ROBOTiX CLUB
National Institute of Technology
Raipur

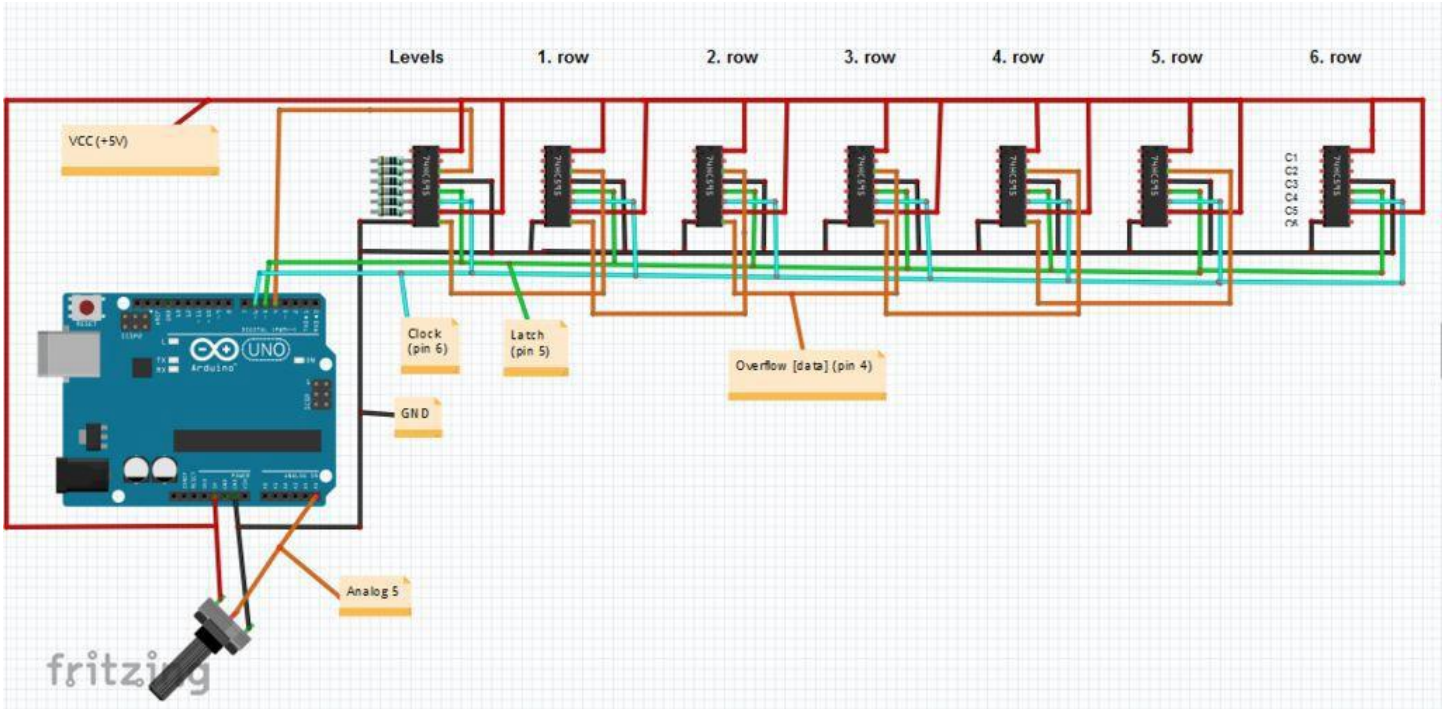
Project Name - LED Cube

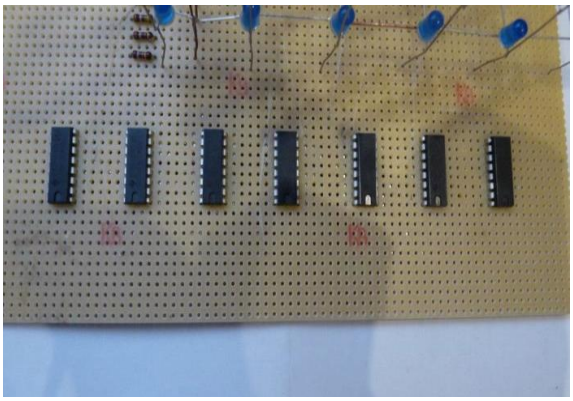
Mentor - Kanishka Gabel **Members -** N Suraj Kumar, Priyadarshi Arun, Yogesh Markandey

1. Introduction

The 3D **LED Cube** is a really cool device that enables you to see in three dimensions, get some depth perception and has 512 **LED's** or 512 pixels.
It is based on an Arduino Uno which is an Atmel AVR microcontroller on a development board with some standard interfacing pinouts that allow you to quickly assemble prototypes.

2. Blueprints





3. Working

This is a 6x6x6 LED cube, so it consists of 6 levels, 6 rows, every row has 6 columns.
In a level, all the cathodes are soldered with their neighbors, like a net.
In a column, all the anodes are soldered with the upper and lower neighbors. So if I switch VCC to a column, and GND to a level, the LED, with is in the intersection of the level and the column will light.
The levels and every row are controlled by 1-1 shift register.
We use 7 shift registers, one for the levels to control its ground, and the other 6 for the rows, each one for one row.
A shift register controls a whole row, a shift register has 8 output ports, but a row only has 6 columns, so it is more than enough.

4. Construction

Materials Required:

- 256x 5 [mm] blue LED (optional)
- 6x 180 [Ω] resistors
- 7x **SN74HC595N** shift register
- 1x Arduino UNO
- 1x potentiometer (optional)
- Wires
- Soldering wire and iron
- Solder able perf- board
- Male header connector (optional)
- A piece of wood
- A drill machine

1. Test all the LEDs with a simple circuit. All of them must be fit. I switched +5[V] to the LED's anode, and the GND to the LED's cathode through a 180 [Ω] resistor.

2. Then I curved the legs of the LEDs with a pliers, bended the anodes to a right angle and placed them into the holes upside down, so the anodes are faced the outside part of the cube.

3. It's time to start soldering. We soldered the cathodes to each other and then I used some copper wire to connect the two sides and to make the structure stronger. If you are ready, carefully try to get out the level from the wood. If you have all the 6 levels,

then you have to solder the anodes of the second level to the anodes of the first level, and so on until you have all the levels soldered.

4. After this, solder the anodes of the first level to a solder able perf board. To connect the levels to the resistors i used some copper wire.

About the Code

To control the shift registers you can use binary or hexadecimal numbers, i preferred the hexa one because it is shorter.
I created an array for the shift registers,
SR[6] controls the levels, and the others (SR[1-5]) control the rows. For example SR [6] = 0x7F that means only the first level gets GND, the others don't SR [5] = 0xC0 means on the 5. Row only the first 2 led will light on the 1 level.

```
int dataPin = 4;
int latchPin = 5;
int clockPin = 6;

byte SR[7];
// an array for the shift registers, SR[0-5] control the rows and the
//SR[6] control the levels

int LEVEL[6] = {0x7F, 0xBF, 0xDF, 0xEF, 0xF7, 0xFB};
// this is an array for each levels, 0x7F means only the first row will
//get GND and so on

int LEVEL2[5] = {0x3F, 0x9F, 0xCF, 0xE7, 0xF3};
// this is an array for the neighbour levels

int ROW2[5] = {0xC0, 0x60, 0x30, 0x18, 0x0C}; // for every neighbour rows
int potdelay; // this delay is adjustable by the potentiometer

void setup()
{
pinMode(dataPin, OUTPUT);
pinMode(latchPin, OUTPUT);
pinMode(clockPin, OUTPUT);
Serial.begin(9600);
}
void loop()
{ potdelay = analogRead(A5); // this will read the potentiometer's state
  three();
  two();
```

```
  one();
  delay(1500);
  full();
  delay(2000);
  frame6x6();
  GrowingCube();
  frame4x4();
  frame6x6();
  randomLED();
  FrameInFrame();
  frame4x4();
  centerCube();
  randomCUBE();
  custom();
}

void one()
{
SR[6] = 0x00;
SR[0] = SR[1] = 0x30;
shift_out();
}

void two()
{
for (int k = 0; k < 100; k ++)
{
SR[6] = 0xF8;
SR[0] = SR[1] = 0x30;
shift_out();
delayMicroseconds(1);
clear_out();
SR[6] = 0xF4;
SR[0] = SR[1] = 0x48;
shift_out();
delayMicroseconds(1);
clear_out();
SR[6] = 0xEF;
SR[0] = SR[1] = 0x08;
shift_out();
delayMicroseconds(1);
clear_out();
SR[6] = 0xDF;
SR[0] = SR[1] = 0x30;
shift_out();
delayMicroseconds(1);
clear_out();
SR[6] = 0xBF;
```

```
SR[0] = SR[1] = 0x40;
shift_out();
delayMicroseconds(1);
clear_out();
SR[6] = 0x7F;
SR[0] = SR[1] = 0x78;
shift_out();
delayMicroseconds(1);
clear_out();
}
}

void three()
{
for (int k = 0; k < 150; k ++) //with this loop you can setup a 'delay'
{
SR[6] = 0x7F;
SR[0] = SR[1] = 0x30;
shift_out();
delayMicroseconds(1);
clear_out();
SR[6] = 0xBF;
SR[0] = SR[1] = 0x48;
shift_out();
delayMicroseconds(1);
clear_out();
SR[6] = 0xDF;
SR[0] = SR[1] = 0x08;
shift_out();
delayMicroseconds(1);
clear_out();
SR[6] = 0xEF;
SR[0] = SR[1] = 0x10;
shift_out();
delayMicroseconds(1);
clear_out();
SR[6] = 0xF7;
SR[0] = SR[1] = 0x08;
shift_out();
delayMicroseconds(1);
clear_out();
SR[6] = 0xF8;
SR[0] = SR[1] = 0x78;
shift_out();
delayMicroseconds(1);
clear_out();
}
}
```

```
void GrowingCube()
{
SR[6] = 0x3F; //2x2
SR[0] = SR[1] = 0x0C;
shift_out();
delay(300);
SR[6] = 0x1F; //3x3
SR[0] = SR[1] = SR[2] = 0x1C;
shift_out();
delay(300);
SR[6] = 0x0F; //4x4
SR[0] = SR[1] = SR[2] = SR[3] = 0x3C;
shift_out();
delay(300;
SR[6] = 0x07; //5x5
SR[0] = SR[1] = SR[2] = SR[3] = SR[4] = 0x7C;
shift_out();
delay(300);
SR[6] = 0x03; //6x6
SR[0] = SR[1] = SR[2] = SR[3] = SR[4] = SR[5] = 0xFC;
shift_out();
delay(300);
clear_out();
delayMicroseconds(10);
SR[6] = 0x80; //5x5
SR[5] = SR[4] = SR[3] = SR[2] = SR[1] = 0xF8;
shift_out();
delay(300);
clear_out();
delayMicroseconds(10);
SR[6] = 0xC0; //4x4
SR[5] = SR[4] = SR[3] = SR[2] = 0xF0;
shift_out();
delay(300);
clear_out();
delayMicroseconds(10);
SR[6] = 0xE0; //3x3
SR[5] = SR[4] = SR[3] = 0xE0;
shift_out();
delay(300); clear_out();
delayMicroseconds(10);
SR[6] = 0xF0; //2x2
SR[5] = SR[4] = 0xC0;
shift_out();
delay(300);
clear_out();
delayMicroseconds(10);
}
```

```
void frame6x6()
{
for (int k = 0; k < 200; k ++)
{
SR[6] = 0x7B;
SR[0] = SR[5] = 0xFF;
SR[1] = SR[2] = SR[3] = SR[4] = 0x84;
shift_out();
delayMicroseconds(5);
clear_out();
delayMicroseconds(5);
SR[6] = 0x84;
SR[0] = SR[5] = 0x84;
shift_out();
delayMicroseconds(5);
clear_out();
delayMicroseconds(5);
}
}

void frame4x4()
{
for (int k = 0; k < 200; k ++)
{
SR[6] = 0xB7;
SR[1] = SR[4] = 0x78;
SR[2] = SR[3] = 0x48;
shift_out();
delayMicroseconds(5);
clear_out();
delayMicroseconds(5);
SR[6] = 0xCF;
SR[1] = SR[4] = 0x48;
shift_out();
delayMicroseconds(5);
clear_out();
delayMicroseconds(5);
}
}

void FrameInFrame()
{
for (int k = 0; k < 200; k ++)
{
SR[6] = 0x7B; //top and button
SR[0] = SR[5] = 0xFF;
SR[1] = SR[2] = SR[3] = SR[4] = 0x84;
shift_out();
```

```
delayMicroseconds(5);
clear_out();
delayMicroseconds(5);
SR[6] = 0xB7;
SR[0] = SR[5] = 0x84;
SR[1] = SR[4] = 0x78;
SR[2] = SR[3] = 0x48;
shift_out();
delayMicroseconds(5);
clear_out();
delayMicroseconds(5);
SR[6] = 0xCF;
SR[0] = SR[5] = 0x84;
SR[1] = SR[4] = 0x48;
shift_out();
delayMicroseconds(5);
clear_out();
delayMicroseconds(5);
}
}

void custom()
{
for (int k = 0; k < 200; k ++)
{
SR[6] = 0x7B; //top and button
SR[0] = SR[5] = 0xFF;
SR[1] = SR[2] = SR[3] = SR[4] = 0x84;
shift_out();
delayMicroseconds(5);
clear_out();
delayMicroseconds(5);
SR[6] = 0xB7;
SR[0] = SR[5] = 0x00;
SR[1] = SR[4] = 0x78;
SR[2] = SR[3] = 0x48;
shift_out();
delayMicroseconds(5);
clear_out();
delayMicroseconds(5);
SR[6] = 0xCF;
SR[2] = SR[3] = 0x30;
shift_out();
delayMicroseconds(5);
clear_out();
delayMicroseconds(5);
}
}
```

```
void centerCube()
{
SR[6] = 0xCF;
SR[2] = SR[3] = 0x30;
shift_out();
delay(500);
}

void randomCUBE()
{ //2x2 cube
for (int k = 0; k < 20; k ++)
{
int SRrandom = random(0, 5);
SR[6] = LEVEL2[random(0, 5)];
SR[SRrandom] = SR[SRrandom + 1] = ROW2[random(0, 5)];
shift_out();
delay(potdelay);
clear_out();
}
}

void randomLED()
{ //only 1 LED will light
for (int k = 0; k < 20; k ++)
{
SR[6] = LEVEL[random(0, 6)]; //randomly choose a level
bitSet(SR[random(0, 6)], random(2, 8));
// in that level randomly choose a row and in that row a led that will
//light, the others will off

shift_out();
delay(potdelay);
clear_out();
}
}

void full()
{ // if you use this program the whole cube will light
for (int i = 0; i < 6; i ++)
{ //this line will switch all the rows high
SR[i] = 0xFC;
}
SR[6] = 0x00; // and this line will switch all the levels low
shift_out();
}


//_____________________________________________________________________
```

```
void clear_out()
{ //this will clear_out all the leds
for (int i = 0; i < 6; i++)
{
SR[i] = 0x00;
// i use hexadecimal codes instead of binary
//because i think it is more shorter
}

SR[6] = 0xFF;
shift_out(); //this will shift_out the 'clear_out' command.
}

void shift_out()
{ // this will update the hexadecimal code to the shift registers
digitalWrite(latchPin, 0);
for (int i = 0; i < 7; i++)
{
shiftOut(dataPin, clockPin, MSBFIRST, SR[i]);
}
digitalWrite(latchPin, 1);
}
```