Practical No. 1

188_Neha_Yadav

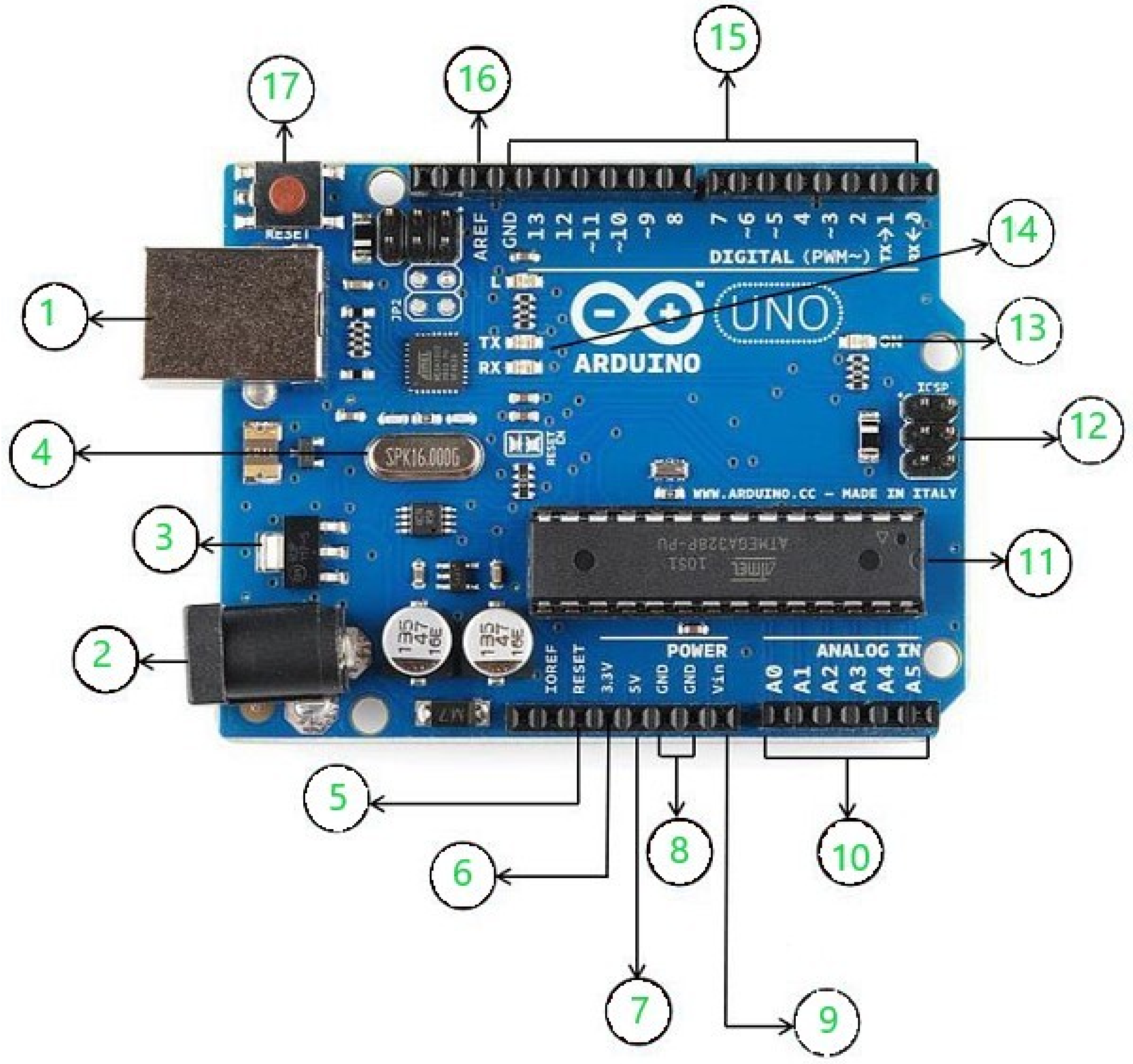
Aim: Introduction to Basic IoT Components.

Objectives:
1. To learn Arduino UNO basics
2. Write a program to blink Arduino onboard LED and to interface external LED with Arduino.

Theory:
Theory about Arduino UNO hardware, functions like setup( ), loop(), pinMode(), delay(), digitalWrite() etc. and breadboard.

Arduino UNO hardware.

Arduino Uno is one of the most basic and popular boards that Arduino offers. This is because it features an ATMega328 microcontroller that is both cheap and powerful enough for most basic beginner-level projects. Once you're familiar with Arduino IDE, you can move up to boards with more powerful and sophisticated chipsets like the MKR range which is concerned with IoT applications and inter compatibility, or the Nano range which as the name suggests is designed to keep the form factor as small as possible while packing most of the features and power of the full-sized boards.

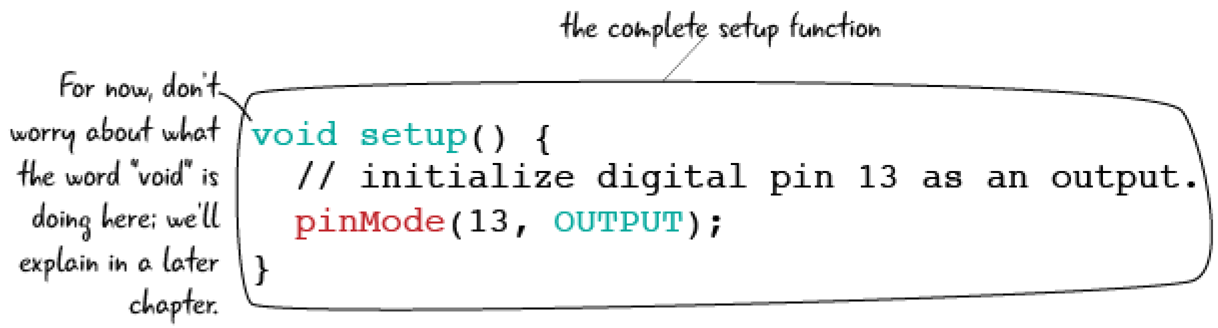


Using the above image as a reference, the labeled components of the board respectively are-

1. **USB:** can be used for both power and communication with the IDE
2. **Barrel Jack:** used for power supply
3. **Voltage Regulator:** regulates and stabilizes the input and output voltages

4. **Crystal Oscillator:** keeps track of time and regulates processor frequency
5. **Reset Pin:** can be used to reset the Arduino Uno
6. **3.3V pin:** can be used as a 3.3V output
7. **5V pin:** can be used as a 5V output
8. **GND pin:** can be used to ground the circuit
9. **Vin pin:** can be used to supply power to the board
10. **Analog pins(A0-A5):** can be used to read analog signals to the board
11. **Microcontroller(ATMega328):** the processing and logical unit of the board
12. **ICSP pin:** a programming header on the board also called SPI
13. **Power indicator LED:** indicates the power status of the board
14. **RX and TX LEDs:** receive(RX) and transmit(TX) LEDs, blink when sending or receiving serial data respectively
15. **Digital I/O pins:** 14 pins capable of reading and outputting digital signals; 6 of these pins are also capable of PWM
16. **AREF pins:** can be used to set an external reference voltage as the upper limit for the analog pins
17. **Reset button:** can be used to reset the board

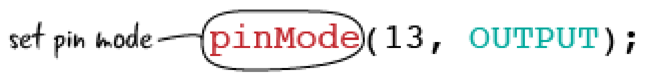
setup(): setting initial conditions


the complete setup function
For now, don't worry about what the word "void" is doing here; we'll explain in a later chapter.

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

Curly braces denote when we begin and end blocks of code (functions).
A line of code is defined by one and only one instruction ended with a semicolon.

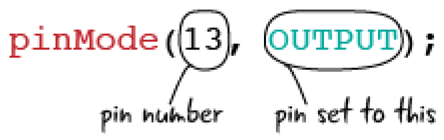
setup() continued: setting the pinMode()



```
pinMode(13, OUTPUT);
```

Semicolons end statements of code, like periods end sentences.

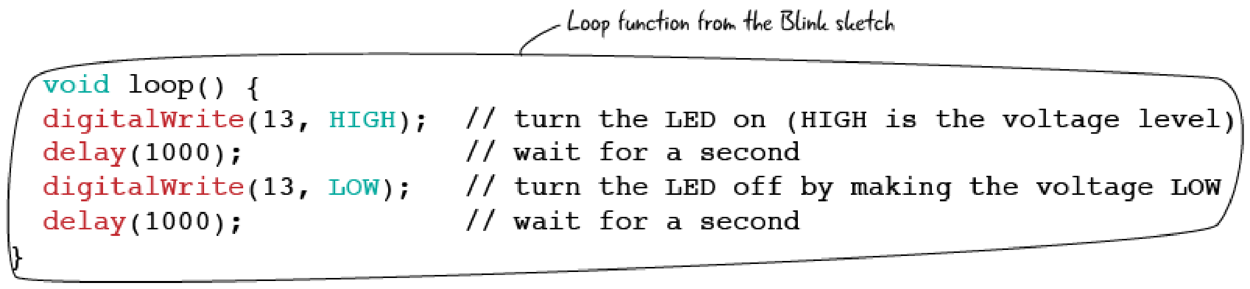
When we want to use a function or instruction like pinMode(), we say that we are "calling" the function.



```
pinMode(13, OUTPUT);
```

Loop()

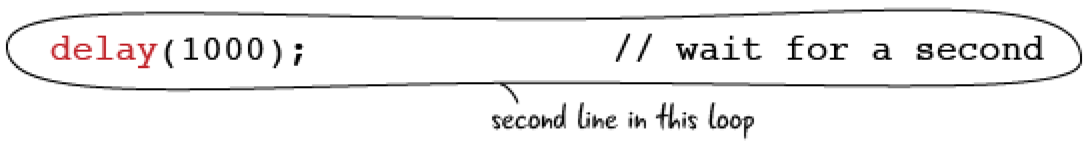
The loop function contains the code that you want to have repeated over and over again. As long as the Arduino is running, this code will keep repeating, after the code in setup has run once.


Loop function from the Blink sketch

```
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
}
```

Loop will continue running as long as the Arduino is on.

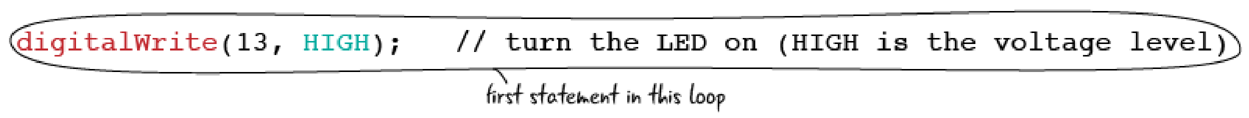
After we turn the pin on, we want to put a short delay in the program. This delay will pause our program, preventing the Arduino from reading the following statement for a small amount of time.
Delay requires that we include between the parentheses the number of milliseconds (one thousand milliseconds equal one second) to wait. In this case, we have stated that the Arduino will wait one thousand milliseconds, or one second, before moving onto the next statement of the program.


second line in this loop

```
delay(1000);                    // wait for a second
```

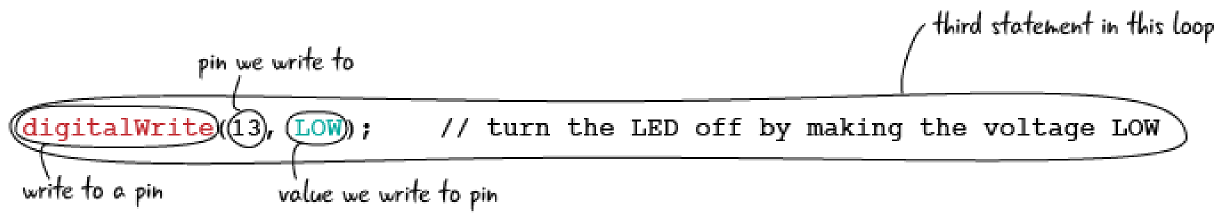
The “digitalWrite” function in this context is used to set whether the pin is ON or OFF. When we write, or set the value of the pin to “HIGH,” we are turning the pin completely on.

digitalWrite(pin #, HIGH) will turn pin # on



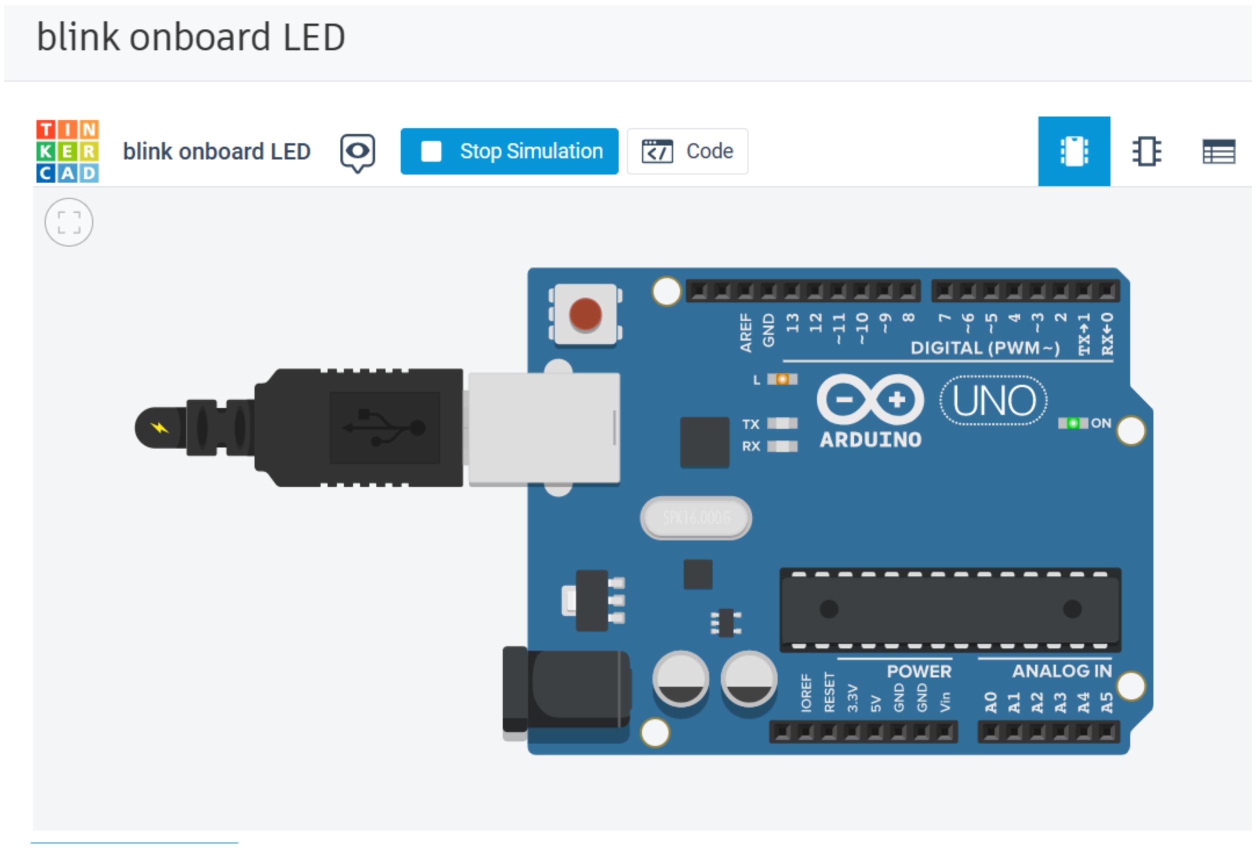
```
digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
```

digitalWrite(pin #, LOW) will turn pin # off

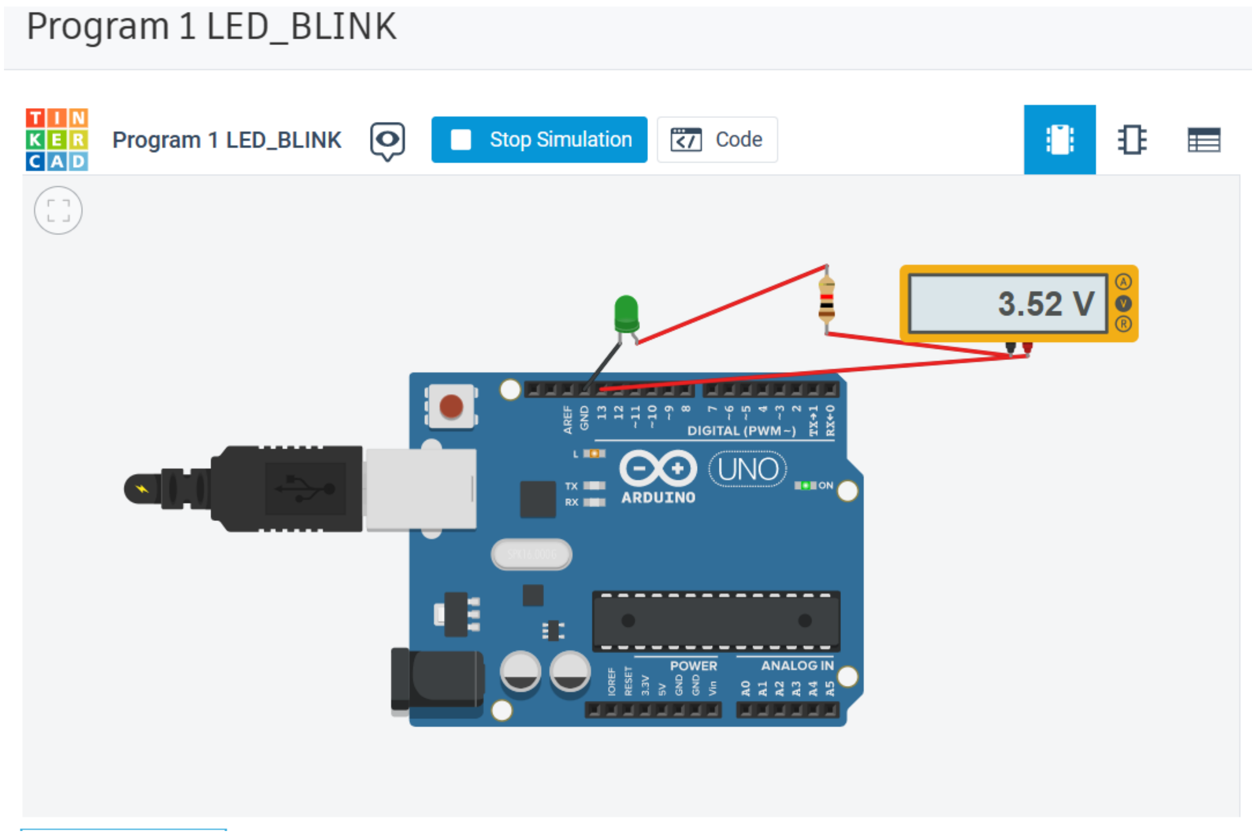


```
digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
```

1.To learn Arduino UNO basics.
Circuit(Output):



2. Write a program to blink Arduino onboard LED and external LED.
Circuit:



Program :

```
void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.begin(9600);

}

void loop()
{
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  // Wait for 1000 millisecond(s)
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
  Serial.print("Hello nehaa!!");
  // Wait for 1000 millisecond(s)
}
```

Conclusion:
Thus, learnt about basic components of IoT like Arduino UNO, Breadboard, resisters, LED’s and interfacing LED with Arduino.