF283C
:1004700069F4CD2B69F061E082E00E94C700A1140D
:10048000B10409F494CF0E94000091CFCD2B99F3D1
:1004900060E0F2CFCD2B29F0012B69F3EF2859F75B
:1004A000F7CFEF28A9F3012B99F3E5CF209709F4B3
:1004B000B4CFB0CF209749F20115110531F6D1CF55
:1004C000E114F10409F4B5CF0115110529F6B1CFF6
:0A04D000209739F6BACFF894FFCF59
:00000001FF
```

Question 4: Can the project be implemented by using interrupt?

Answer: Yes, an interruption can be used to implement the project. The start and stop operations of the CPU can be controlled using interrupts. Interrupts can be produced by the Arduino UNO's pins 2 and 3. Additionally, there is a feature known as digital PinInterrupt (pin). However, if an interrupt is produced, neither new data nor new output will occur.

Question 5: Is the main routine required to be an infinite loop? Provide an explanation in favor of your answer.

Answer: Yes, an infinite loop must be used for the main routine. We regularly get 4-bit data, which we encrypt. The machine must therefore continue to operate in a loop. After receiving input, the program would get stuck without a loop and stop providing constant feedback. An infinite loop was, therefore, necessary.

Question 6: Is there any difference between level-triggered and edge-triggered operation for the

given project?

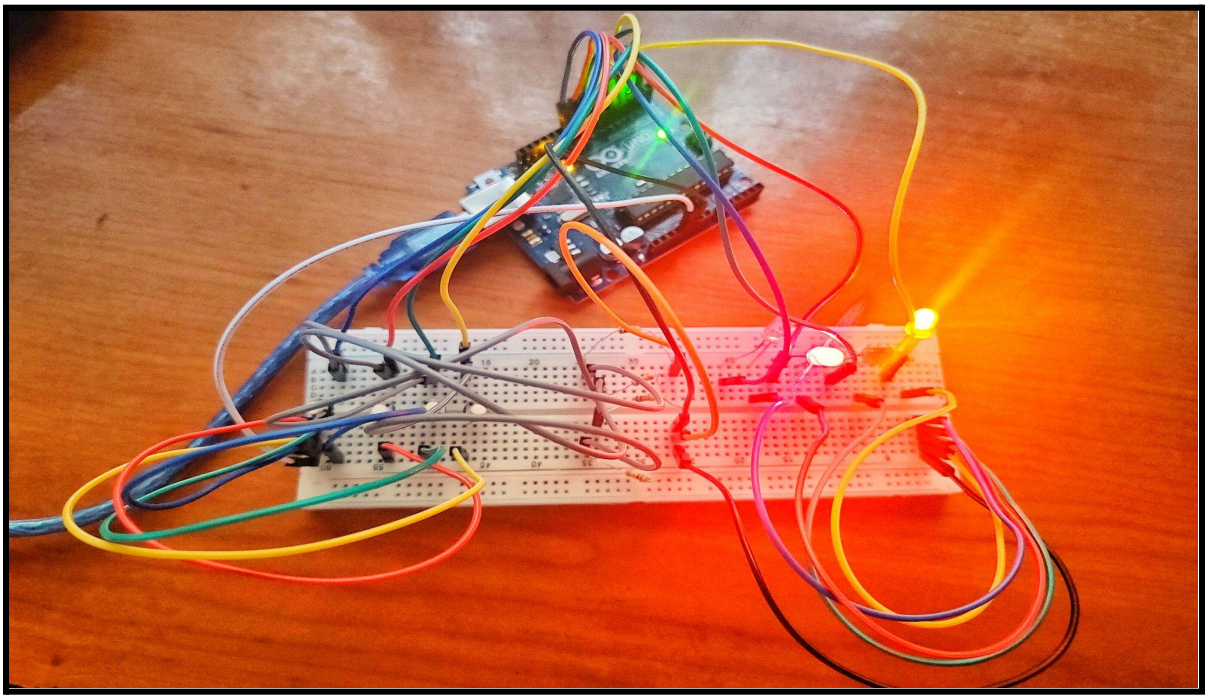
Answer: A level-triggered interrupt module always initiates an interrupt when the level of the interrupt source is asserted. An edge-triggered interrupt module generates an interrupt when the level of the interrupt source shifts from inactive to active. As a result, an edge-triggered interrupt module can only have one interrupt per edge. In contrast, a level-triggered interrupt module can have several interrupts while the interrupt level is asserted. Since continuous input would not be possible with an edge-triggered interrupt module, this project would not work.

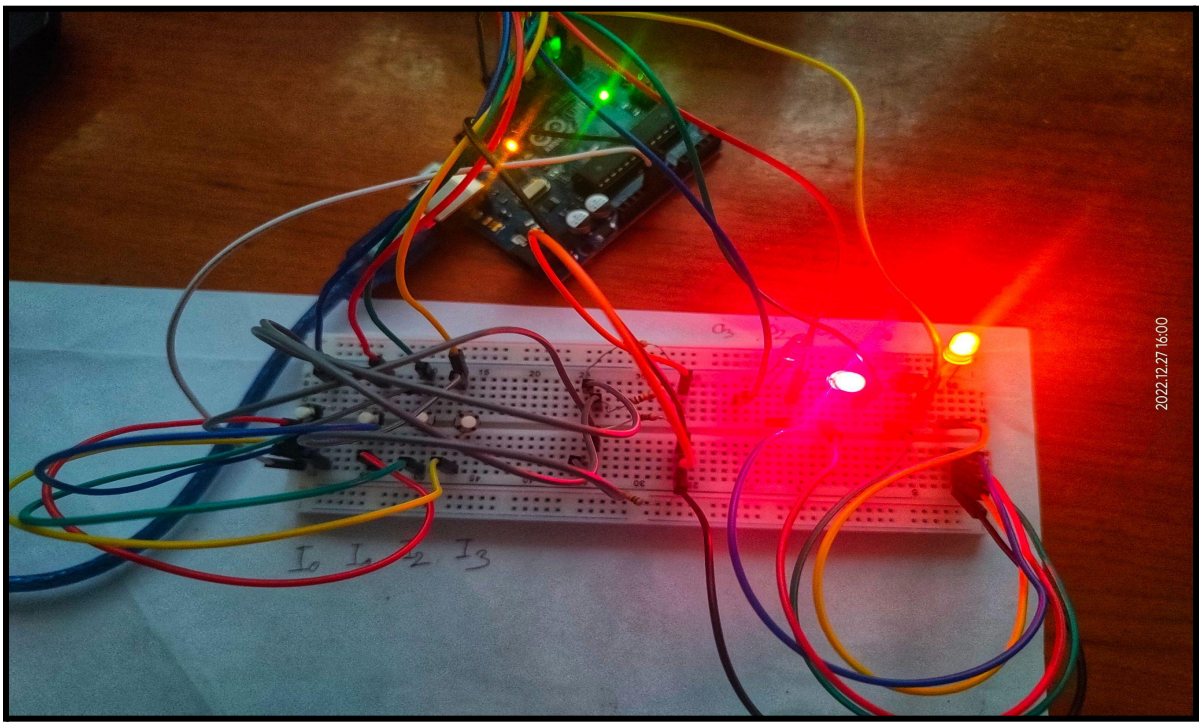
Question 7: Is the project referring to encryption or decryption from input to output?

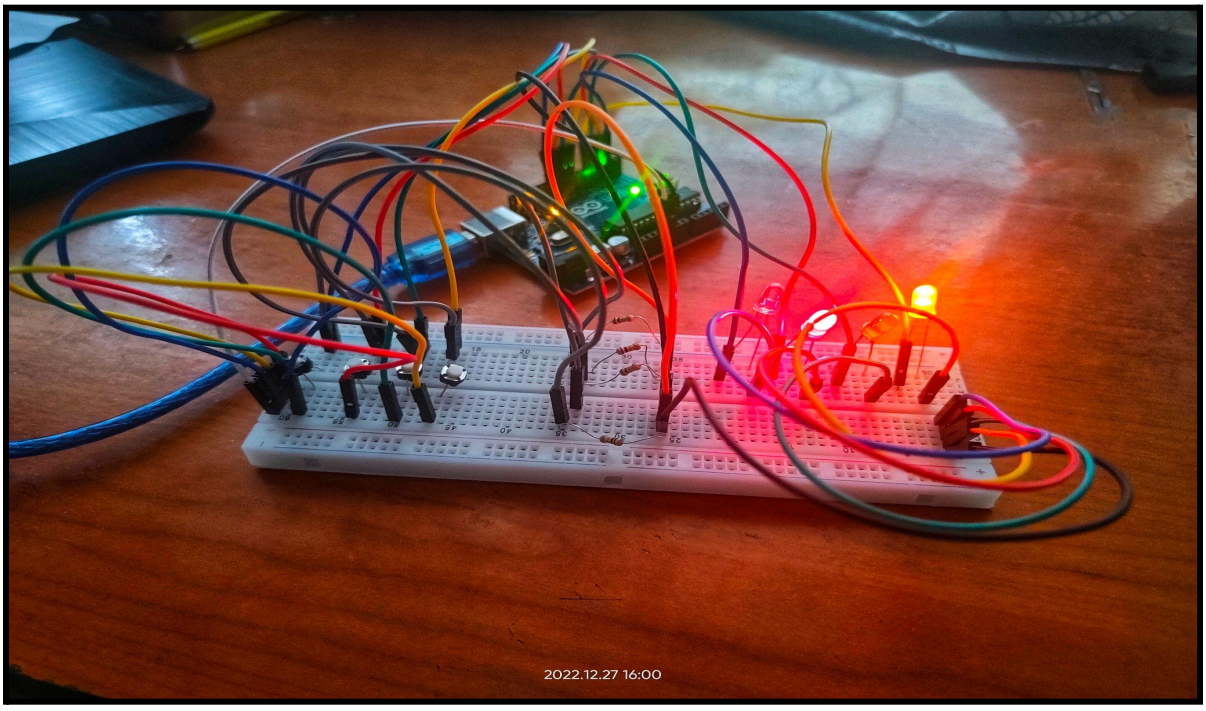
Answer: The input and output lengths are the same, and the encryption table is available. The information we've received is being encrypted. The output is the input's encrypted data.

Hardware Implementation

Here is our hardware implementation:




I0 I1 I2 I3
2022.12.27 16:00


2022.12.27 16:00

References

https://en.wikipedia.org/wiki/Proteus_Design_Suite

https://en.wikipedia.org/wiki/Arduino_Uno

https://digitalguardian.com/blog/what-data-encryption

https://pediaa.com/what-is-the-difference-between-edge-and-level-triggering/

https://store.roboticsbd.com/components/761-push-button-switch-robotics-bangladesh.html