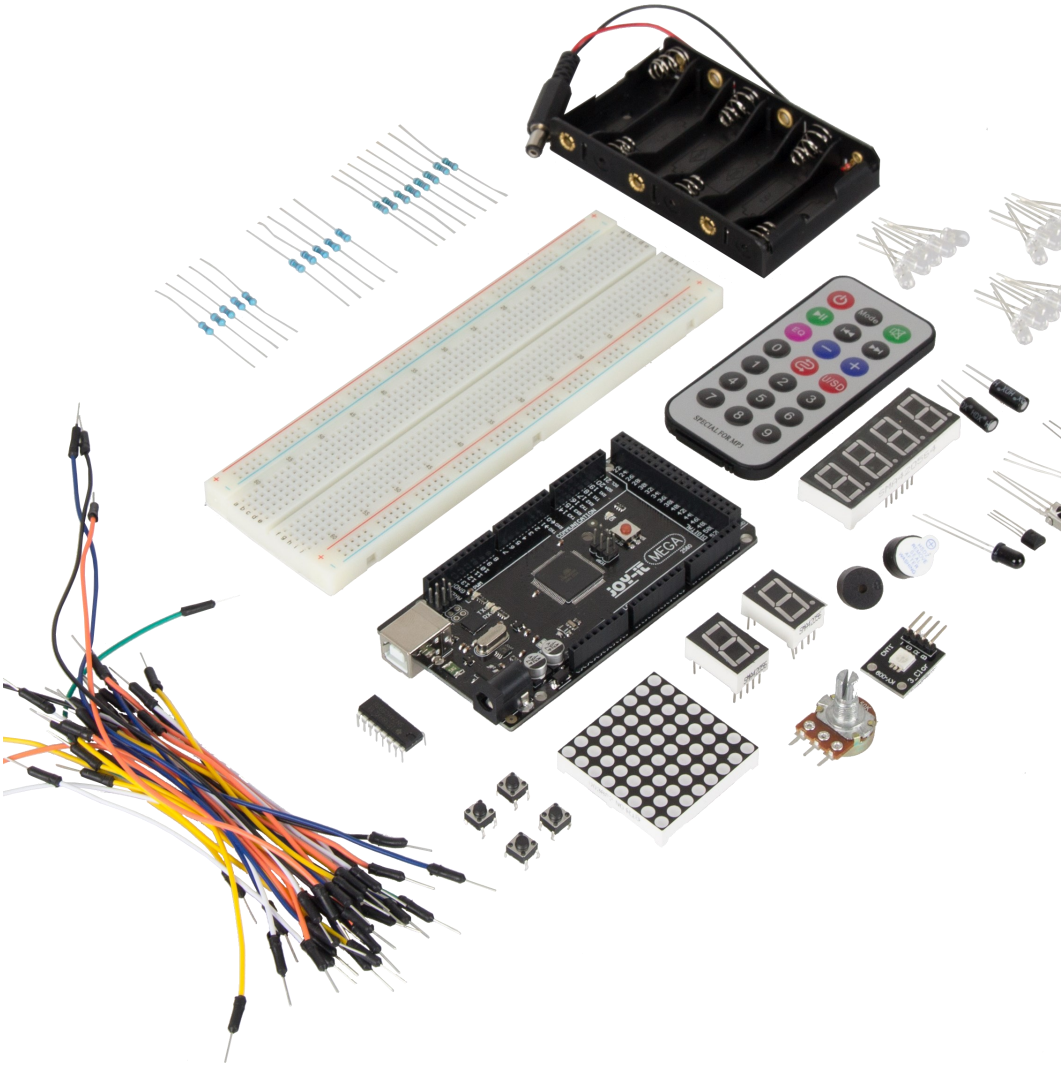


Mega2560

Microcontroller Learning-Kit

Index

1. Introduction

Dear customer,
Thank you for choosing our product.
In the following, we will show you what to observe during the use.
If you encounter any unexpected problems, please do not hesitate to contact us.

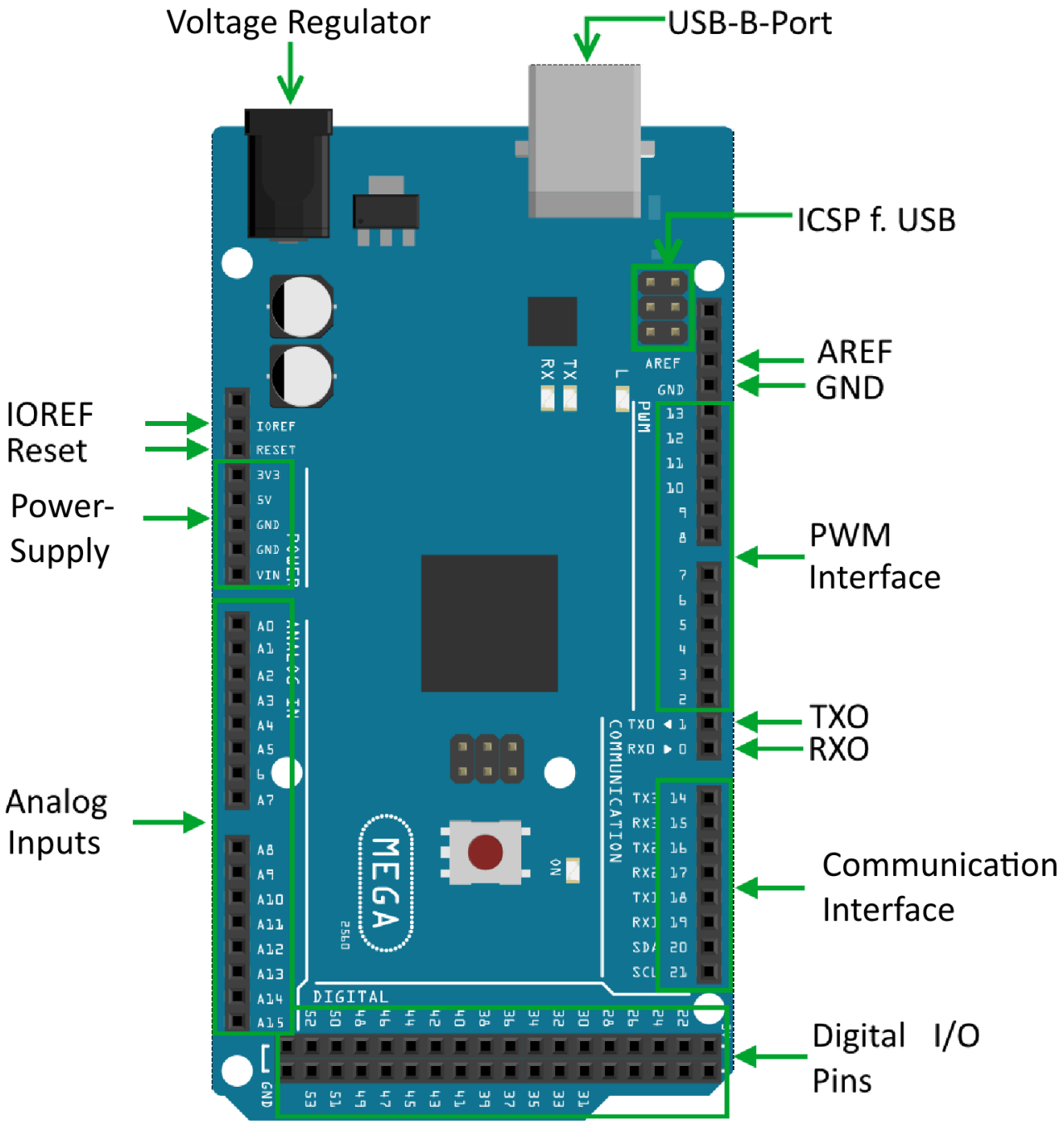
2. General Informations

2.1 Technical Specifications

Our board is a high-quality replica and compatible with the Arduino Mega 2560.
But it is explicitly not an original Arduino.
The Mega-Board is the right microcontroller board for those who want to get into the
in the world of programming. This set guides you through a variety of projects.
Its ATMega2560 microcontroller offers you enough power for your ideas and projects.
It measures 101.52 mm x 53.3 mm and has many connection options with 54 digital inputs and outputs and 16 analog inputs.

Model	ARD_Mega2560R3
Microcontroller	ATmega2560
Input-Voltage	7-12V
Input-Voltage (max.)	6-20V
Digital IO	54 (14 with PWM)
Analog IO	16
DC Current IO	40mA
DC Current 3.3V	50mA
Memory	256kB (8kB for Bootloader)
SRAM	8kB
EEPROM	4kB
Clock Speed	16 MHz
Dimensions	101.52mm x 53.3mm

2.2 Connection Assignment



3. Softwareinstallation

In order to start programming the JOY-iT ARD_Mega2560R3, a development environment and the driver for the corresponding operating system.

The Arduino IDE, which was developed by the Arduino manufacturer as a OpenSource software was released under GPLv2, which is based on the concept and structure of the for beginners.

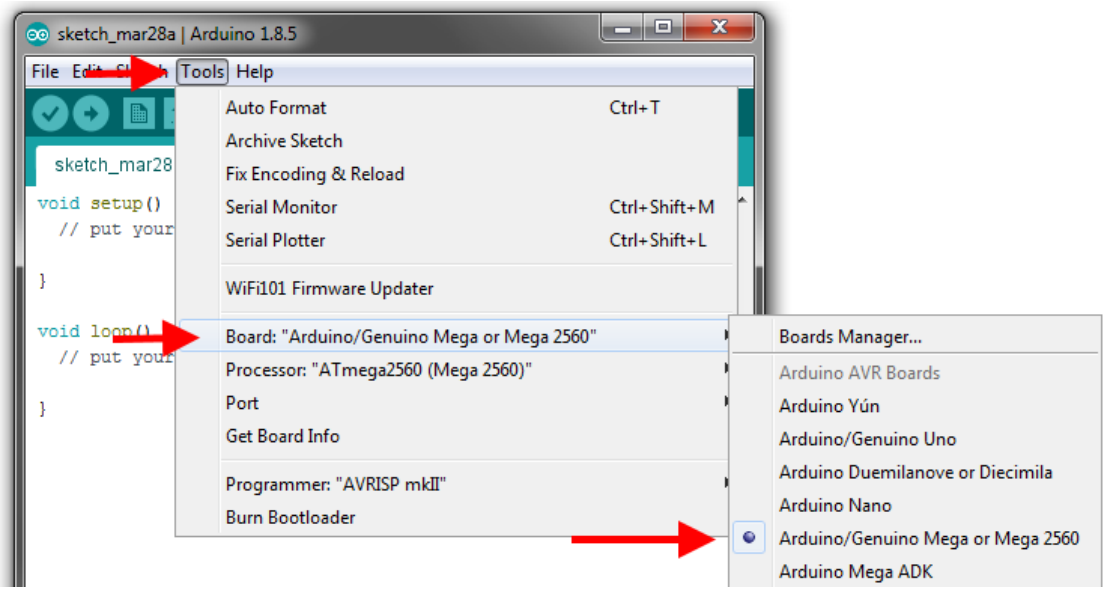
This is fully compatible with the JOY-iT ARD_Mega2560R3 and, in addition to the programming environment, you will also find the drivers you need to get started right away.

You can find the software for download here.

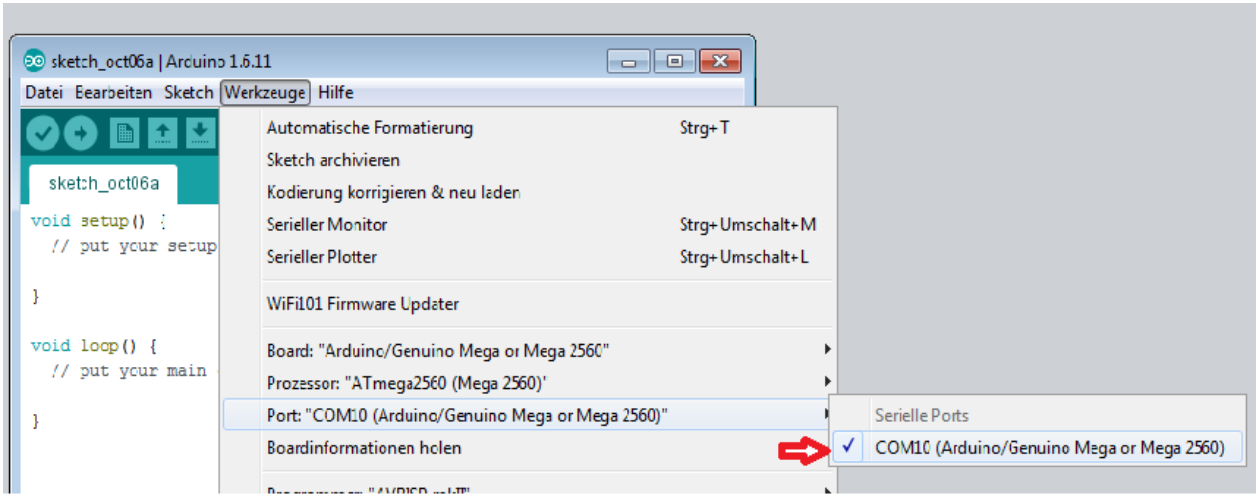
3.1 Software Setup

After installing the software, the corresponding microcontroller board must be set up in the programming environment. To do this, follow these two steps:

Select "Arduino/Genuino Mega or Mega 2560" under[Tools->Board].



2. Select the port marked"(Arduino/Genuino Mega or Mega 2560)" under[Tools -> Port].



4. Project Examples

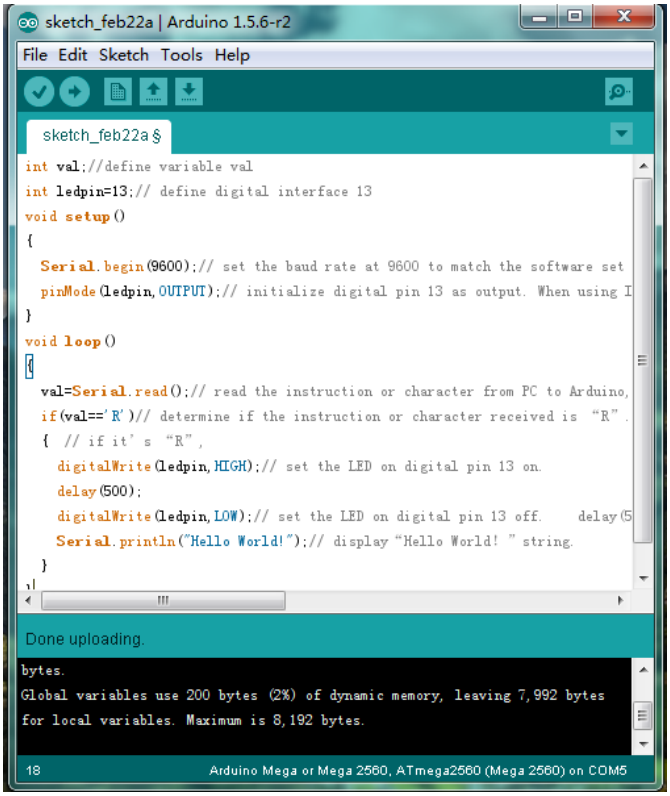
4.1 Hello World

We'll start with something simple.
For this project you only need the board and a USB cable to connect the "Hello World"! This is a communication test for your Mega2560 and your PC, as well as a basic project for your first attempt in the Arduino world!

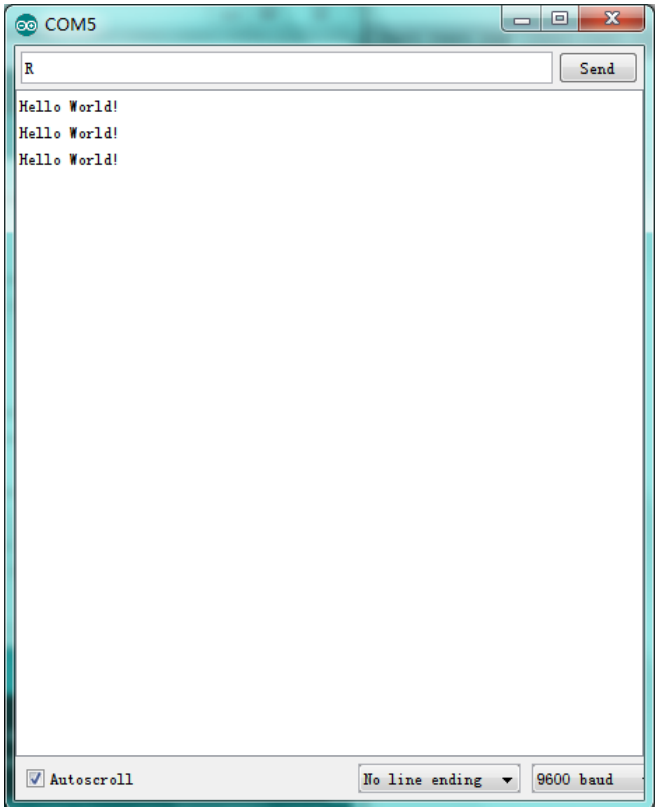
Hardware	Amount
Mega2560 Board	1
USB Cable	1

After the installation of the drivers is complete, let's open the Arduino software and write a code that will allow the Mega2560 to display "Hello World!" under your instruction. Of course, you can also write a code that lets the Mega2560 play "Hello World!" repeatedly without instruction. A simple if () command will do this.
We can instruct the LEDs on pin 13 to blink first and then display "Hello World" after the Arduino receives the command to do so.

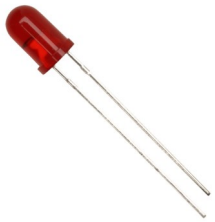
```
int val;              // defines the variable “Val”
int ledpin=13;        // defines the digital interface 13
void setup()
{
  Serial.begin(9600);
  // sets the baudrate to 9600 to fit the software konfiguration

  pinMode(ledpin,OUTPUT);
  // sets the digital pin 13 to output
}

void loop()
{
  val=Serial.read();

  if(val=='R')
  // checks if the character is a R.
  {
    // if so:
    digitalWrite(ledpin,HIGH);   // turns on LED
    delay(500);
    digitalWrite(ledpin,LOW);    // turns off LED
    delay(500);
    Serial.println("Hello World!"); // Shows „Hello World!”.
  }
}
```



Click on the serial port monitor, insert "R", LED will light up once, PC will receive the information "Hello World" from Arduino.



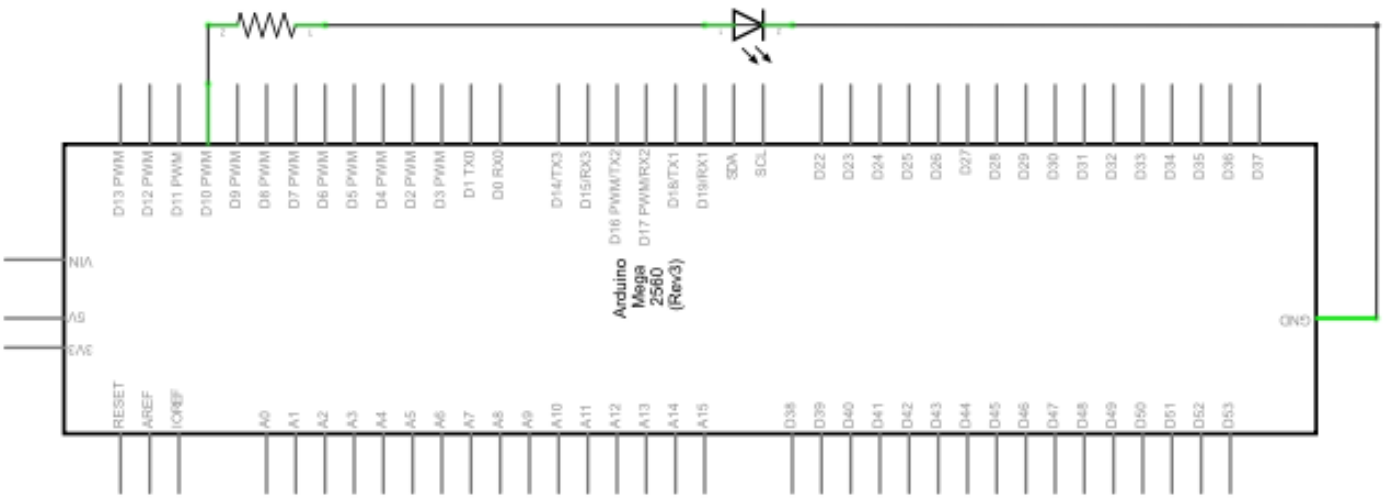
4.2 Flashing LED

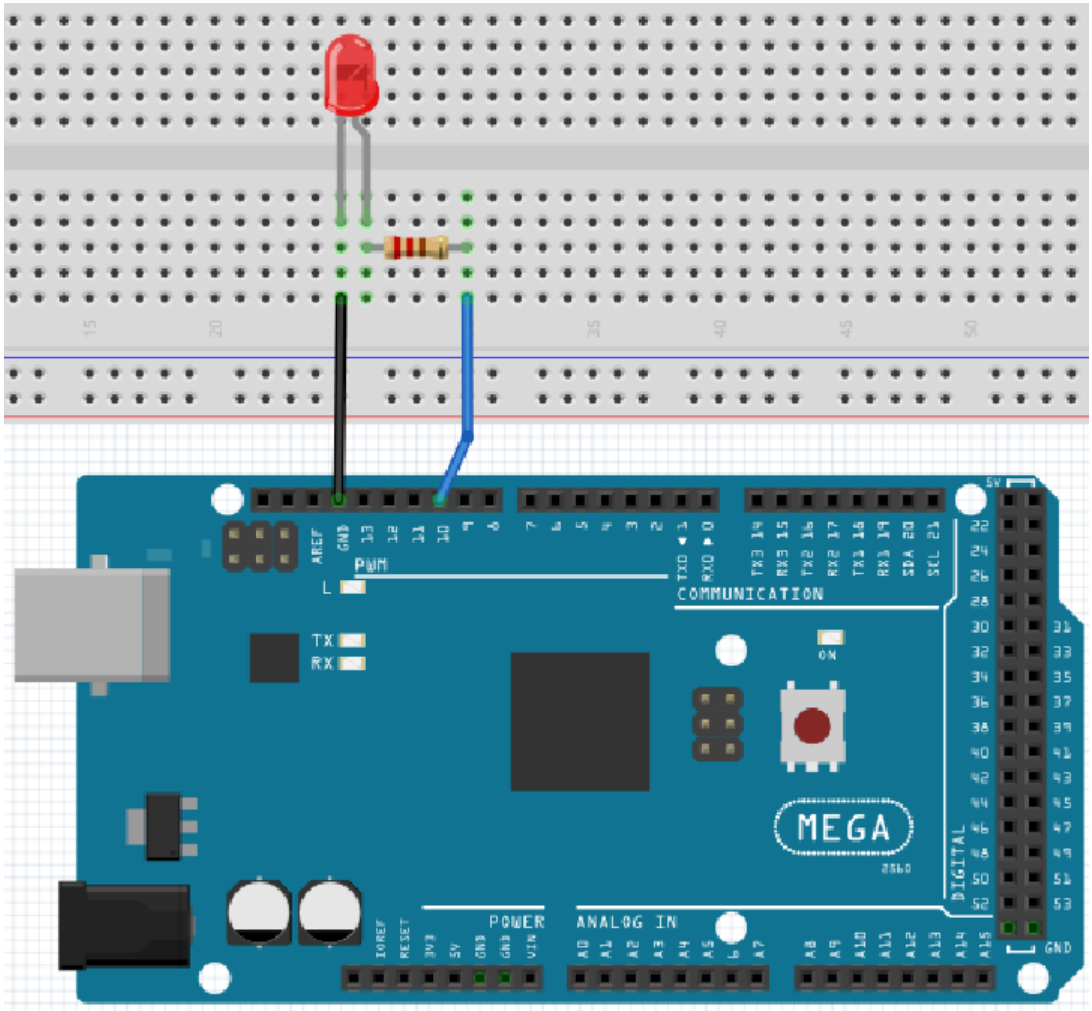


The "flashing LED" experiment is quite simple.
In the "Hello World!" program we have already encountered the LED. This time we will connect an LED to one of the digital pins. The following parts are required:

Hardware	Amount
Mega2560 Board	1
USB Cable	1
Red M5 LED	1
220Ω Resistor	1
Breadboard	1
Breadboard Jumper Cable	2

We follow the following diagram. Here we use the digital pin 10. we connect the LED with a 220 Ohm resistor to avoid damage by too high currents.





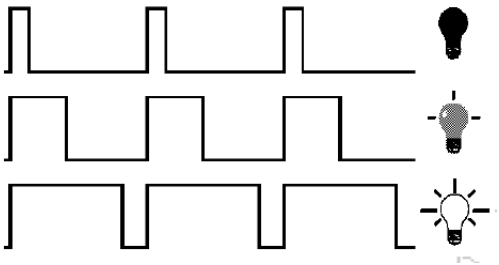

```
int ledPin = 10;                    // defines Digital Pin 10.

void setup()
{
  pinMode(ledPin, OUTPUT); // defines pin to output
}

void loop()
{
  digitalWrite(ledPin, HIGH);    // turns on led
  delay(1000);                   // waits a second
  digitalWrite(ledPin, LOW);     // turns off led
  delay(1000);                   // waits a second}
```

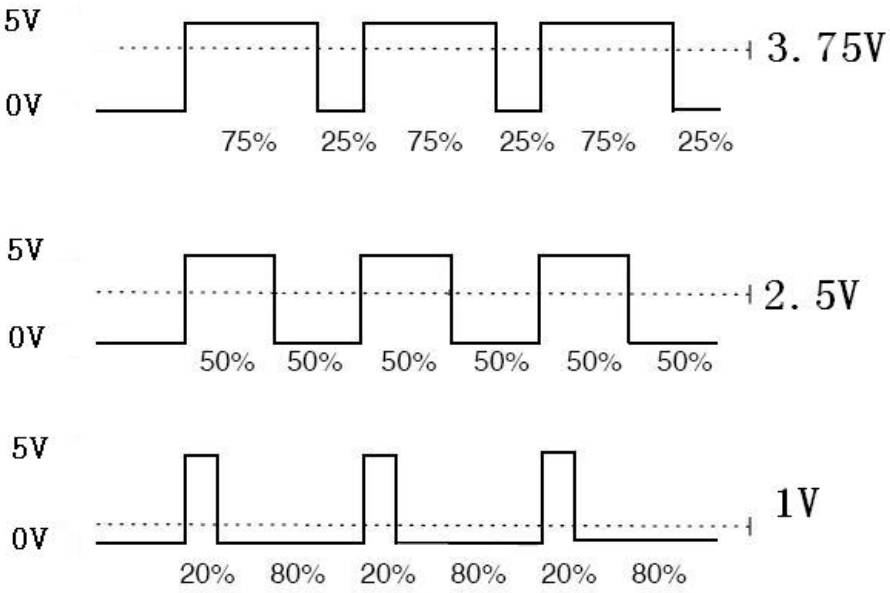
After downloading this program, you will see the LED connected to pin 10 turn on and off at an interval of about one second.

4.3 PWM Light Control

PWM, short for pulse width modulation, is a technique used to encode analog signal levels into digital ones. A computer is not capable of outputting analog voltage.
It can only output digital voltage with values such as 0V or 5V.
Therefore, a high-resolution counter is used to encode a specific analog signal level by modulating the load factor of PWM. The PWM signal is also digitized, since the power supply is either 5V (ON) or 0V (OFF) at any time.
The voltage or current is applied to the analog load (the device that consumes the energy) by repeated pulse sequences are supplied by permanently switching between the switched-on and switched-off state. The value of the output voltage is determined on the basis of the on and off states.

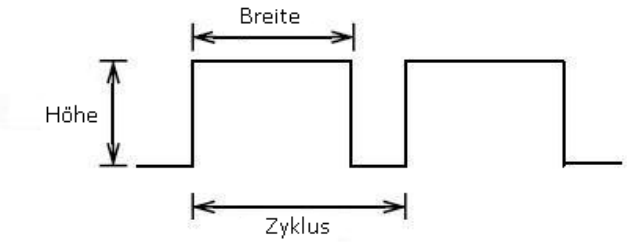
Voltage = (ON-Duration/ Pulse-Duration) x max. Voltage





There are many applications for PWM: regulation of lamp brightness, regulation of motor speed, etc..

Below are the three basic parameters of PWM:



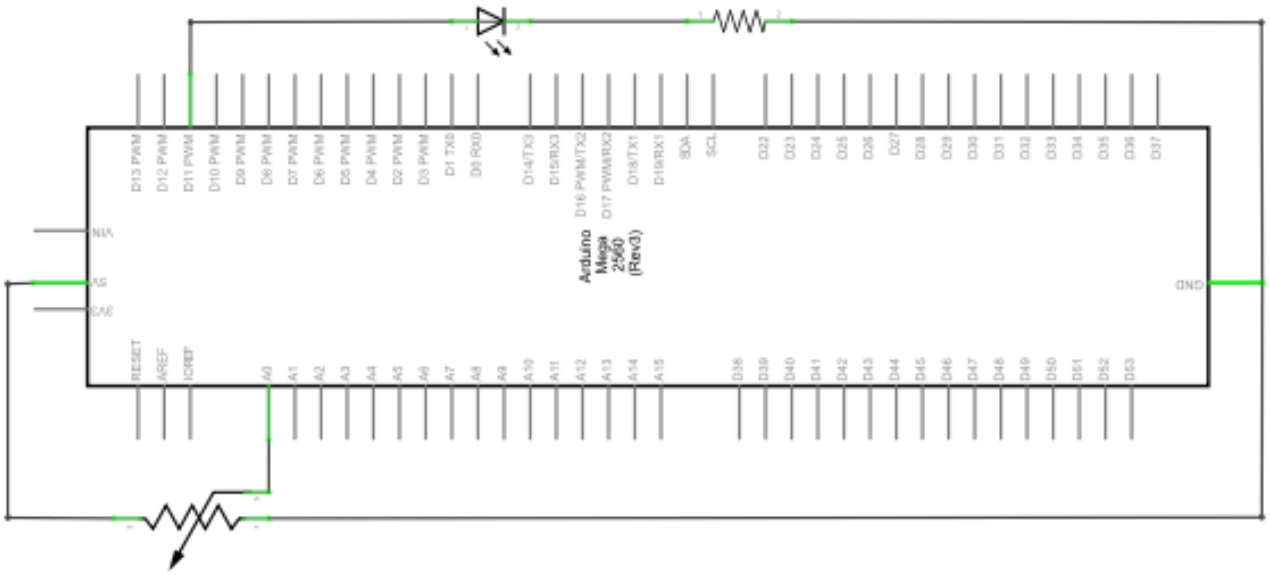
1. the amplitude of the pulse width (minimum/maximum)
2. the pulse period (the mutual pulse frequency in one second)
3. the voltage level (like 0V-5V)

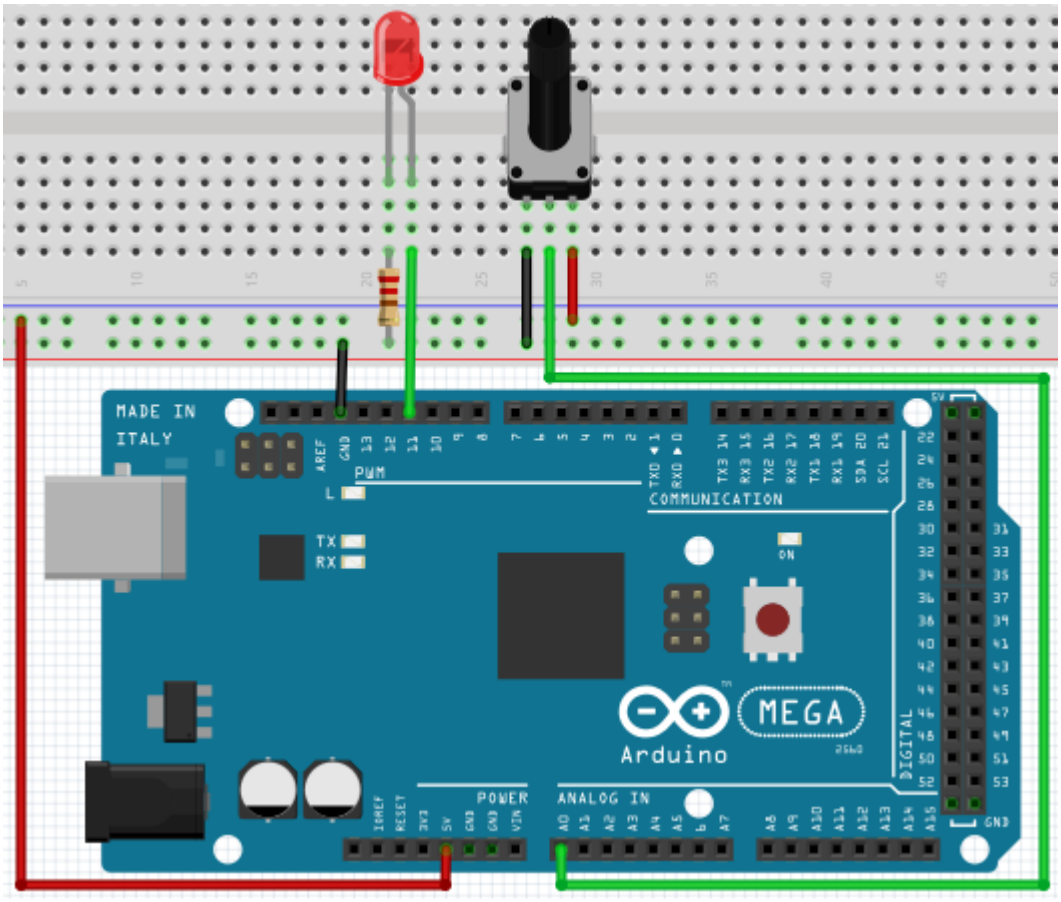
There are 6 PWM interfaces on the Mega2560: digital pin 3, 5, 6, 9, 10 and 11.

In previous experiments we learned the "key-controlled LED", where we used a digital signal to control a digital pin.
Now we will use a potentiometer to control the brightness of the LED.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
Red M5 LED	1
Variable Resistor	1
220Ω Resistor	1
Breadboard	1
Breadboard Jumper Cable	6

The input of the potentiometer is analog, so we connect it to the analog port. We connect the LEDs to the PWM port. Another PWM signal can regulate the brightness of the LED.





In this experiment we will read the analog value of the potentiometer and assign the value to the PWM port so that a corresponding change in the brightness of the LED can be observed.
In addition, the analog value will be displayed on the screen.
You can see this as the "Read Analog Value" project to which an analog PWM value is assigned.

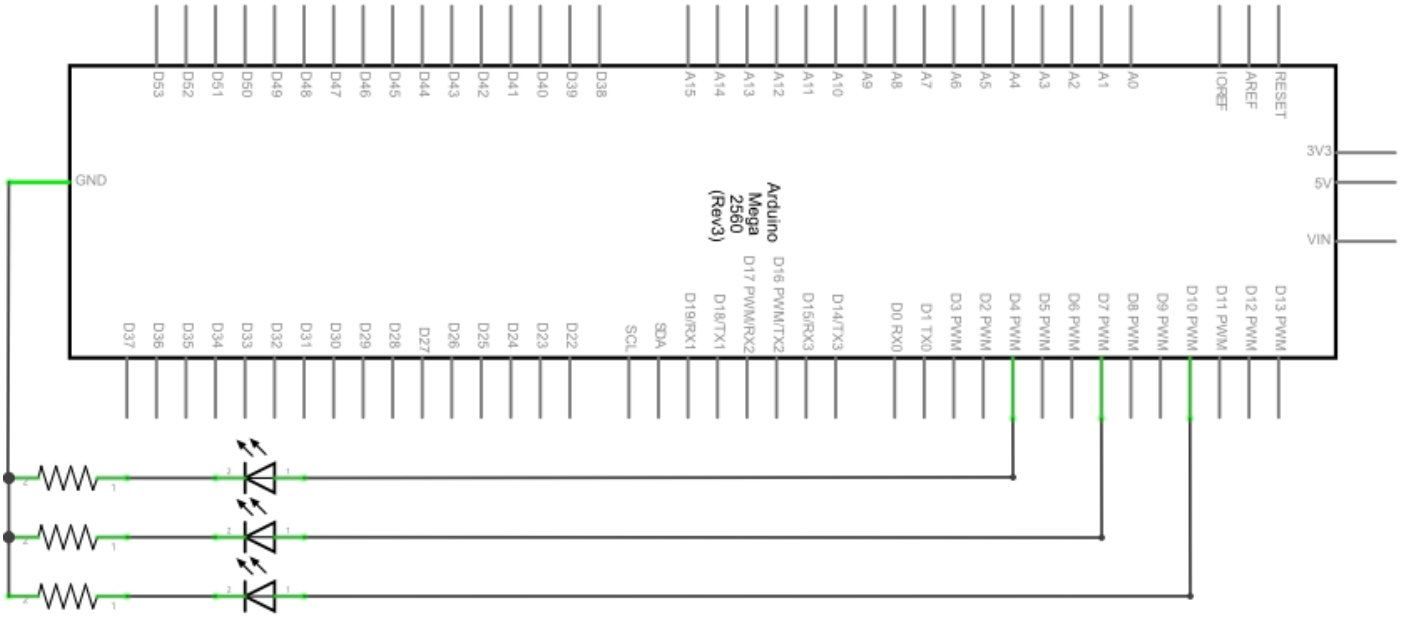
```
int potpin = 0;       // Initialises analog Pin 0
int ledpin = 11;      // Initialises digital Pin 11 (PWM Output)

int val = 0;          // Saves the Sensors Value

void setup()
{
      pinMode(ledpin,OUTPUT);     // defines digital Pin 11 to „Output“
      Serial.begin(9600);         // Sets Baudrate to 9600
}

void loop()
{
      val = analogRead(potpin);
      // reads the Analog-Value of the sensor and assigns it to „Val“
      Serial.println(val);        // shows the value of „Val“
      analogWrite(ledpin,val/4);
      // turns on the LED and sets the brightness (Max. Value: 255)
      delay(10);                  // Waits 0,01 Seconds
}
```

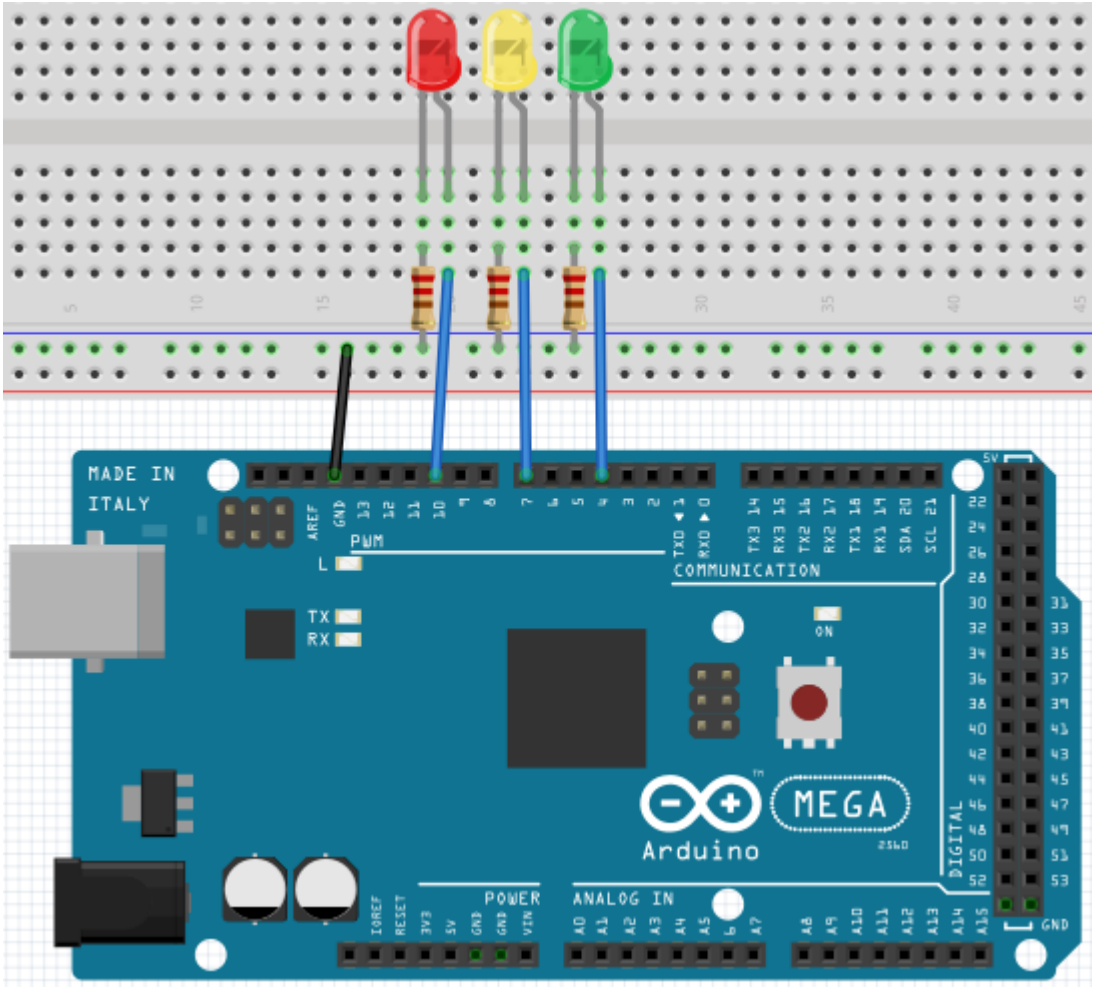
After transmitting the program and moving the potentiometer, we can observe changes in the values displayed. We can also see an obvious change in LED brightness.

4.4 Traffic Lights

In the previous program we performed the flashing LED experiment with an LED.
Now it is time to conduct a more complicated experiment: Traffic lights.
Actually, these two experiments are similar. During this experiment we will use 3 LEDs with different colors, while in the last experiment only one LED was used.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
Red M5 LED	1
Yellow M5 LED	1
Green M5 LED	1
220Ω Resistor	1
Breadboard	1
Breadboard Jumper Cable	3





Since this is a simulation of traffic light, the lighting time of each individual LED should be exactly as long as with real traffic lights. In this program we will use the Arduino delay function to control the delay time.

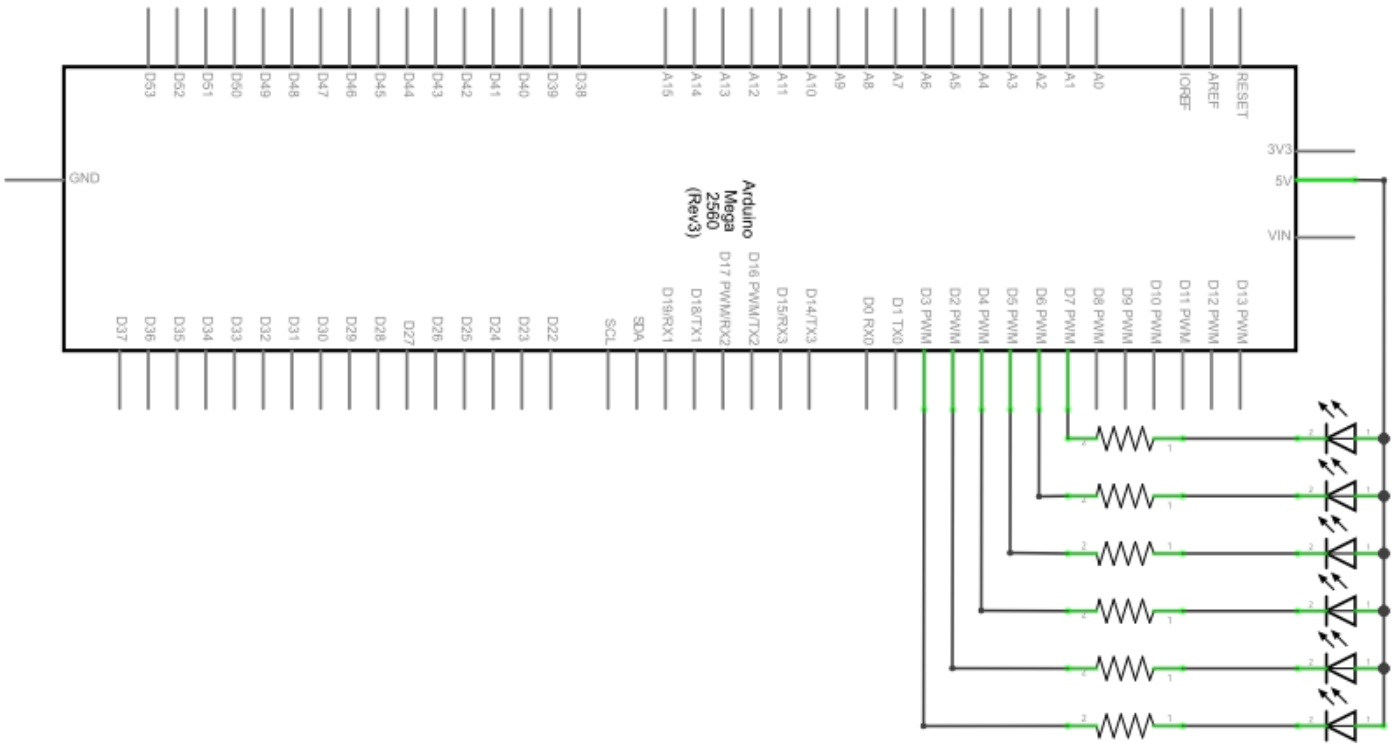
```
int redled =10;             // Initialises digital Pin 8
int yellowled =7;           // Initialises digital Pin 7
int greenled =4;            // Initialises digital Pin 4

void setup()
{
  pinMode(redled, OUTPUT); // Sets Pin with red LED to „Output“
  pinMode(yellowled, OUTPUT);// Sets Pin with yellow to „Output“
  pinMode(greenled, OUTPUT);// Sets Pin with green LED to „Output“
}

void loop()
{
  digitalWrite(greenled, HIGH);  // turns on green leed
  delay(5000);                   // Waits 5 Seconds
  digitalWrite(greenled, LOW);   // turns off green LED
  for(int i=0;i<3;i++)           // flashes 3x
  {
    delay(500);
    digitalWrite(yellowled, HIGH);    // turns on the yellow LED
    delay(500);
    digitalWrite(yellowled, LOW);     // turns on the yellow LED
  }
  delay(500);
  digitalWrite(redled, HIGH);         // turns on the red LED
  delay(5000);
  digitalWrite(redled, LOW);          // turns on the red LED
}
```

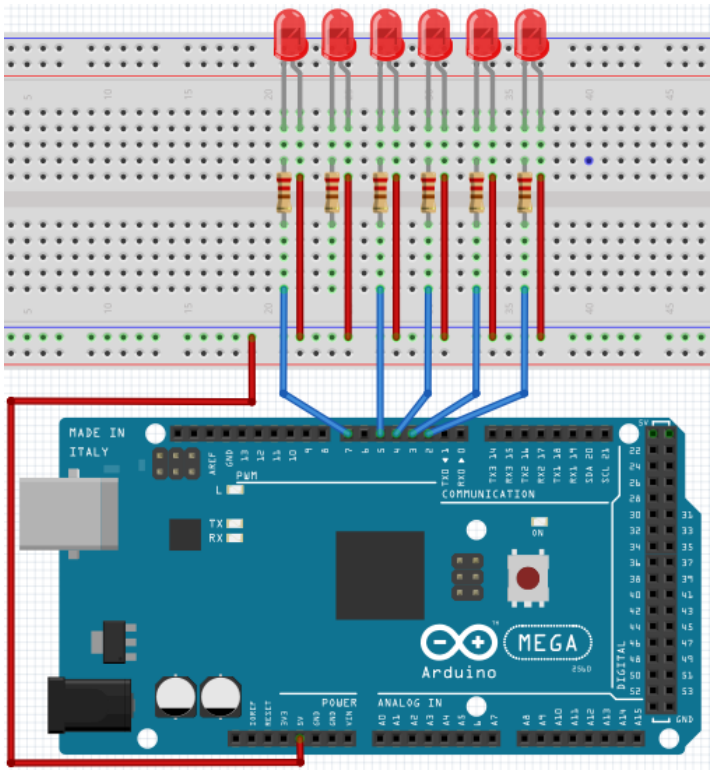
When the upload process is complete, the traffic light is visible.
The green light will light for 5 seconds and then turn off. Then the yellow light will flash 3 times and then the red light will glow for 5 seconds to form a circuit.

4.5 LED Chasing-Effect

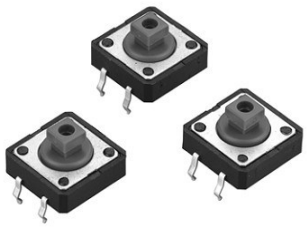
We often see billboards with coloured LEDs. These are constantly changing to form different effects. In this experiment a program is created which simulates the LED hunting effect.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
LED	6
220Ω Resistor	6
Breadboard	1
Breadboard Jumper Cable	12





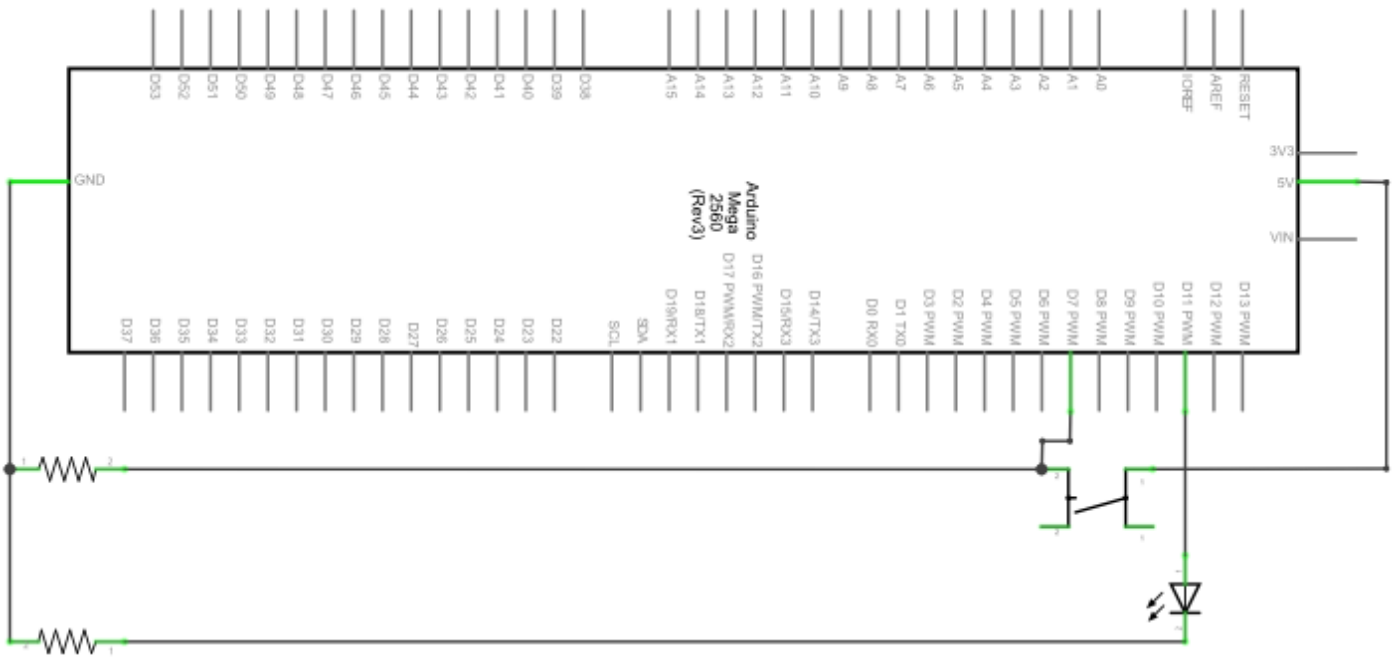

```
int BASE = 2 ;              // I/O Pin for the first LED
int NUM = 6;                // Amount of LEDs
void setup()
{
   for (int i = BASE; i < (BASE + NUM); i ++)
   {
     pinMode(i, OUTPUT);    // Sets I/O Pins to output
   }
}
void loop()
{
   for (int i = BASE; i < (BASE + NUM); i ++)
   {
     digitalWrite(i, LOW);
     // Set I/O Pin to „low“
     // turns on leds one by one
     delay(200);                  // delay
   }
   for (int i = BASE; i < (BASE + NUM); i ++)
   {
     digitalWrite(i, HIGH);
     // Sets I/O Pin to „high“
     // turns off led one by one
     delay(200);                  // delay
   }
}
```

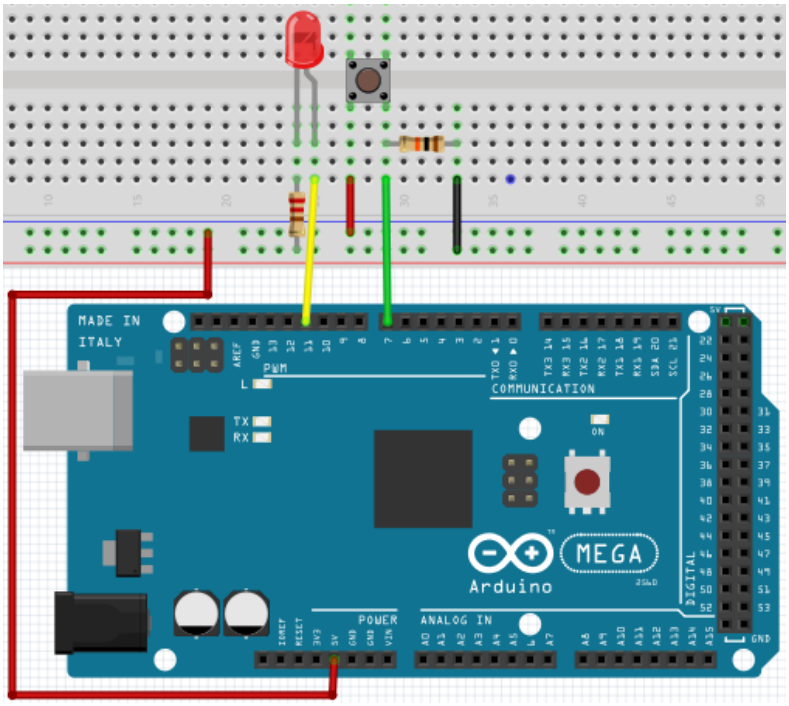


4.6 Button-Controlled LED

I/O port is the interface for INPUT and OUTPUT. So far, we've only used the exit.
In this experiment we will try to use the input to read the output value of the connected device. We will use a button and an LED with input and output to provide a better understanding of the I/O function.
Key switches, most of us know, have a switching value (digital value). When the switch is pressed, the circuit closes and is in a conductive state.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
LED	1
220Ω Resistor	1
10kΩ Resistor	1
Button	1
Breadboard	1
Breadboard Jumper Cable	5





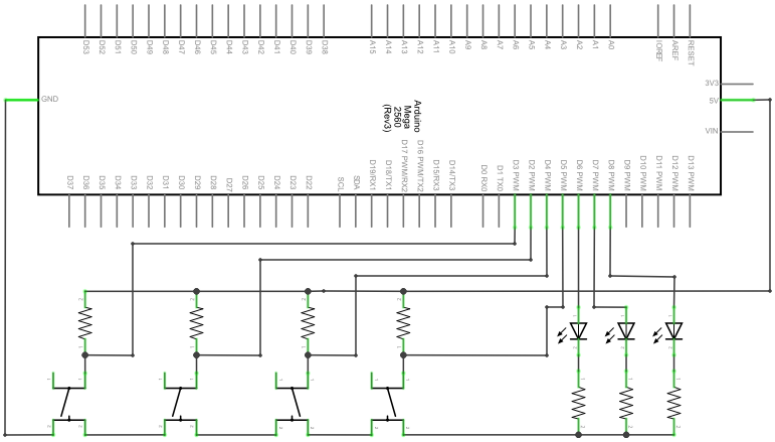

```
int ledpin=11;   // Initialises Pin 11
int inpin=7;     // Initialises Pin 7
int val;         // Definies „Val”
void setup()
{
      pinMode(ledpin,OUTPUT);     // Sets LED Pin to „Output“
      pinMode(inpin,INPUT);       // Sets Button to „Input“
}
void loop()
{
      val=digitalRead(inpin);
      // reads value of pin 7 and assigns to „Val“
      if(val==LOW)
      // check if button is pressed, turn led on if so
      {
            digitalWrite(ledpin,LOW);
      }
      else
      {
            digitalWrite(ledpin,HIGH);
      }
}
```

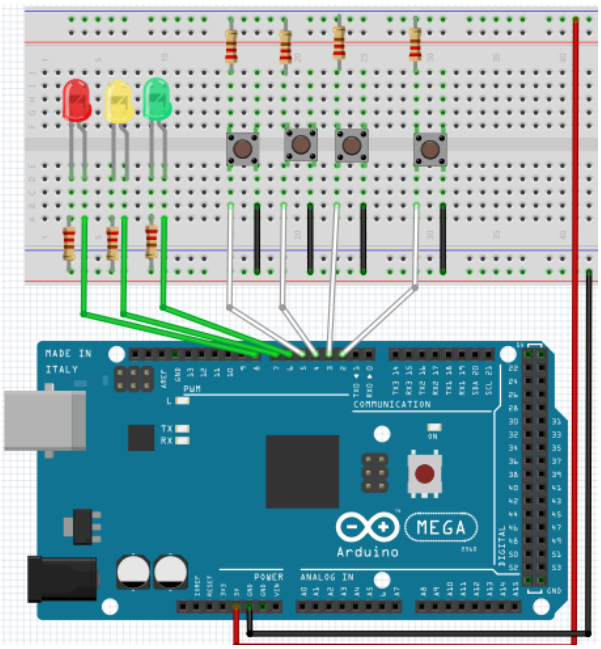
If the button is pressed, the LED lights up, otherwise it remains off.
The simple principle of this experiment is often used in a variety of circuits and electrical devices.
is used.

4.7 Responder Experiment

In this program there are 3 buttons and a reset button, which control the 3 corresponding LEDs with the help of 7 digital I/O PINs.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
Red M5 LED	1
Yellow M5 LED	1
Green M5 LED	1
220Ω Resistor	7
Button	4
Breadboard	1
Breadboard Jumper Cable	13





```
int redled=8;          // Set red LED to „Output“
int yellowled=7;       // Set yellow LED to „Output“
int greenled=6;        // Set green LED to „Output“
int redpin=5;          // Initiale Pin for red Taste
int yellowpin=4;       // Initiale Pin for yellow Button
int greenpin=3;        // Initiale Pin for green Button
int restpin=2;         // Initiale Pin for Reset-Button
int red;
int yellow;
int green;
void setup()
{
      pinMode(redled,OUTPUT);
      pinMode(yellowled,OUTPUT);
      pinMode(greenled,OUTPUT);
      pinMode(redpin,INPUT);
      pinMode(yellowpin,INPUT);
      pinMode(greenpin,INPUT);
}
void loop()            //reads the buttons
{
      red = digitalRead(redpin);
      yellow = digitalRead(yellowpin);
      green = digitalRead(greenpin);
      if(red==LOW)RED_YES();
      if(yellow==LOW)YELLOW_YES();
      if(green==LOW)GREEN_YES();
}

void RED_YES()
// execute code until red led is on
// finishes loop when the reset button is pressed
{
  while(digitalRead(restpin)==1)
  {
      digitalWrite(redled,HIGH);
      digitalWrite(greenled,LOW);
      digitalWrite(yellowled,LOW);
  }
  clear_led();
}
```

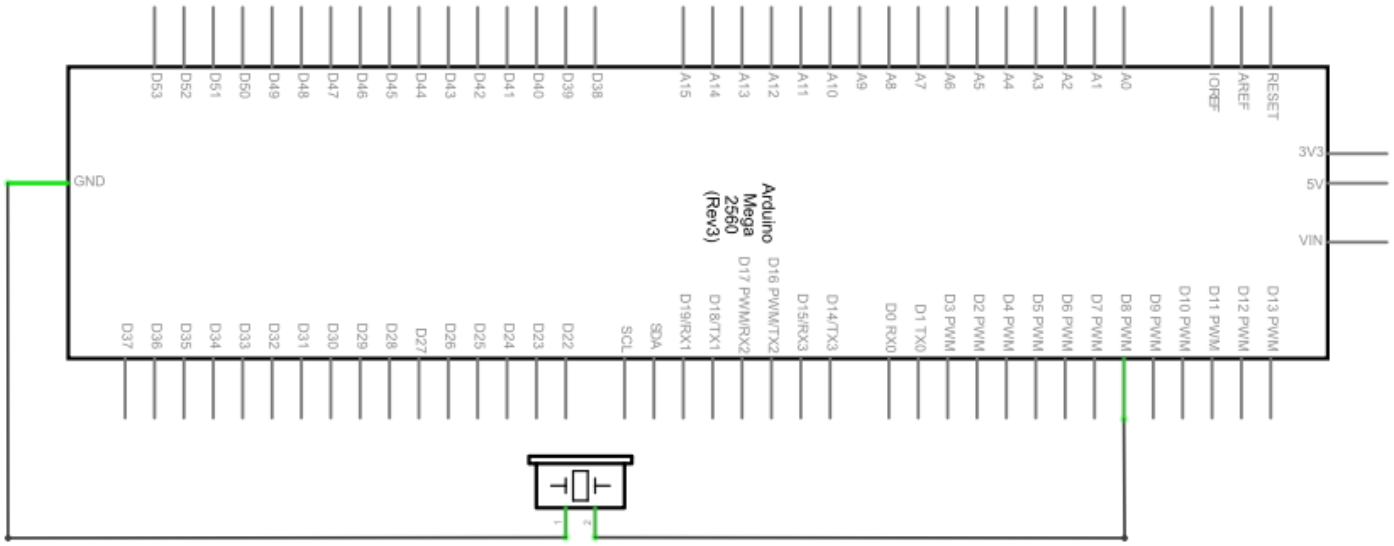
```
void YELLOW_YES()
// execute code until yellow led is on
// finishes loop when the reset button is pressed
{
  while(digitalRead(restpin)==1)
  {
      digitalWrite(redled,LOW);
      digitalWrite(greenled,LOW);
      digitalWrite(yellowled,HIGH);
  }
  clear_led();
}
void GREEN_YES()
// execute code until green led is on
// finishes loop when the reset button is pressed
{
  while(digitalRead(restpin)==1)
  {
      digitalWrite(redled,LOW);
      digitalWrite(greenled,HIGH);
      digitalWrite(yellowled,LOW);
  }
  clear_led();
}
void clear_led()// All LEDs off
{
  digitalWrite(redled,LOW);
  digitalWrite(greenled,LOW);
  digitalWrite(yellowled,LOW);
}
```

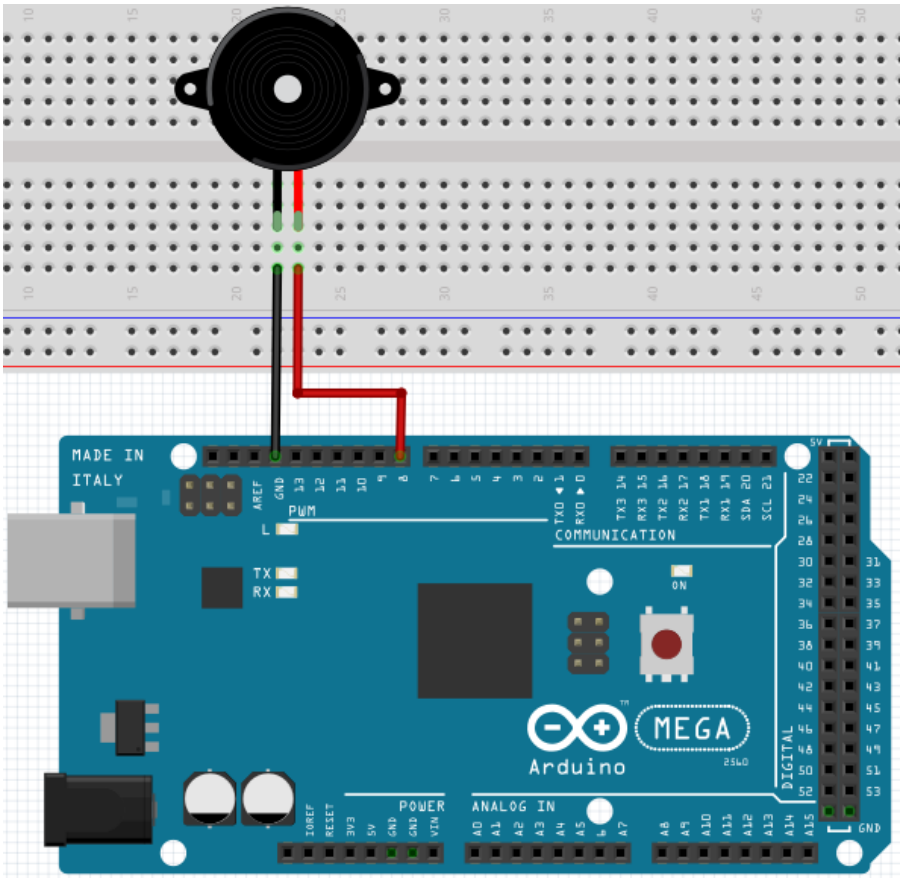
Please make sure that you put both code parts together in your sketch of the Arduino program.
When a key is pressed, the corresponding LED switches on.
If the reset button is pressed, the corresponding LED switches off again.

4.8 Active Buzzer

Active buzzers are used in computers, printers, alarm clocks, electric toys etc. as a noise emitting element. It has an internal vibration source.
Connected to a 5V power supply, it can hum repeatedly.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
Buzzer	1
Breadboard	1
Breadboard Jumper Cable	2

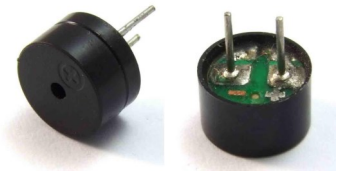





```
int buzzer=8;
// Initialise digital I/O Pin, which controlles the buzzer
void setup()
{
      pinMode(buzzer,OUTPUT);          // sets pin mode to output
}
void loop()
{
      digitalWrite(buzzer, HIGH);            // makes noises
}
```

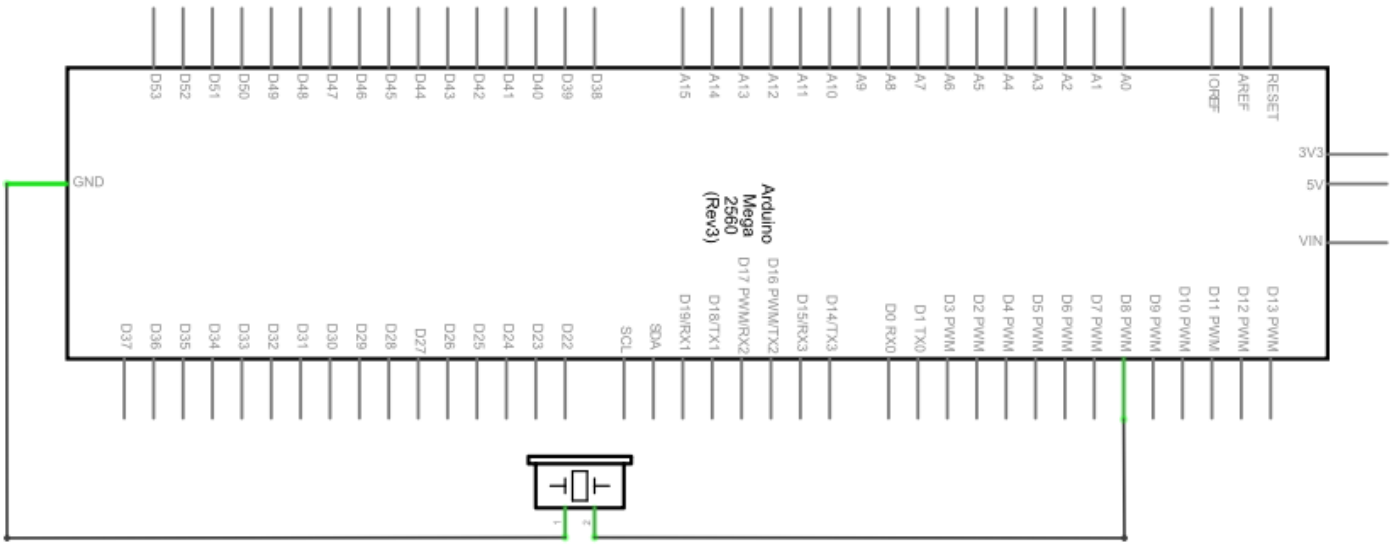
The project is completed after the transfer of the program.
After transmission, the buzzer is supplied with power and emits noise.

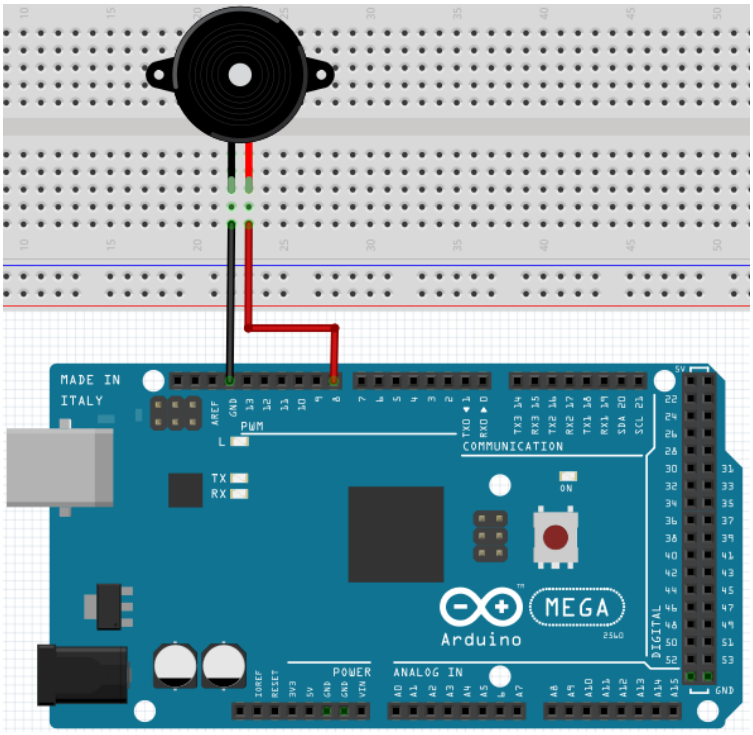
4.9 Passive Buzzer



Many interactive projects are possible with the Mega2560.
The projects so far have mainly dealt with LEDs, but one frequently used project is the acoustic-optical display. For this purpose, a passive buzzer is used which, in the
in contrast to the active buzzer, cannot activate itself.
Activation takes place via a pulse frequency. Different frequencies result in different tones during the buzzer. This can be used, for example, to play the melody of a song.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
Passive Buzzer	1
Breadboard	1
Breadboard Jumper Cable	2





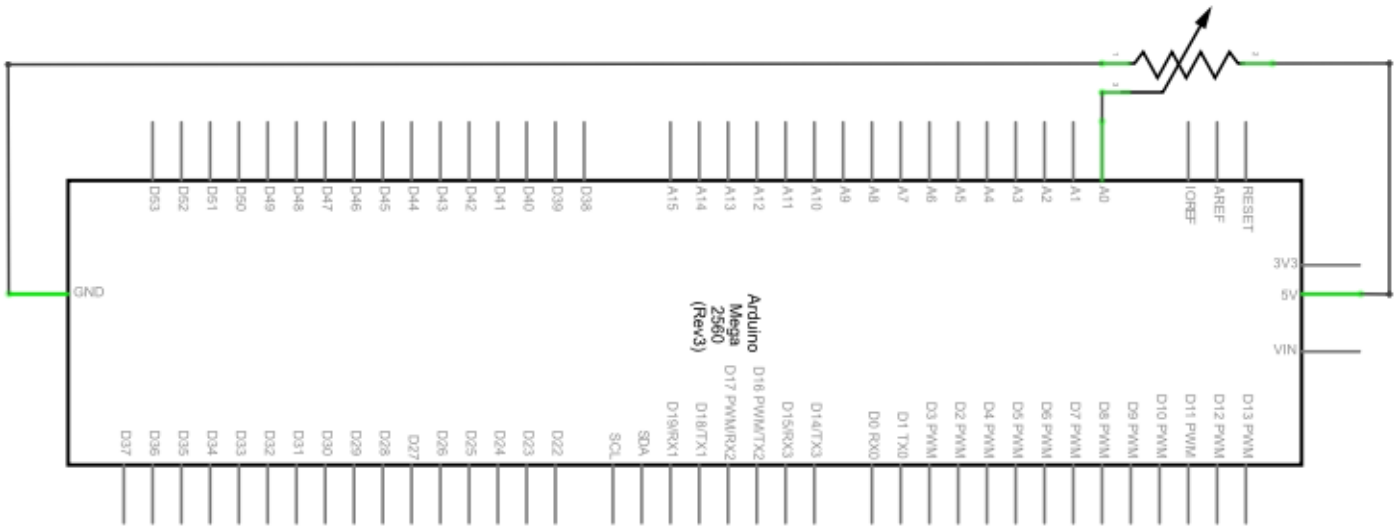

```
int buzzer=8;    // chooses digital I/O Pin for buzzer
void setup()
{
      pinMode(buzzer,OUTPUT); // sets digital IOs to output
}
void loop()
{
      unsigned char i,j;          // defines Variable
      while(1)
      {
            for(i=0;i<80;i++)     // makes frequency sound
            {
                  digitalWrite(buzzer,HIGH); // Sound
                  delay(1);                  // 1ms delay
                  digitalWrite(buzzer,LOW);  // no sound
                  delay(1);                  // 1ms delay
            }
            for(i=0;i<100;i++)               // makes frequency sound
            {
                  digitalWrite(buzzer,HIGH); // sound
                  digitalWrite(buzzer,LOW);  // no sound
                  delay(2);                  // 2ms delay
            }
      }
}
```

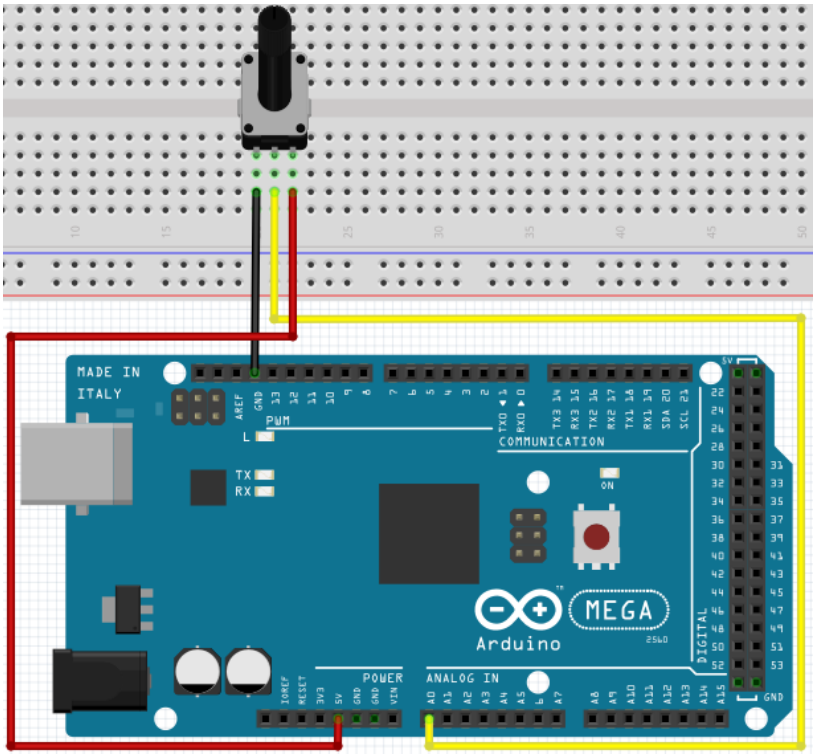
4.10 Read Analog Values

This project is about the analog interfaces of the Mega2560, an analogRead () command can read the value of the interface. Due to the analog-to-digital conversion of the Mega2560, the read-out values are between 0 and 1023.
To be able to read out the values, it is important to pay attention to the correct baud rate. The baud rate of the computer should correspond to the baud rate of the device. If you open the serial monitor of the Arduino program, you can configure the baud rate in the lower right corner.

In this project, the set resistance value of a potentiometer is converted to an analog signal and then output on the screen.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
Potentiometer	1
Breadboard	1
Breadboard Jumper Cable	3





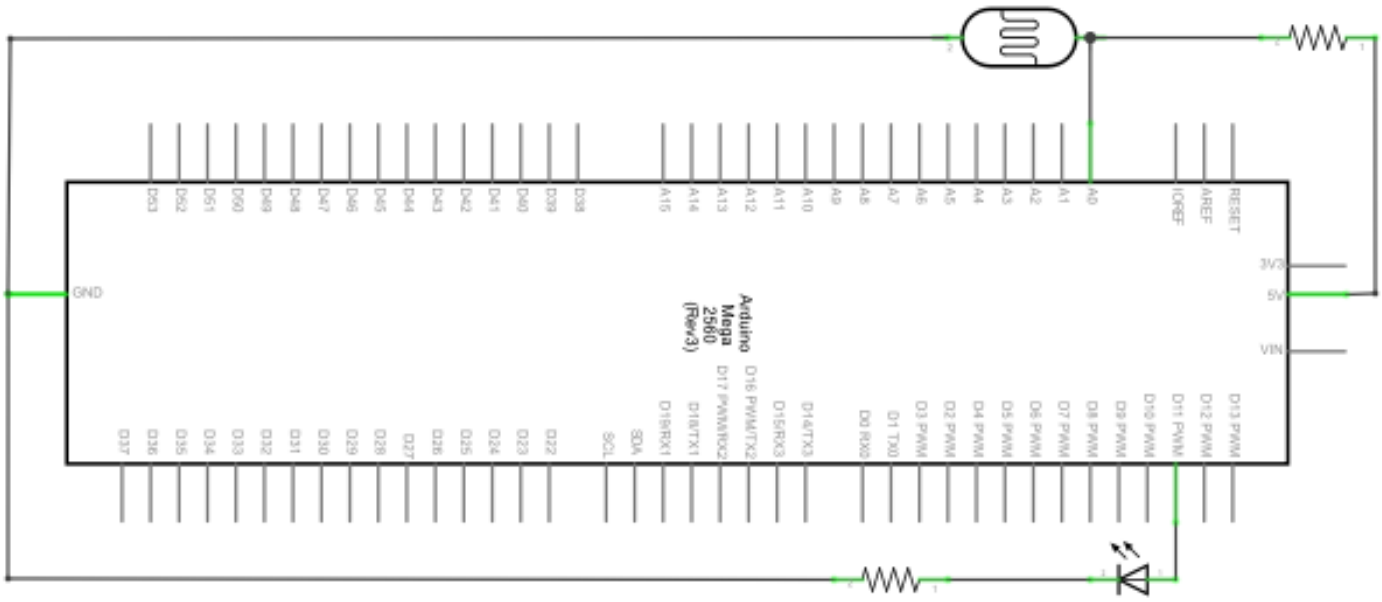

```
int potpin=0;    // Initialises analog Pin 0
int ledpin=13;   // Initialises digital Pin 13
int val=0;       // Definies „Val“
void setup()
{
      pinMode(ledpin,OUTPUT);     // sets digital pin to output
      Serial.begin(9600);         // Set Baudrate to 9600
}
void loop()
{
      digitalWrite(ledpin,HIGH); // turns on led on pin 13
      delay(50);                 // Waits 0,05 Seconds
      digitalWrite(ledpin,LOW);  // turns off led on pin 13
      delay(50);                 // Waits 0,05 Seconds
      val=analogRead(potpin);    // reads analog value and assigns it to val
      Serial.println(val);       // shows value of „Val“
}
```

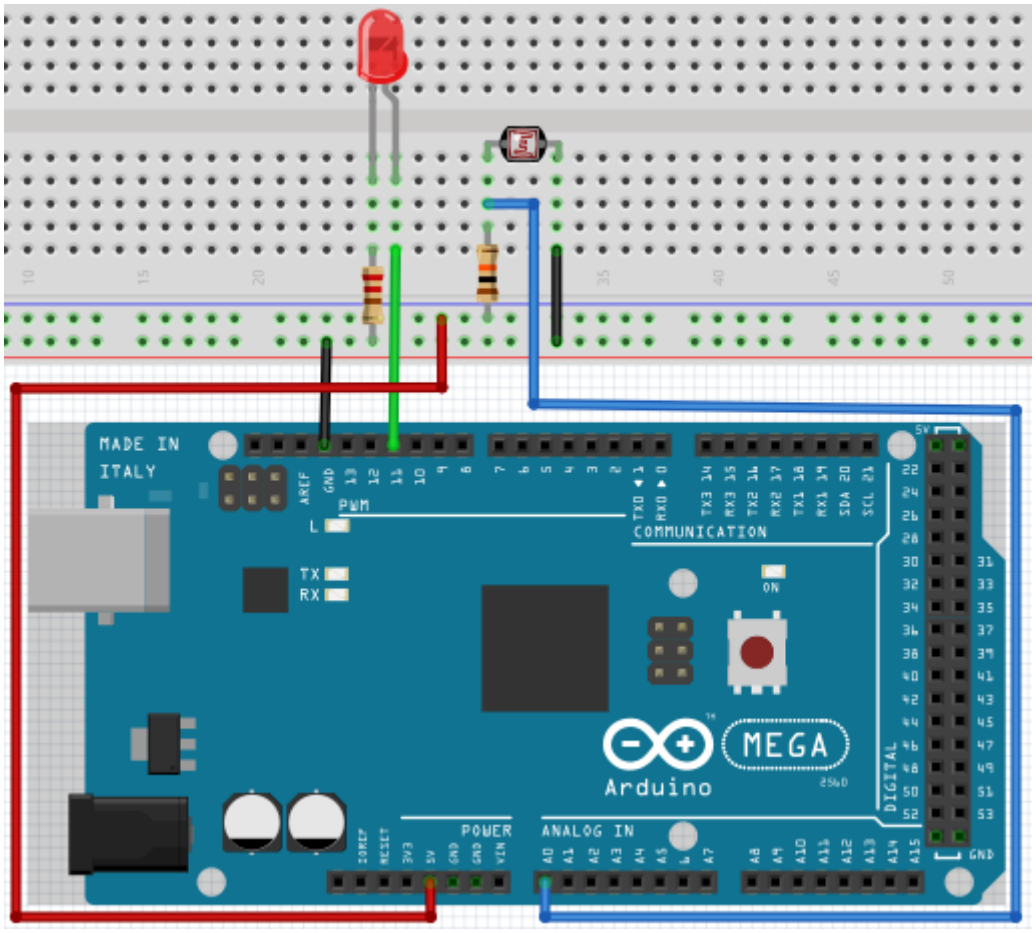
The values read out are displayed in the serial monitor.

4.11 Light Dependent Resistor

A photoresistor is a resistor whose resistance varies depending on the incident light intensity.
It is based on the photoelectric effect of semiconductors. When the incident light is intense, the resistance is reduced. When the incident light is weak, the resistance increases. Photo resistors are normally used for light measurement, light control and photovoltaic conversion (converts the change of light into a change of electricity).
They are used in various light control circuits, e.g. as optical switches.
In this project this effect is used to adjust an LED to the current luminous intensity.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
Red M5 LED	1
Light Dependent Resistor	1
220Ω Resistor	1
10kΩ Resistor	1
Breadboard	1
Breadboard Jumper Cable	5





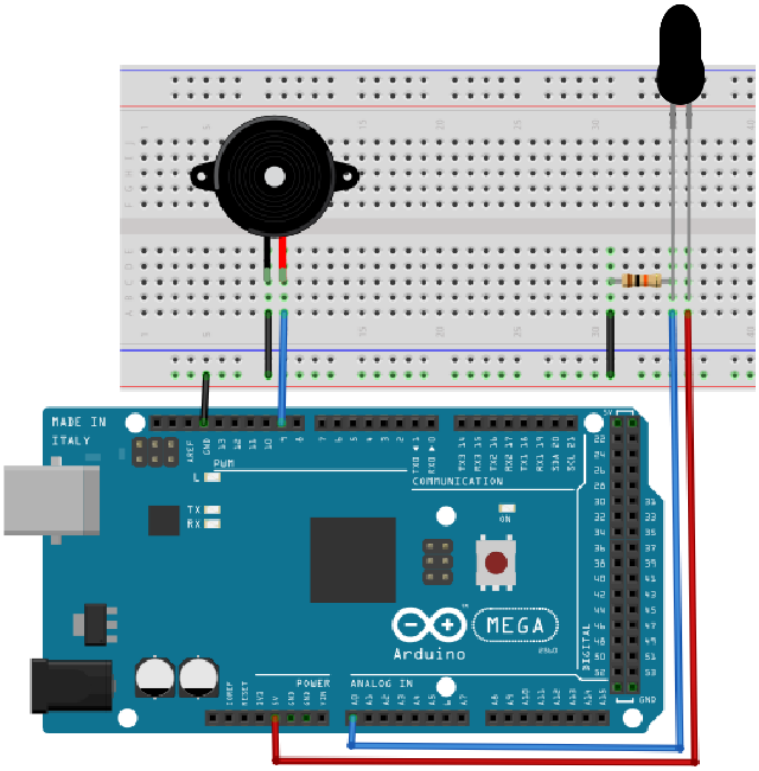

```
int potpin=0;
// Initialises analog Pin 0
int ledpin=11;
// Initialises digital Pin 11
// output which regulates the led brightness

int val=0;       // Initialises Variable „Val“
void setup()
{
      pinMode(ledpin,OUTPUT);     // Set Pin 11 to output
      Serial.begin(9600);         // Set Baudrate to „9600“
}
void loop()
{
      val=analogRead(potpin);
      // reads the sensors values and assigns it to val
      Serial.println(val);          // shows the values of val
      analogWrite(ledpin,val);
      // turns on the led and sets the brightness
      delay(10);                    // Waits 0,01 Seconds
}
```

4.12 Flame Sensor

The flame sensor (infrared receiving triode) is especially used on robots to find flame sources. This sensor has a high sensitivity to flames.
The flame sensor is based on the principle that infrared radiation is very sensitive to fire. It has a specially designed infrared tube to detect fire and then convert the brightness of the flame into a signal. These signals are then transmitted to the central processor and processed accordingly.

When the sensor approaches a fire, the analog voltage value changes. A multimeter can be used to check that the voltage is around 0.3V when no fire is approaching.
When a fire approaches, the voltage is about 1.0V. The higher the voltage, the closer the fire.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
Flame Sensor	1
Buzzer	1
10kΩ Resistor	1
Breadboard	1
Breadboard Jumper Cable	6

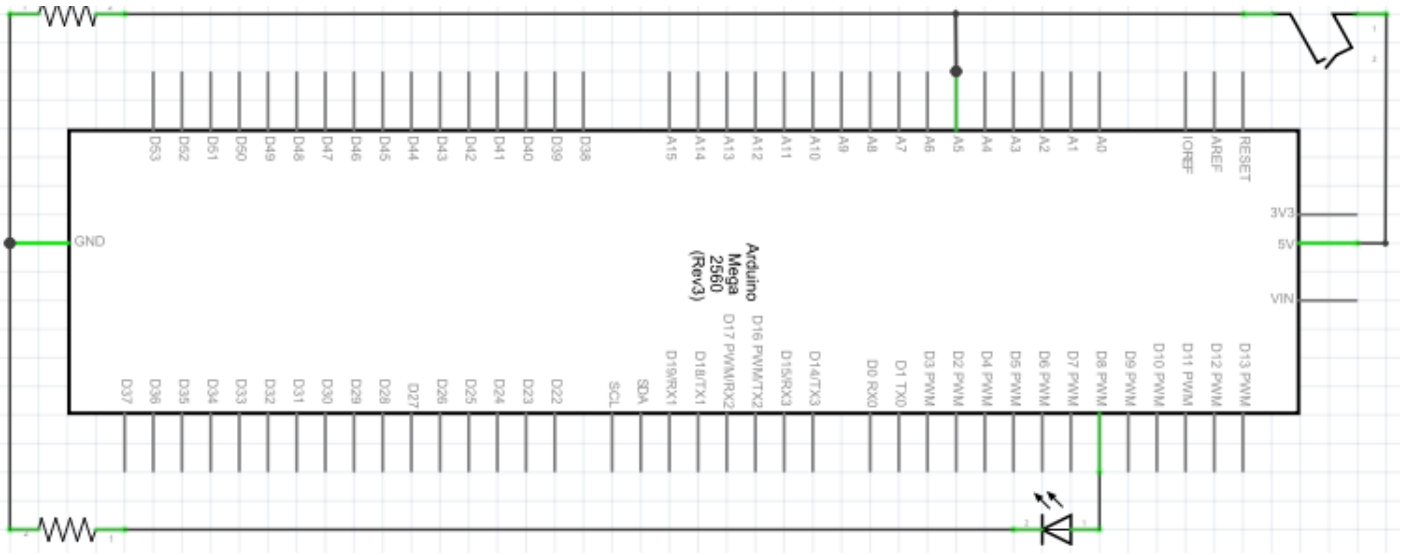


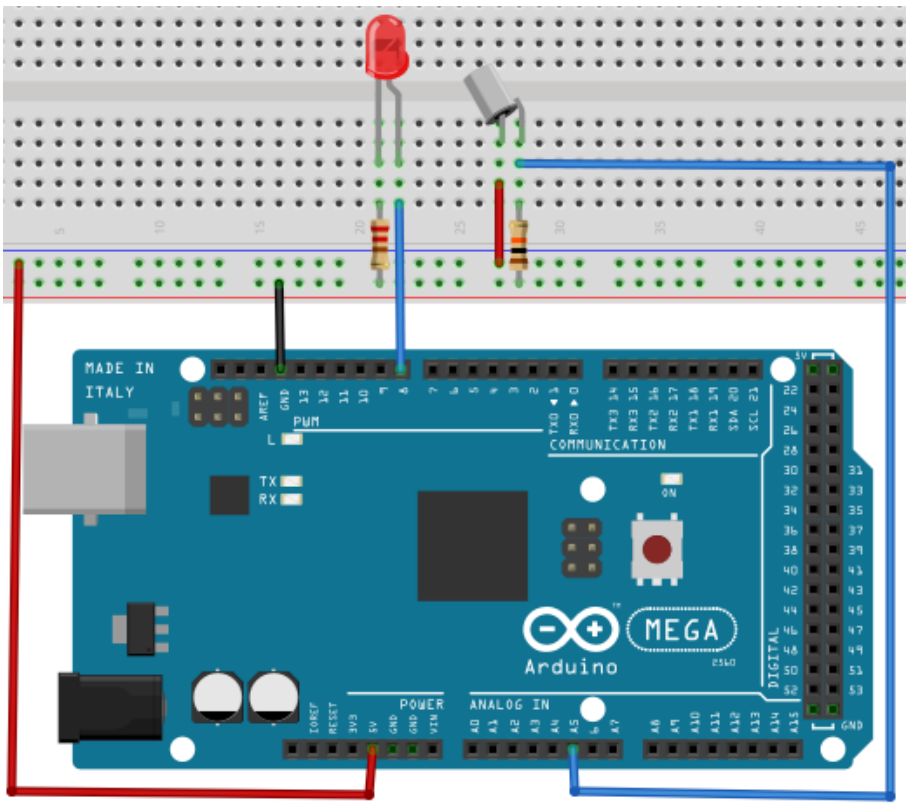

```
int flame=0;     // analog Pin 0 for Sensor
int Beep=9;      // digital Pin 9 for Buzzer
int val=0;       // Initialises Variable
void setup()
{
      pinMode(Beep,OUTPUT);       // sets buzzer pin to output
      pinMode(flame,INPUT);       // sets sensor pin to input
      Serial.begin(9600);         // Sets Baudrate to „9600“
}
void loop()
{
      val=analogRead(flame);      // reads the sensors values
      Serial.println(val);        // prints the analog values
      if(val>=600)
      // buzzer makes noises if analog values are above 600
      {
            digitalWrite(Beep,HIGH);
      }
      else
      {
            digitalWrite(Beep,LOW);
      }
      delay(500);
}
```

4.13 Tilt Switch

The position switch controls the ON/OFF switch of an LED. The switch is on when one end of the switch is below the horizontal position. The voltage value of the analog port to which the position switch is connected can be used to check the position of the switch.
is located.

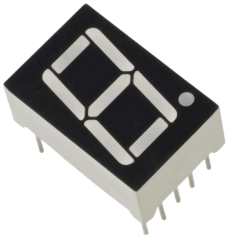
Hardware	Amount
Mega2560 Board	1
USB Cable	1
Tilt Switch	1
Red M5 LED	1
220Ω Resistor	1
Breadboard	1
Breadboard Jumper Cable	5





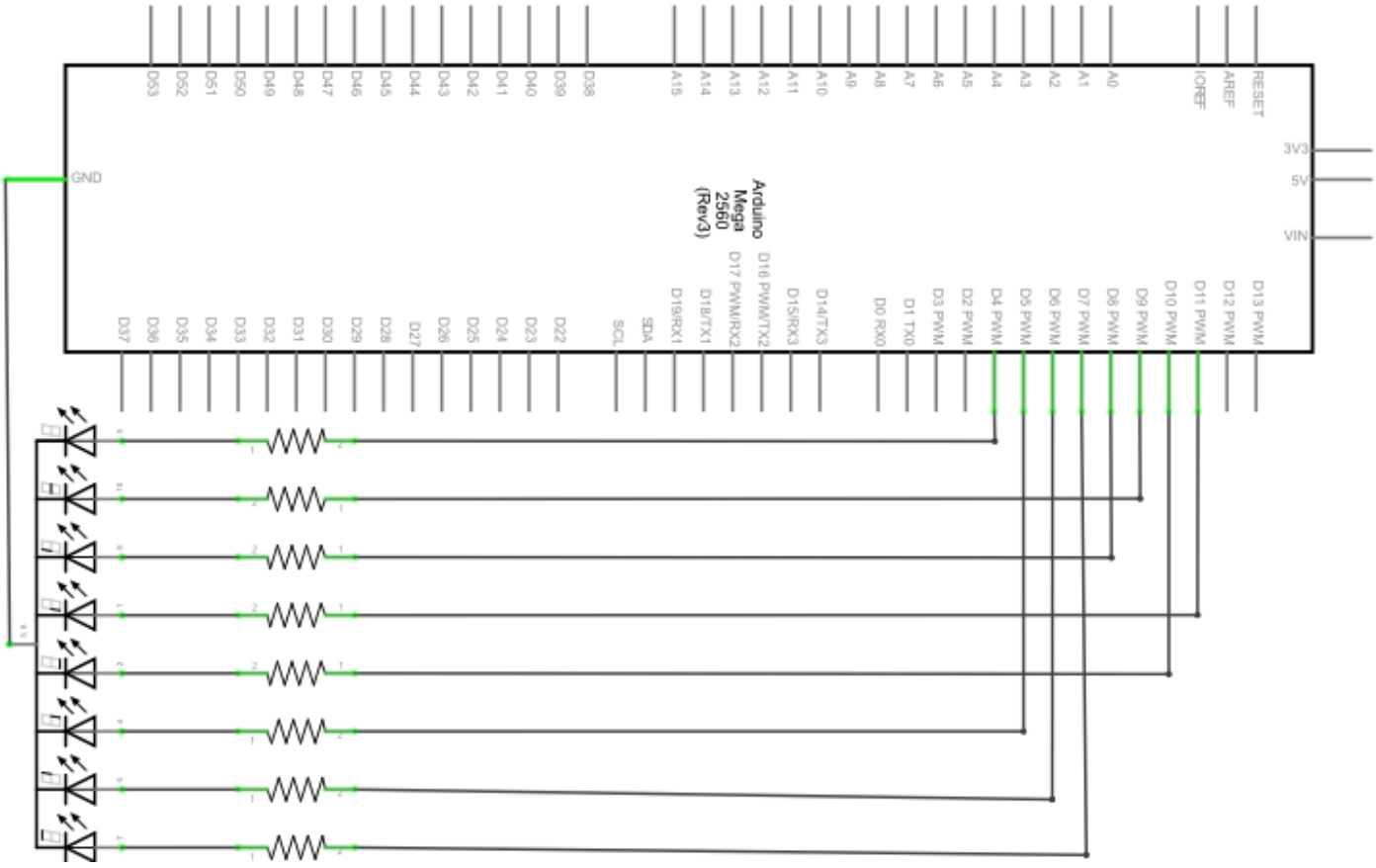

```
void setup()
{
      pinMode(8,OUTPUT);    // sets digital Pin 8 to „output“
}

void loop()
{
      int i;                // Defines Variable i
      while(1)
      {
            i=analogRead(5);// reads voltage value of the analog pin 5
            if(i>512)       // if bigger than 512 (2.5V)
            {
                  digitalWrite(8,LOW);  // turn on LED
            }
            else
            {
                  digitalWrite(8,HIGH); // turn off LED
            }
      }
}
```

4.14 1-Digit LED Segment-Display

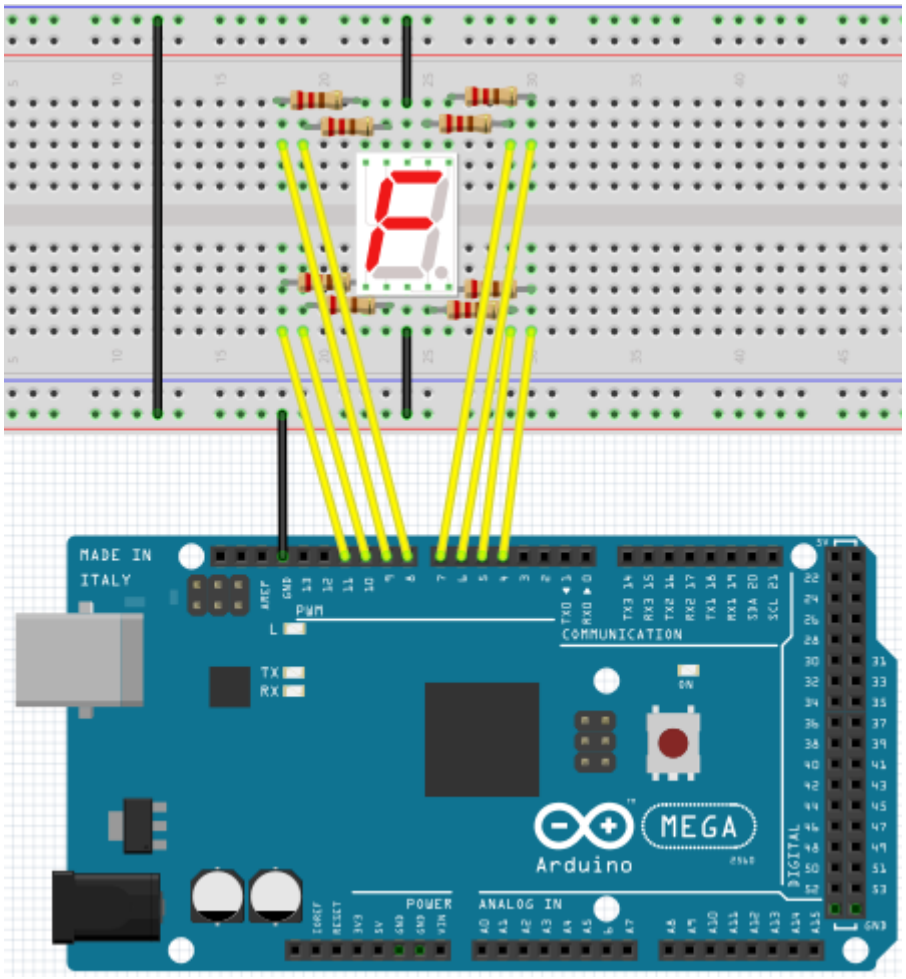


LED segment displays are widely used for displaying numerical information.
They are often used in displays of electromagnetic ovens, fully automatic washing machines, water temperature displays, electronic clocks, etc. The LED segment display is a semiconductor and a light emitting device. Its base unit is an LED. The LED segment display can be divided into the 7-segment and the 8-segment display. The 8-segment display has one LED unit more (for the decimal point display) than the 7-segment display.

Depending on the wiring of the LED units, the LED segment display can be divided into displays with common anode and displays with common cathode. The display with common anode combines all anodes of the LED units into one common anode (COM). For the common anode display, the common anode (COM) must be connected to +5V. When the cathode level of a segment is low, the segment is on. If the cathode level of a segment is high, the segment is off. For the common cathode display, the common cathode (COM) must be connected to GND. When the anode level of a segment is high, the segment is on. If the anode level of a segment is low, the segment is off.



Hardware	Amount
Mega2560 Board	1
USB Cable	1
8-Segment-Display	1
220Ω Resistor	8
Breadboard	1
Breadboard Jumper-Cable	12



```
// Sets the IO Pin for every segment
int a=7;               // Sets digital Pin 7 for Segment a
int b=6;               // Sets digital Pin 6 for Segment b
int c=5;               // Sets digital Pin 5 for Segment c
int d=10;              // Sets digital Pin 10 for Segment d
int e=11;              // Sets digital Pin 11 for Segment e
int f=8;               // Sets digital Pin 8 for Segment f
int g=9;               // Sets digital Pin 9 for Segment g
int dp=4;              // Sets digital Pin 4 for Segment dp

void digital_0(void)  // shows number 5
{
  unsigned char j;
  digitalWrite(a,HIGH);
  digitalWrite(b,HIGH);
  digitalWrite(c,HIGH);
  digitalWrite(d,HIGH);
  digitalWrite(e,HIGH);
  digitalWrite(f,HIGH);
  digitalWrite(g,LOW);
  digitalWrite(dp,LOW);
}

void digital_1(void)        // shows number 1
{
  unsigned char j;
  digitalWrite(c,HIGH);     // sets Pin 5 to „high“
  digitalWrite(b,HIGH);     // turns off Segment b
  for(j=7;j<=11;j++)        // turns off othere segments
  digitalWrite(j,LOW);
  digitalWrite(dp,LOW);     // turns off Segment dp
}

void digital_2(void)        // shows number 2
{
  unsigned char j;
  digitalWrite(b,HIGH);
  digitalWrite(a,HIGH);
  for(j=9;j<=11;j++)
  digitalWrite(j,HIGH);
  digitalWrite(dp,LOW);
  digitalWrite(c,LOW);
  digitalWrite(f,LOW);
}
```

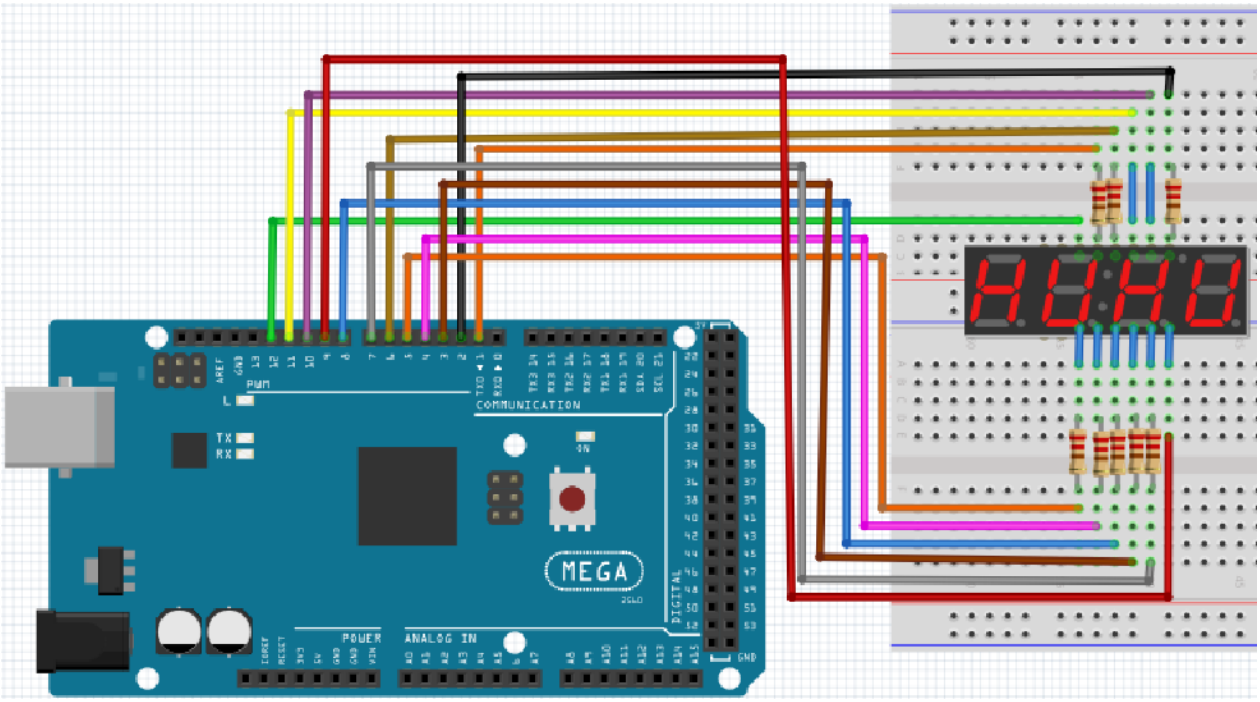
```
void digital_3(void)        // shows number 3
{     digitalWrite(g,HIGH);
      digitalWrite(a,HIGH);
      digitalWrite(b,HIGH);
      digitalWrite(c,HIGH);
      digitalWrite(d,HIGH);
      digitalWrite(dp,LOW);
      digitalWrite(f,LOW);
      digitalWrite(e,LOW);
}
void digital_4(void)        // shows number 4
{     digitalWrite(c,HIGH);
      digitalWrite(b,HIGH);
      digitalWrite(f,HIGH);
      digitalWrite(g,HIGH);
      digitalWrite(dp,LOW);
      digitalWrite(a,LOW);
      digitalWrite(e,LOW);
      digitalWrite(d,LOW);
}
void digital_5(void)        // shows number 5
{
      unsigned char j;
      digitalWrite(a,HIGH);
      digitalWrite(b, LOW);
      digitalWrite(c,HIGH);
      digitalWrite(d,HIGH);
      digitalWrite(e, LOW);
      digitalWrite(f,HIGH);
      digitalWrite(g,HIGH);
      digitalWrite(dp,LOW);
}
void digital_6(void)        // shows number 6
{
      unsigned char j;
      for(j=7;j<=11;j++)
      digitalWrite(j,HIGH);
      digitalWrite(c,HIGH);
      digitalWrite(dp,LOW);
      digitalWrite(b,LOW);
}
void digital_7(void)        // shows number 7
{
      unsigned char j;
      for(j=5;j<=7;j++)
      digitalWrite(j,HIGH);
      digitalWrite(dp,LOW);
      for(j=8;j<=11;j++)
      digitalWrite(j,LOW);
}
```

```
void digital_8(void)        // shows number 8
{
      unsigned char j;
      for(j=5;j<=11;j++)
      digitalWrite(j,HIGH);
      digitalWrite(dp,LOW);
}
void digital_9(void)        // shows number 9
{
      unsigned char j;
      digitalWrite(a,HIGH);
      digitalWrite(b,HIGH);
      digitalWrite(c,HIGH);
      digitalWrite(d,HIGH);
      digitalWrite(e, LOW);
      digitalWrite(f,HIGH);
      digitalWrite(g,HIGH);
      digitalWrite(dp,LOW);
}
void setup()
{
      int i;                        // declares a variable
      for(i=4;i<=11;i++)
      pinMode(i,OUTPUT);            // sets Pin 4-11 to output
}
void loop()
{
      while(1)
      {
            digital_0();                        // shows number 0
            delay(1000);                        // waits a second
            digital_1();                        // shows number 1
            delay(1000);                        // waits a second
            digital_2();                        // shows number 2
            delay(1000);                        // waits a second
            digital_3();                        // shows number 3
            delay(1000);                        // waits a second
            digital_4();                        // shows number 4
            delay(1000);                        // waits a second
            digital_5();                        // shows number 5
            delay(1000);                        // waits a second
            digital_6();                        // shows number 6
            delay(1000);                        // waits a second
            digital_7();                        // shows number 7
            delay(1000);                        // waits a second
            digital_8();                        // shows number 8
            delay(1000);                        // waits a second
            digital_9();                        // shows number 9
            delay(1000);                        // waits a second
      }
}
```

4.15 4-Digit LED Segment-Display

In this project a 4-digit 7-segment display is operated. Current limiting resistors are indispensable for LED displays. There are two wiring methods for current limiting resistors. In the first one, a resistor is connected to each anode, i.e. 4 in total for the d1-d4 anode. An advantage of this method is that it requires less resistance, only 4 pieces. But this method cannot obtain constant brightness.
The second method is to connect a resistor to each PIN.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
4-Digit 8-Segment-Display	1
220Ω Resistor	8
Breadboard	1
Breadboard Jumper Cable	12



```
// PIN for Anode
int a = 1;
int b = 2;
int c = 3;
int d = 4;
int e = 5;
int f = 6;
int g = 7;
int dp = 8;

// PIN for Kathode
int d4 = 9;
int d3 = 10;
int d2 = 11;
int d1 = 12;

// sets Variable
long n = 1230;
int x = 100;
int del = 55;

void setup()
{
      pinMode(d1, OUTPUT);
      pinMode(d2, OUTPUT);
      pinMode(d3, OUTPUT);
      pinMode(d4, OUTPUT);
      pinMode(a, OUTPUT);
      pinMode(b, OUTPUT);
      pinMode(c, OUTPUT);
      pinMode(d, OUTPUT);
      pinMode(e, OUTPUT);
      pinMode(f, OUTPUT);
      pinMode(g, OUTPUT);
      pinMode(dp, OUTPUT);
}
void loop()
{
      Display(1, 1);
      Display(2, 2);
      Display(3, 3);
      Display(4, 4);
}
```

```
void WeiXuan(unsigned char n)//
{
    switch(n)
    {
      case 1:
            digitalWrite(d1,LOW);
            digitalWrite(d2, HIGH);
            digitalWrite(d3, HIGH);
            digitalWrite(d4, HIGH);
            break;
      case 2:
            digitalWrite(d1, HIGH);
            digitalWrite(d2, LOW);
            digitalWrite(d3, HIGH);
            digitalWrite(d4, HIGH);
            break;
      case 3:
           digitalWrite(d1,HIGH);
            digitalWrite(d2, HIGH);
            digitalWrite(d3, LOW);
            digitalWrite(d4, HIGH);
            break;
      case 4:
            digitalWrite(d1, HIGH);
            digitalWrite(d2, HIGH);
            digitalWrite(d3, HIGH);
            digitalWrite(d4, LOW);
            break;
       default :
            digitalWrite(d1, HIGH);
            digitalWrite(d2, HIGH);
            digitalWrite(d3, HIGH);
            digitalWrite(d4, HIGH);
            break;
      }
}
void Num_0()
{
      digitalWrite(a, HIGH);
      digitalWrite(b, HIGH);
      digitalWrite(c, HIGH);
      digitalWrite(d, HIGH);
      digitalWrite(e, HIGH);
      digitalWrite(f, HIGH);
      digitalWrite(g, LOW);
      digitalWrite(dp,LOW);
}
```

```
void Num_1()
{
      digitalWrite(a, LOW);
      digitalWrite(b, HIGH);
      digitalWrite(c, HIGH);
      digitalWrite(d, LOW);
      digitalWrite(e, LOW);
      digitalWrite(f, LOW);
      digitalWrite(g, LOW);
      digitalWrite(dp,LOW);
}
void Num_2()
{
      digitalWrite(a, HIGH);
      digitalWrite(b, HIGH);
      digitalWrite(c, LOW);
      digitalWrite(d, HIGH);
      digitalWrite(e, HIGH);
      digitalWrite(f, LOW);
      digitalWrite(g, HIGH);
      digitalWrite(dp,LOW);
}
void Num_3()
{
      digitalWrite(a, HIGH);
      digitalWrite(b, HIGH);
      digitalWrite(c, HIGH);
      digitalWrite(d, HIGH);
      digitalWrite(e, LOW);
      digitalWrite(f, LOW);
      digitalWrite(g, HIGH);
      digitalWrite(dp,LOW);
}
void Num_4()
{
      digitalWrite(a, LOW);
      digitalWrite(b, HIGH);
      digitalWrite(c, HIGH);
      digitalWrite(d, LOW);
      digitalWrite(e, LOW);
      digitalWrite(f, HIGH);
      digitalWrite(g, HIGH);
      digitalWrite(dp,LOW);
}
```

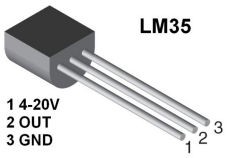
```
void Num_5()
{
      digitalWrite(a, HIGH);
      digitalWrite(b, LOW);
      digitalWrite(c, HIGH);
      digitalWrite(d, HIGH);
      digitalWrite(e, LOW);
      digitalWrite(f, HIGH);
      digitalWrite(g, HIGH);
      digitalWrite(dp,LOW);
}
void Num_6()
{
      digitalWrite(a, HIGH);
      digitalWrite(b, LOW);
      digitalWrite(c, HIGH);
      digitalWrite(d, HIGH);
      digitalWrite(e, HIGH);
      digitalWrite(f, HIGH);
      digitalWrite(g, HIGH);
      digitalWrite(dp,LOW);
}
void Num_7()
{
      digitalWrite(a, HIGH);
      digitalWrite(b, HIGH);
      digitalWrite(c, HIGH);
      digitalWrite(d, LOW);
      digitalWrite(e, LOW);
      digitalWrite(f, LOW);
      digitalWrite(g, LOW);
      digitalWrite(dp,LOW);
}
void Num_8()
{
      digitalWrite(a, HIGH);
      digitalWrite(b, HIGH);
      digitalWrite(c, HIGH);
      digitalWrite(d, HIGH);
      digitalWrite(e, HIGH);
      digitalWrite(f, HIGH);
      digitalWrite(g, HIGH);
      digitalWrite(dp,LOW);
}
```

```
void Num_9()
{
      digitalWrite(a, HIGH);
      digitalWrite(b, HIGH);
      digitalWrite(c, HIGH);
      digitalWrite(d, HIGH);
      digitalWrite(e, LOW);
      digitalWrite(f, HIGH);
      digitalWrite(g, HIGH);
      digitalWrite(dp,LOW);
}
void Clear()               // clears screen
{
      digitalWrite(a, LOW);
      digitalWrite(b, LOW);
      digitalWrite(c, LOW);
      digitalWrite(d, LOW);
      digitalWrite(e, LOW);
      digitalWrite(f, LOW);
      digitalWrite(g, LOW);
      digitalWrite(dp,LOW);
}
void pickNumber(unsigned char n) // chooses number
{
      switch(n)
      {
            case 0:Num_0();
                  break;
            case 1:Num_1();
                  break;
            case 2:Num_2();
                  break;
            case 3:Num_3();
                  break;
            case 4:Num_4();
                  break;
            case 5:Num_5();
                  break;
            case 6:Num_6();
                  break;
            case 7:Num_7();
                  break;
            case 8:Num_8();
                  break;
            case 9:Num_9();
                  break;
            default:Clear();
                  break;
      }
}
```

```
void Display(unsigned char x, unsigned char Number)
{
      WeiXuan(x);
      pickNumber(Number);
      delay(1);
      Clear() ;             // clears screen
}
```

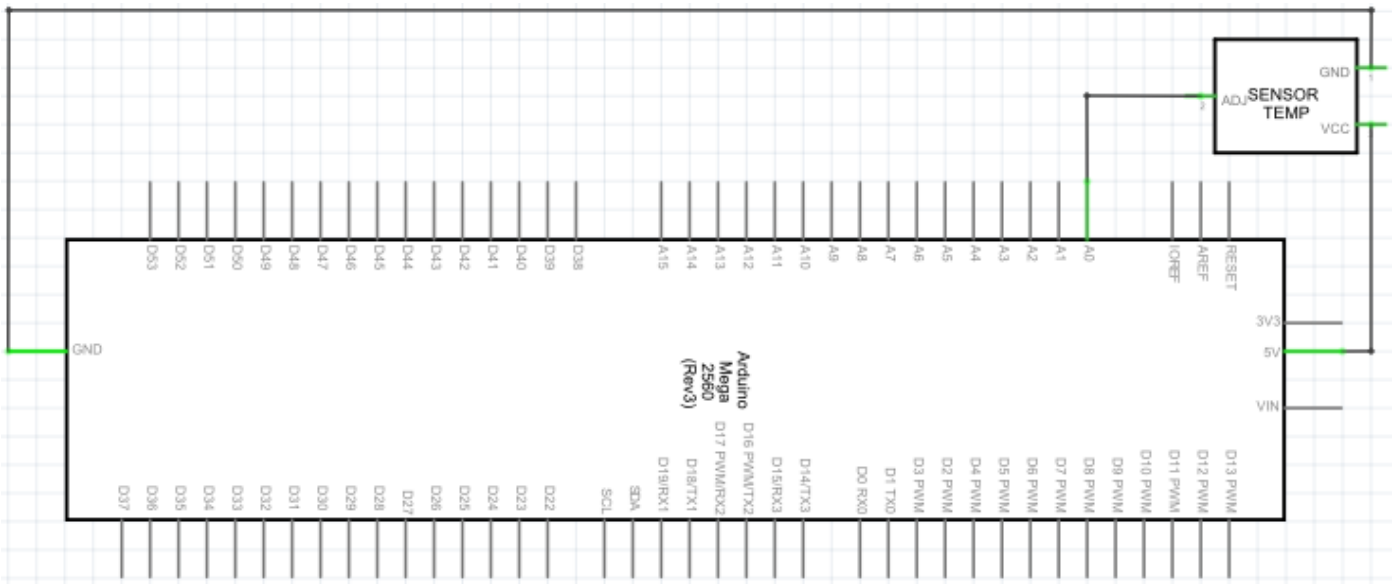
If the above code is transferred completely to the Mega2560, the display shows "1234".

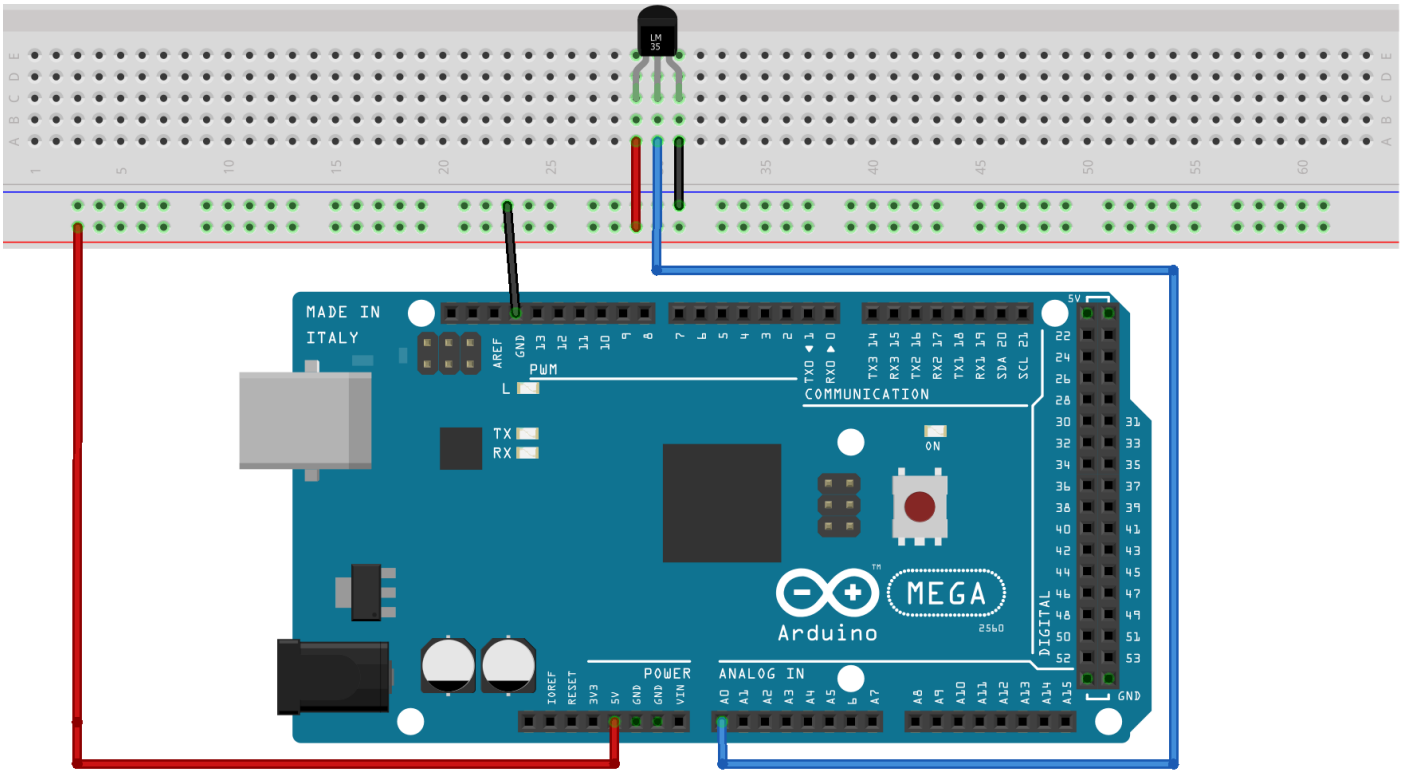
4.16 LM35 Temperature Sensor



The LM35 is a common and easy to use temperature sensor.
You don't need any other hardware. The only difficulty is writing the code that converts the analog values it reads to Celsius temperature.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
LM35	1
Breadboard	1
Breadboard Jumper Cable	12

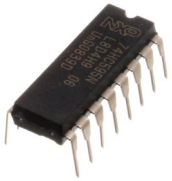





```
int potPin = 0;                    // Initialises Port A0 for Sensor
void setup()
{
      Serial.begin(9600);          // sets Baudrate to „9600“
}
void loop()
{
      int val;                     // Defines Variable
      int dat;                     // Defines Variable
      val=analogRead(0);           // reads Analog value from Sensor
      dat=(125*val)>>8;            // temperature calculation
      Serial.print("Temp:");       // prints „temp:“
      Serial.print(dat);           // prints the temperature
      Serial.println(" C");        // prints the character „C“
      delay(500);                  // Waits 0,5 Seconds
}
```

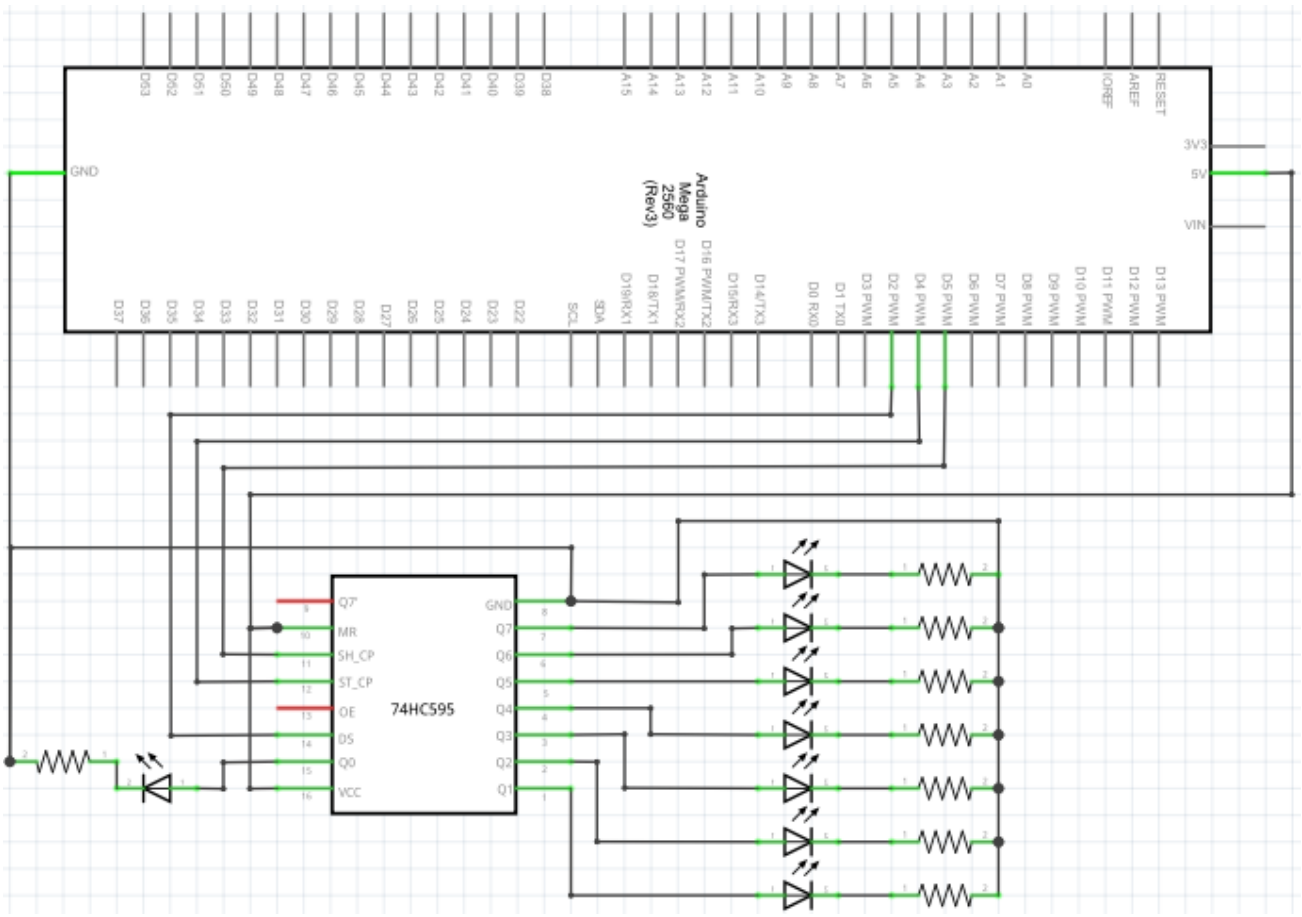
The temperature output can now be monitored on the serial monitor.

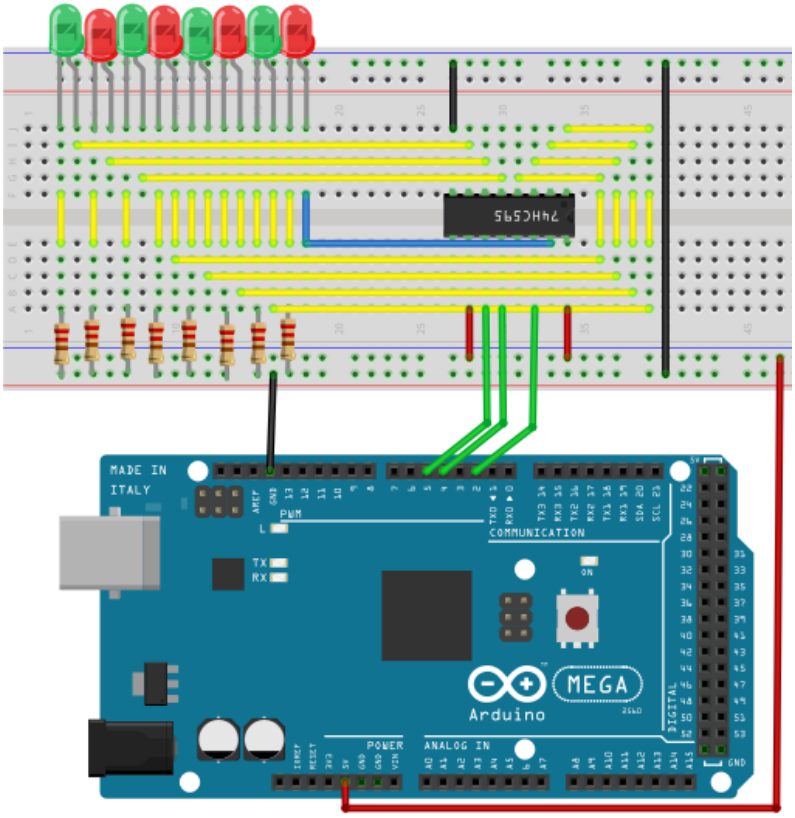
4.17 74HC595



The 74HC595 is a combination of an 8-digit shift register, flag and a tri-state output. In this project the 74HC595 is used to save resources with 8 LEDs. The required I/O ports are reduced from 8 to 3 ports.

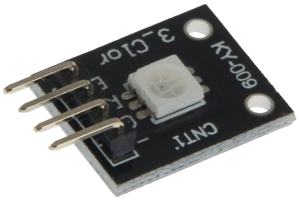
Hardware	Amount
Mega2560 Board	1
USB Cable	1
74HC595 Chip	1
Red M5 LED	4
Green M5 LED	4
220Ω Resistor	8
Breadboard	1
Breadboard Jumper Cable	37



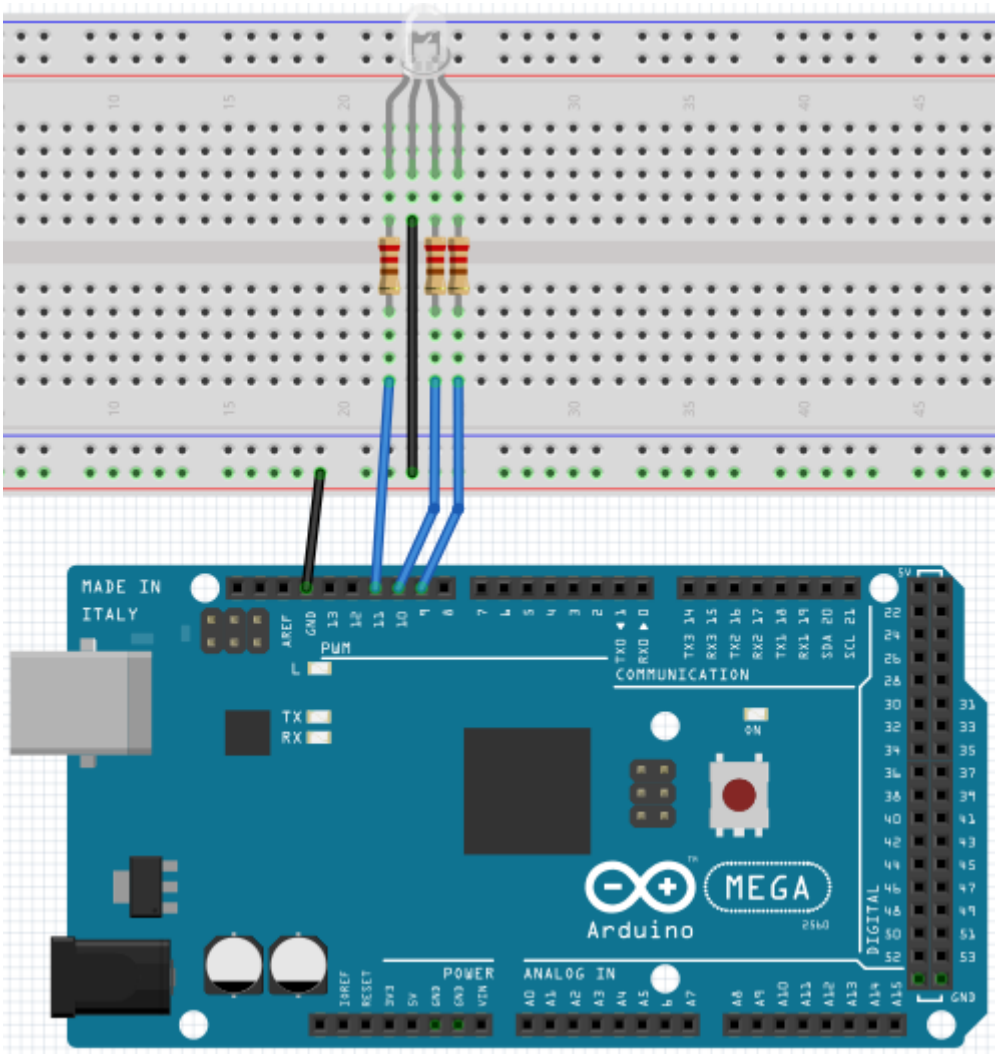



```
int data = 2;          // Sets Pin 14 of 74HC595 to data input
int clock = 5;         // Sets Pin 11 of 74HC595 to clock Pin
int latch = 4;         // Sets Pin 12 of 74HC595 to output
int ledState = 0;
const int ON = HIGH;
const int OFF = LOW;
void setup()
{
      pinMode(data, OUTPUT);
      pinMode(clock, OUTPUT);
      pinMode(latch, OUTPUT);
}
void loop()
{
      for(int i = 0; i < 256; i++)
      {
            updateLEDs(i);
            delay(500);
      }
}
void updateLEDs(int value)
{
      digitalWrite(latch, LOW);
      shiftOut(data, clock, MSBFIRST, ~value);
      digitalWrite(latch, HIGH);                    // Lock
}
```

4.18 RGB-LED



This diode is controlled by PWM signals and has a three-color system to display colors. The component can be implemented directly at the Mega2560 interfaces.

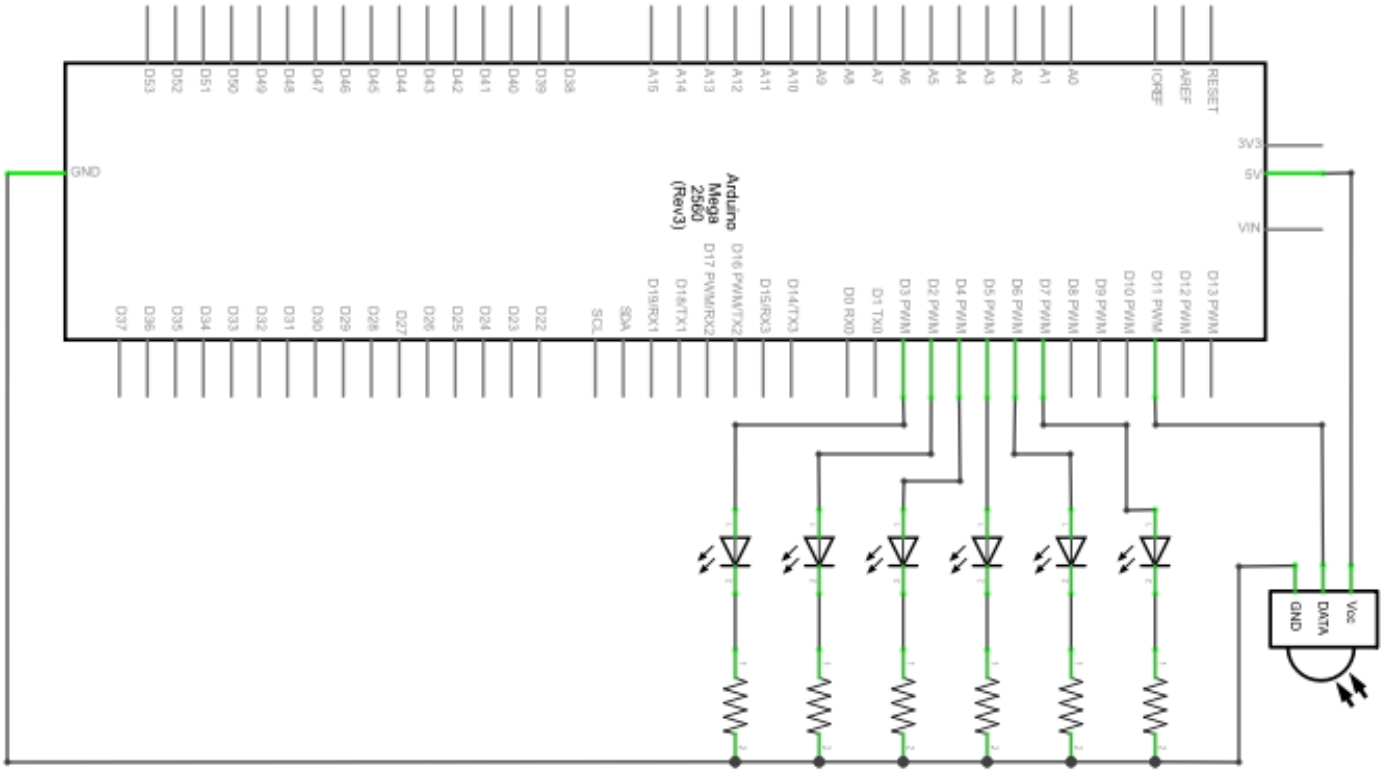


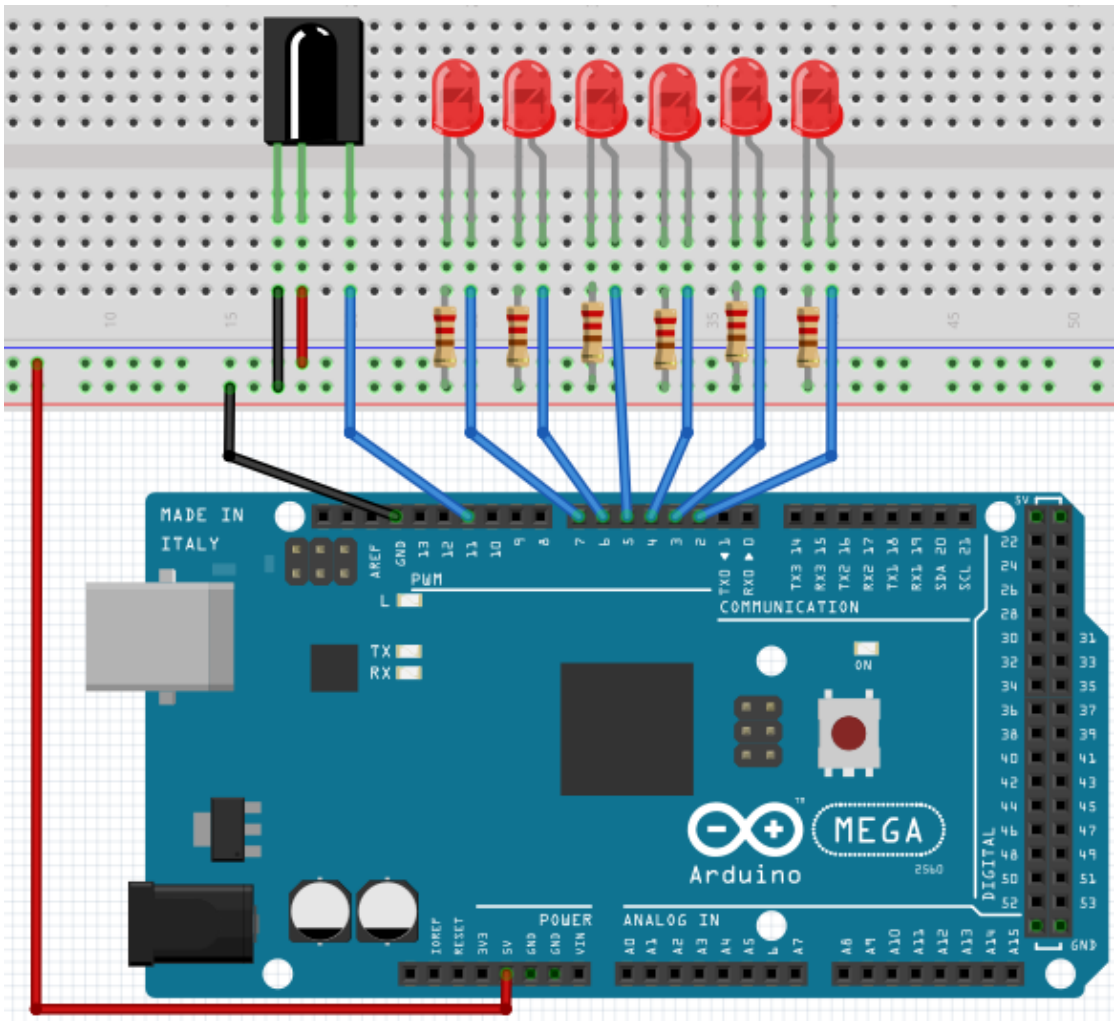
```
int redpin = 11;      // chooses pin for red led
int bluepin =10;      // chooses pin for blue led
int greenpin =9;      // chooses pin for green led aus
int val;

void setup() {
      pinMode(redpin, OUTPUT);
      pinMode(bluepin, OUTPUT);
      pinMode(greenpin, OUTPUT);
      Serial.begin(9600);
}

void loop()
{
      for(val=255; val>0; val--)
      {
            analogWrite(11, val);
            analogWrite(10, 255-val);
            analogWrite(9, 128-val);
            delay(1);
      }
      for(val=0; val<255; val++)
      {
            analogWrite(11, val);
            analogWrite(10, 255-val);
            analogWrite(9, 128-val);
            delay(1);
      }
      Serial.println(val, DEC);
}
```

4.19 Infrared-Remote

The IR receiver converts the incoming light signal into a weak, electrical signal.
To decode the code of a remote control, it is necessary to know the coding method.
The NEC protocol is used in this project.

Hardware	Amount
Mega2560 Board	1
USB Cable	1
Infrared-Receiver	1
Infrared-Remote	1
Red M5 LED	6
220Ω Resistor	6
Breadboard	1
Breadboard Jumper Cable	11



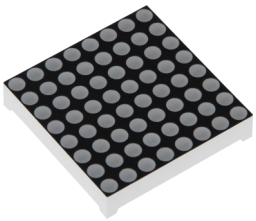


Please make sure that you download and install the IRremote library in your Arduino Library Manager before transferring the code below to your Mega2560.
Only then will the project work as desired.

```
#include <IRremote.h>
int RECV_PIN = 11;
int LED1 = 2;
int LED2 = 3;
int LED3 = 4;
int LED4 = 5;
int LED5 = 6;
int LED6 = 7;
long on1  = 0x00FFA25D;
long off1 = 0x00FFE01F;
long on2 = 0x00FF629D;
long off2 = 0x00FFA857;
long on3 = 0x00FFE21D;
long off3 = 0x00FF906F;
long on4 = 0x00FF22DD;
long off4 = 0x00FF6897;
long on5 = 0x00FF02FD;
long off5 = 0x00FF9867;
long on6 = 0x00FFC23D;
long off6 = 0x00FFB047;
IRrecv irrecv(RECV_PIN);
decode_results results;

void dump(decode_results *results) {
  int count = results->rawlen;
  if (results->decode_type == UNKNOWN)
    {
     Serial.println("Could not decode message");
    }
  else
   {
    if (results->decode_type == NEC)
      {
       Serial.print("Decoded NEC: ");
      }  else if (results->decode_type == SONY)
      {
       Serial.print("Decoded SONY: ");
      }
    else if (results->decode_type == RC5)
      {
       Serial.print("Decoded RC5: ");
      }
    else if (results->decode_type == RC6)
      {
       Serial.print("Decoded RC6: ");
      }
```

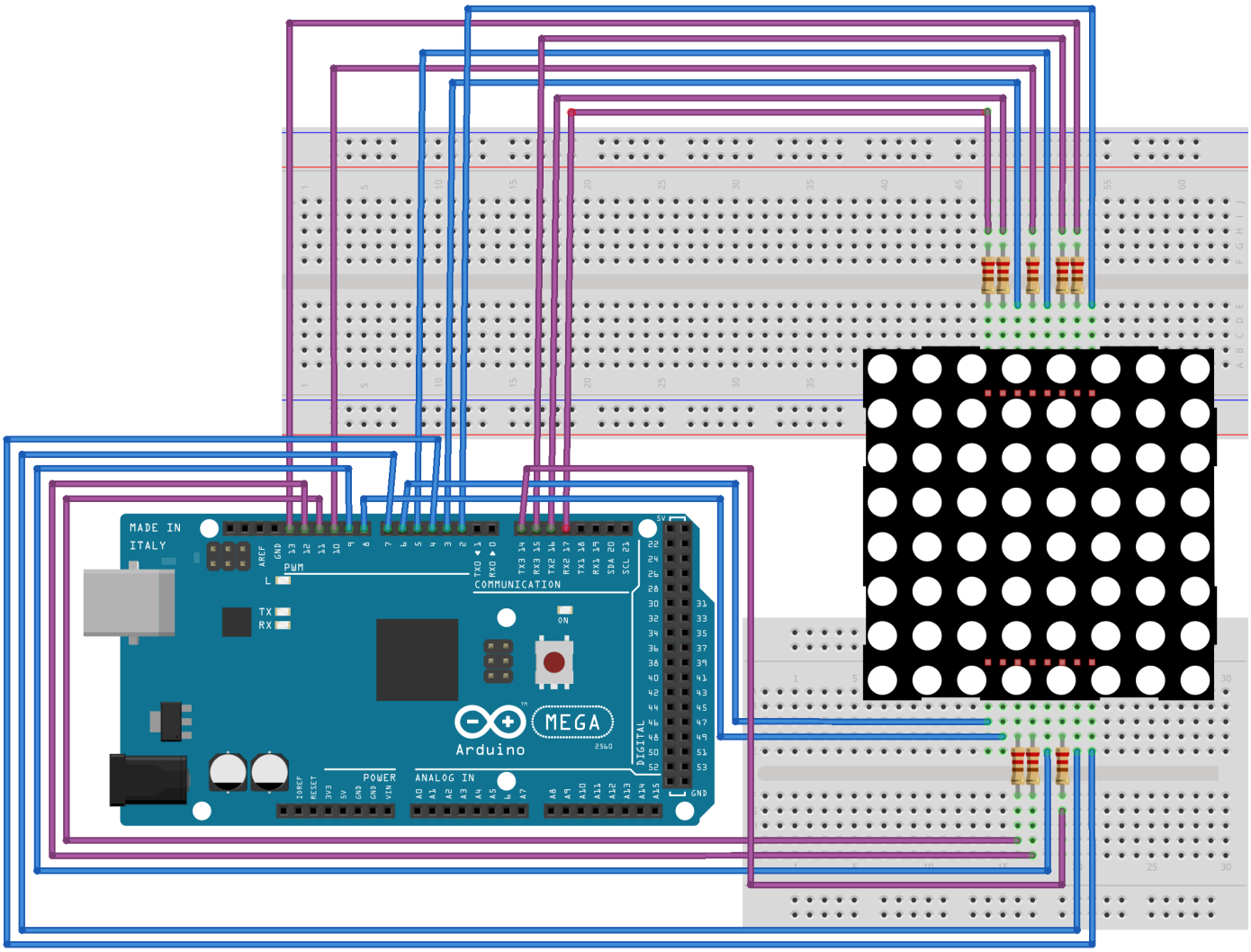
```
Serial.print(results->value, HEX);
     Serial.print(" (");
     Serial.print(results->bits, DEC);
     Serial.println(" bits)");
   }
     Serial.print("Raw (");
     Serial.print(count, DEC);
     Serial.print("): ");

  for (int i = 0; i < count; i++)
     {
          if ((i % 2) == 1) {
          Serial.print(results->rawbuf[i]*USECPERTICK, DEC);
     }
     else
     {
          Serial.print(-(int)results->rawbuf[i]*USECPERTICK, DEC);
     }
    Serial.print(" ");
  }
  Serial.println("");
}

void setup()
{
     pinMode(RECV_PIN, INPUT);
     pinMode(LED1, OUTPUT);
     pinMode(LED2, OUTPUT);
     pinMode(LED3, OUTPUT);
     pinMode(LED4, OUTPUT);
     pinMode(LED5, OUTPUT);
     pinMode(LED6, OUTPUT);
     pinMode(13, OUTPUT);
     Serial.begin(9600);
     irrecv.enableIRIn();    // Start the receiver
}

int on = 0;
unsigned long last = millis();
```

```
void loop()
{
  if (irrecv.decode(&results))
  {
      if (millis() - last > 250)
      {
            on = !on;
            // digitalWrite(8, on ? HIGH : LOW);
            digitalWrite(13, on ? HIGH : LOW);
            dump(&results);
      }
    if (results.value == on1 )
       digitalWrite(LED1, HIGH);
    if (results.value == off1 )
       digitalWrite(LED1, LOW);
    if (results.value == on2 )
       digitalWrite(LED2, HIGH);
    if (results.value == off2 )
       digitalWrite(LED2, LOW);
    if (results.value == on3 )
       digitalWrite(LED3, HIGH);
    if (results.value == off3 )
       digitalWrite(LED3, LOW);
    if (results.value == on4 )
       digitalWrite(LED4, HIGH);
    if (results.value == off4 )
       digitalWrite(LED4, LOW);
    if (results.value == on5 )
       digitalWrite(LED5, HIGH);
    if (results.value == off5 )
       digitalWrite(LED5, LOW);
    if (results.value == on6 )
       digitalWrite(LED6, HIGH);
    if (results.value == off6 )
       digitalWrite(LED6, LOW);
    last = millis();
    irrecv.resume();
  }
}
```



4.20 8x8 LED-Matrix

The internal structure and appearance of an LED matrix are as follows:
The 8x8 LED matrix consists of 64 LEDs. Each LED is placed at the intersection of row and column. If the level for a row is 1 and the level of the corresponding column is 0, the LED will go to its intersection.
Example: If you want to switch on the first LED, set pin 9 to "HIGH LEVEL" and pin 13 to "LOW LEVEL".
If you want to switch on the first row, set pin 9 to "HIGH LEVEL" and tiller 13, 3, 4, 10, 6, 11, 15, 16 to "LOW LEVEL". To activate the first column, set pin 13 to "LOWPEGEL" and tiller 9, 14, 8, 12, 1, 7, 2, 5 to "HIGHPEGEL".



```
// Set an array to store letters from 0
unsigned char Text[]={0x00,0x1c,0x22,0x22,0x22,0x22,0x22,0x1c};
void Draw_point(unsigned char x,unsigned char y)
// Draw-Point Function
{
   clear_();
   digitalWrite(x+2, HIGH);
   digitalWrite(y+10, LOW);
   delay(1);
}
void show_num(void)   // Display function, calls drawing point function
{
  unsigned char i,j,data;
  for(i=0;i<8;i++)
  {
    data=Text[i];
    for(j=0;j<8;j++)
    {
      if(data & 0x01)Draw_point(j,i);
      data>>=1;
    }
  }
}
void setup(){
int i = 0 ;
for(i=2;i<18;i++)
 {
   pinMode(i, OUTPUT);
 }
  clear_();
}
void loop()
{
  show_num();
}
void clear_(void)       // clears screen
{
  for(int i=2;i<10;i++)
  digitalWrite(i, LOW);
  for(int i=0;i<8;i++)
  digitalWrite(i+10, HIGH);
}
```

5. Support

We also support you after the purchase. If there are any questions left or if you encounter any problems feek free to contact us by e-mail, telephone or by our ticket support system on our website.

E-Mail: service@joy-it.net

Ticket-System: http://support.joy-it.net

Phone: +49 (0)2845 98469 – 66 (11- 18 o'clock)

Please visit our website for more informations:

www.joy-it.net

6. EU-Declaration-Of-Conformity

Manufacturer:	JOY-iT Europe GmbH Pascalstr. 8 47506 Neukirchen-Vluyn
Article Name:	ard_mega2560R3 / ARD-Set01
Description:	Microcontroller-Board / Set
Intended Purpose:	Test setups / prototypes

The manufacturer, JOY-IT Europe GmbH, Pascalstr. 8, D-47506 Neukirchen-Vluyn, Germany, hereby declares that the product "ard_Mega2560IP "meets the essential requirements of the following guidelines when used as intended:

2014/ 30/EU (EMV) & 2011/65/EU (Rohs)

The following standards have been applied to assess the device:

EN 61326-1: 2013
Electrical equipment for measurement, control and laboratory equipment - EMC requirements Part 1 General requirements

Date	Name	Signature	Position in the company
03.03.2017	Yue Yang		Executive Manager