Group - 11

Soumava Paul 16EE10056
Yerramsetty Rohit 16EE10055

Experiment 1

Familiarization with Arduino

Aim:

● Familiarization with Arduino Uno board and Arduino IDE
● Blinking an LED
● Using a Push button (4x4 matrix keypad) as a toggle switch to control an LED
● Controlling the brightness of an LED with a PWM
● Making a Macbook like breaking light with PWM

Theory

The board can be powered from your computer's USB port, most USB chargers, or an AC adapter (9 volts recommended, 2.1 mm barrel tip, centre positive). Whenever power is provided at the power socket, Arduino will use that, and if there is no power at the power socket, Arduino will use power from the USB socket. The Arduino UNO Board has following-

14 Digital I/O pins (pins 0–13)

These pins can be either inputs or outputs. Inputs are used to read information from sensors, while outputs are used to control actuators. One has to specify the direction (in or out) in the sketch one creates in the IDE. Digital inputs can only read one of two values, and digital outputs can only output one of two values (HIGH and LOW).

6 Analogue In pins (pins 0–5)

The analogue input pins are used for reading voltage measurements from analogue sensors. In contrast to digital inputs, which can distinguish between only two different levels (HIGH and LOW), analogue inputs can measure 1,024 different levels of voltage.

6 Analogue Out pins (pins 3, 5, 6, 9, 10, and 11)

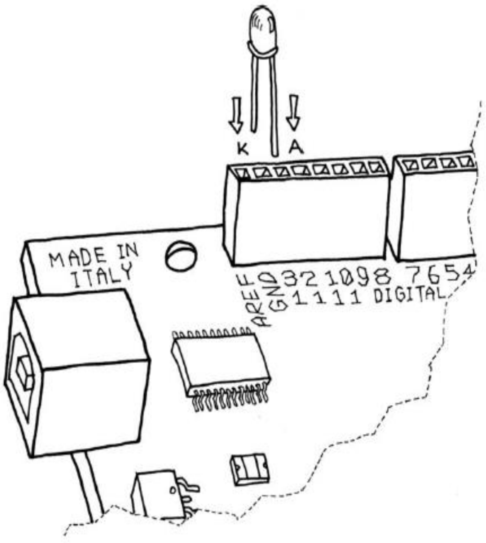
These are actually six of the digital pins that can perform a third function: they can provide analogue output. As with the digital I/O pins, one has to specify what the pin should do in the sketch.

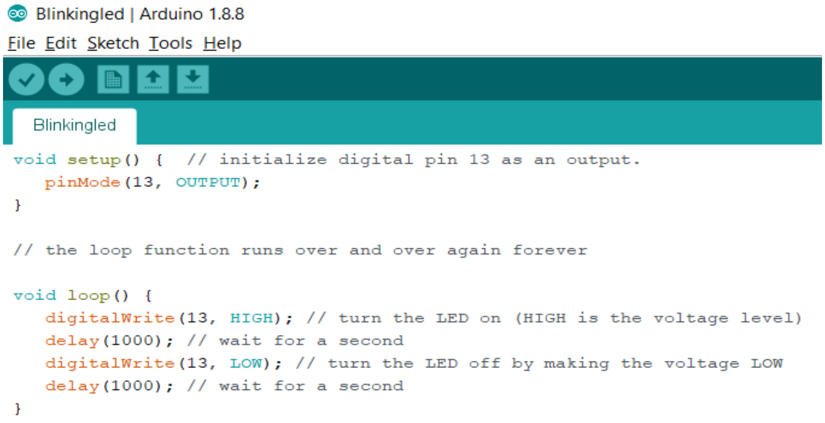
The programming cycle on Arduino is basically as follows:

1. Plug your board into a USB port on your computer.
2. Write a sketch that will bring the board to life.
3. Upload this sketch to the board through the USB connection and wait a couple of seconds for the board to restart.
4. The board executes (performs) the sketch that is written.

Blinking an LED

Circuit Diagram and Code

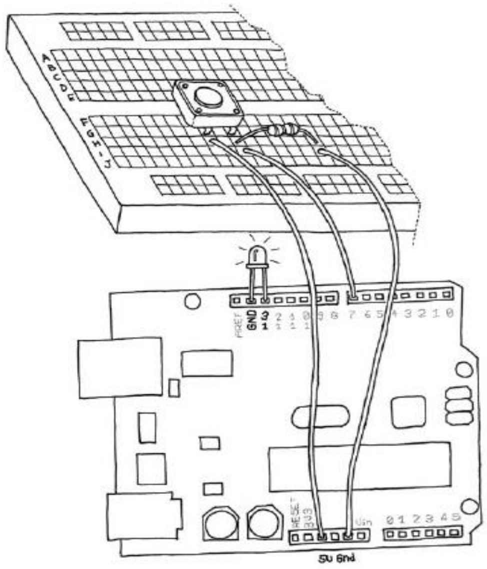


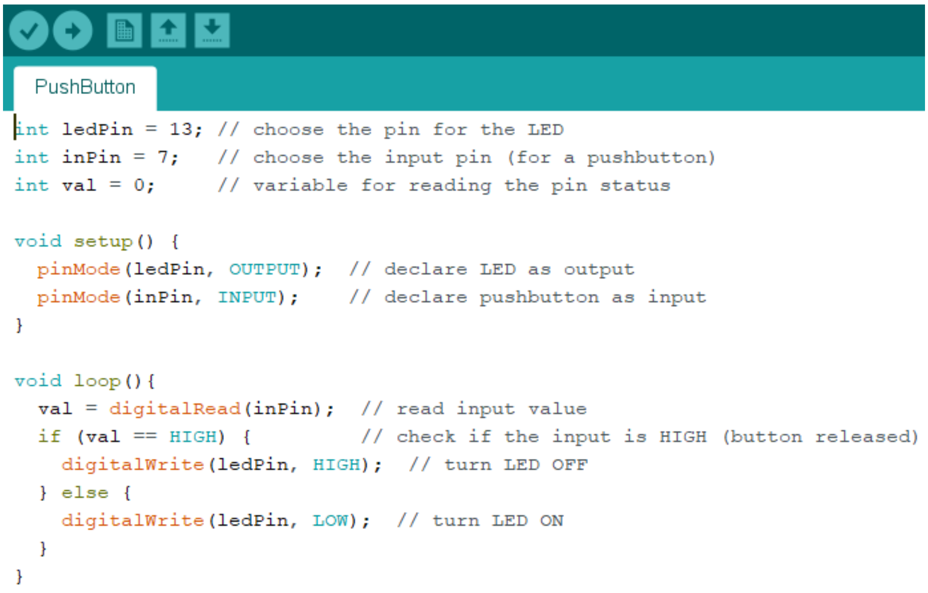


```
void setup() {  // initialize digital pin 13 as an output.
   pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever

void loop() {
   digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
   delay(1000); // wait for a second
   digitalWrite(13, LOW); // turn the LED off by making the voltage LOW
   delay(1000); // wait for a second
}
```

Using a Push button (4x4 matrix keypad) as a toggle switch to control an LED

Circuit Diagram and Code

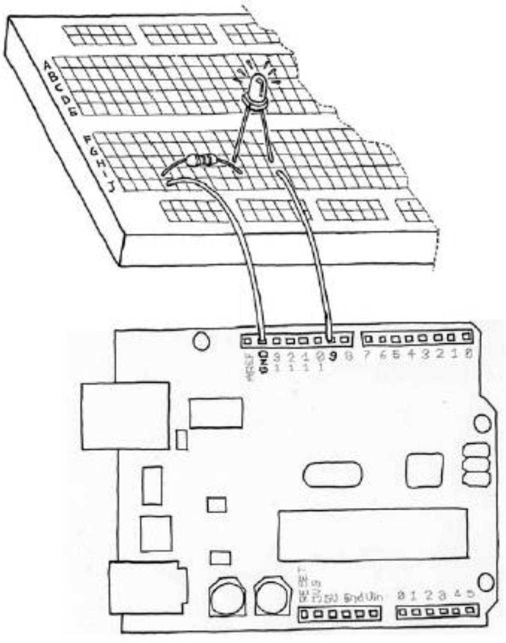


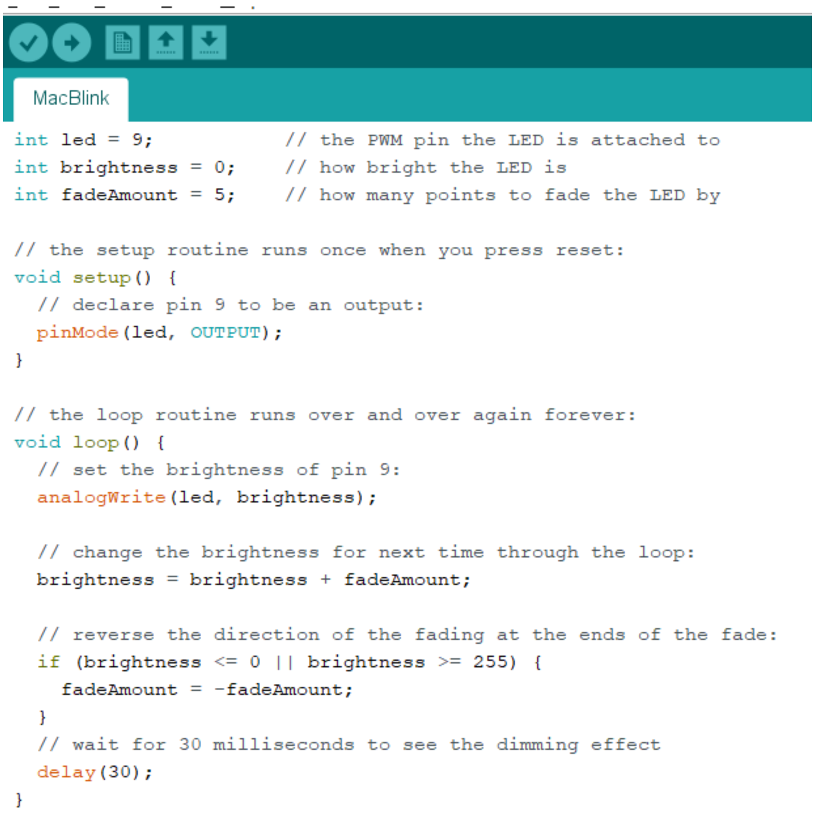


```
int ledPin = 13; // choose the pin for the LED
int inPin = 7;   // choose the input pin (for a pushbutton)
int val = 0;     // variable for reading the pin status

void setup() {
  pinMode(ledPin, OUTPUT);  // declare LED as output
  pinMode(inPin, INPUT);    // declare pushbutton as input
}

void loop(){
  val = digitalRead(inPin);  // read input value
  if (val == HIGH) {         // check if the input is HIGH (button released)
    digitalWrite(ledPin, HIGH);  // turn LED OFF
  } else {
    digitalWrite(ledPin, LOW);  // turn LED ON
  }
}
```

Making a Macbook-like breaking light with PWM

Circuit Diagram and Code





```
int led = 9;           // the PWM pin the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

Discussions

Typical Functions in the Arduino IDE

1) The setup() function is called when a sketch starts. We use it to initialize variables, pin modes, start using libraries, etc. The setup() function will only run once, after each powerup or reset of the Arduino board.
2) The loop() function is where we specify the main behaviour of our interactive device. It will be repeated over and over again until you remove power from the board.
3) The pinMode() function needs two arguments. The first argument tells pinMode() which pin we're talking about, and the second argument tells pinMode() whether we want to use that pin as an input or output. INPUT and OUTPUT are predefined constants.
4) The digitalWrite() function expects two arguments, and just as we saw with the pinMode() function, the first argument tells digitalWrite() what pin we're talking about, and the second argument tells digitalWrite() whether to set the voltage level to 0 (LOW) or to 5 V (HIGH).
5) delay(ms) function pauses the program for the amount of time (in milliseconds) specified as parameter.

Experiments

1) Most Arduinos have an on-board LED we can control. On the UNO, MEGA and ZERO it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to the correct LED pin independent of which board is used.

2) In any experiment involving lighting of an LED, we connect the long leg of the LED (the positive leg, called the anode) to a resistor and the short leg (the negative leg, called the cathode) to the GND. The other end of the resistor is connected to a voltage source. The resistor in series with the LED may be of values ranging from 220 ohm up to 1K ohm. The resistor is used to limit the current flowing through the LED.

3) The pushbutton experiment demonstrates the use of the matrix keypad as a switch. Each time you press the button, the LED is turned on (if it's off) or off (if on). Of the 2 pins used at a time, one is connected to a voltage source and the other one is connected to a resistor. The circuit used is exactly identical to the one given here. Only in place of the pushbutton, we used the matrix keypad.