

Arduino in Agriculture

Gatkal N. R¹., Nalawade S. M². and Nimbalkar D. V³.

¹Ph. D. Student, Department of Farm Machinery and Power Engineering, Dr. Annasaheb Shinde College of Agricultural Engineering and Technology, MPKV, Rahuri- 413722, Ahmednagar, Maharashtra, India.

²Professor and Head, Department of Farm Machinery and Power Engineering, Dr. Annasaheb Shinde College of Agricultural Engineering and Technology, MPKV, Rahuri. District- Ahmednagar State- Maharashtra Pin-413722, India.

³Business Manager, ICAR-CIRCOT, Bombay.

ARTICLE ID: 05

Introduction:

An open-source microcontroller called an Arduino may be updated at any time and is simple to program. In 2005, the first Arduino was introduced. The Arduino microcontroller was created to help professionals and students create devices that interact with the environment using sensors. The Arduino microcontroller has inputs and outputs that may be used to get information. The Arduino can then provide output based on the information it has received. Using HTTP requests, Arduino microcontrollers are also capable of sending and receiving data online. The Esp Board is a simple microcontroller with an internet connection. Esp microcontrollers have two options: they may connect to a Wi-Fi server or take on the role of a Wi-Fi server.

Software and hardware make up the Arduino platform. The development board for Arduino is the hardware that it utilizes. The Arduino IDE is the name of the software used by Arduino to create code (Integrated Development Environment). These microcontrollers may be readily programmed using the C or C++ language in the Arduino IDE and are equipped with either an 8-bit Atmel AVR or a 32-bit Atmel ARM microcontroller, both of which are produced by Atmel.

When utilizing a USB cable to upload fresh code to the Arduino board, another application for it is possible. With the help of the Arduino IDE, users may create programs for the Arduino platform using the C or C++ programming languages on nearly any personal computer. The Arduino IDE is multiplatform software that can run on several platforms including Microsoft, Linux, and Mac OS X making the user community even larger

Market vendors provide a wide variety of Arduino microcontroller boards. A little research is needed in order to use the appropriate Arduino board for the project. Each Arduino board offers unique features and characteristics. The use of Arduino microcontrollers as opposed to other microcontrollers is advantageous for a variety of reasons.

Arduino Co-founder Massimo Banzi mentioned some very important reasons to use Arduino boards (Louis, 2016).

1. **Active User Community:** Users of Arduino can submit a message and discuss their ideas. If you run into an issue when using the Arduino board, you may post it to a community site where other users will typically offer suggestions or solutions.
2. **Growth of Arduino:** Arduino microcontrollers are less expensive than their competitors. for newbies to get going right away, which makes it ideal.
3. **Inexpensive Hardware:** The main website of the Arduino platform offers free usage. Users simply have to pay for the Arduino hardware.
4. **Arduino Board as a Programmer.**

Different types of Arduino boards:

There are several different types of Arduino boards in the Arduino board family. Boards like the Arduino BT have a Bluetooth module built in for wireless connection. These built-in modules can also be purchased separately and used in conjunction with it. Shield refers to these modules. a few of the well-liked Arduino shield as shown in Figure 1.

Arduino Ethernet shield: This shield allows an Arduino board to connect to the internet by Ethernet library and to read and write an SD card using the SD library.

- **Arduino Wireless shield:** this shield allows the Arduino board to communicate wirelessly using Zigbee.
- **Arduino Motor Driver Shield:** this shield allows Arduino boards to interface with the driver of a motor etc.

Elements of Arduino boards: Elements of an Arduino Board can be done into two categories:

a) Hardware

The Arduino microcontroller consists of many components. Here are some of those main components and their functionality:

- i. **Microcontroller:** This serves as the development board's central processing unit and may send and receive commands to any attached peripheral devices. Every board contains a different microcontroller, and they all have different specs.
- ii. **External Power Supply:** The Arduino microcontroller is powered by this power source using a DC voltage that ranges from 9 to 12 volts.
- iii. **USB plug:** On the Arduino board, this connector is a key port. Using a USB cable, it is used to upload programs to microcontrollers. In the absence of an external power supply, the Arduino board is powered by a DC voltage of 5 volts from the USB connection.
- iv. Internal Programmer.
- v. Reset button.
- vi. Analog Pins: These pins are used for analogue input and output. Analog pin counts differ from board to board as well.
- vii. Digital I/O Pins: Digital input and output are handled by these pins. Additionally, each board has a different number of these digital pins.
- viii. Power and GND Pins: The development board has pins that can pass 3.3 and 5 volts as well as ground via them.

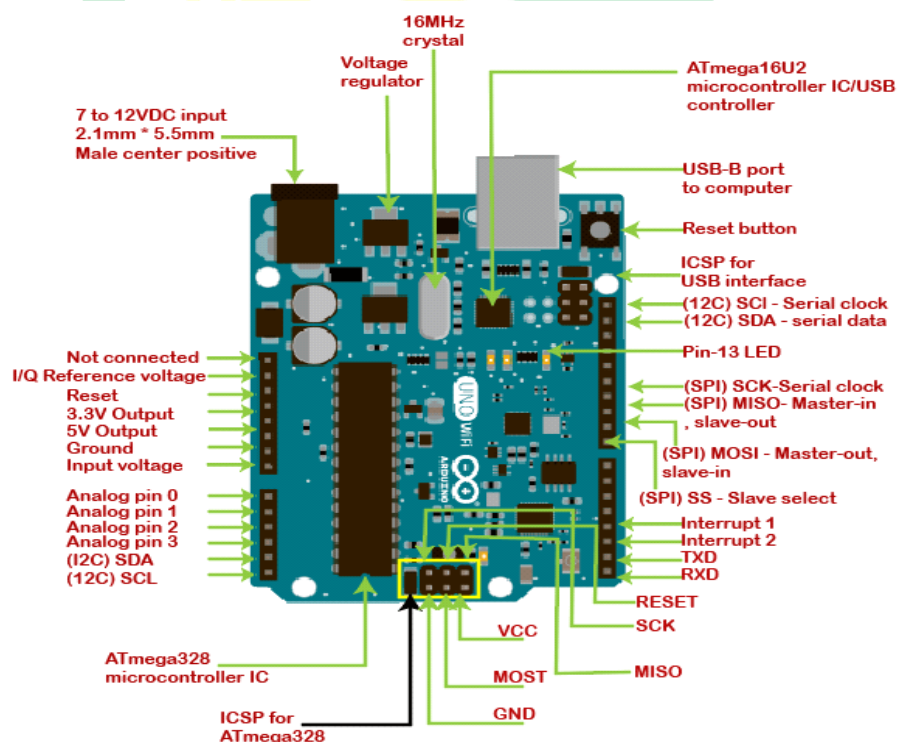


Fig.1: Arduino

1. Arduino Coding Basics:

The code is written in a simple programming language similar to C and C++.

Let's discuss the basics to start with Arduino programming.

i. Brackets

There are two types of brackets used in the Arduino coding, which are listed below.

ii. Parentheses ()

The parentheses brackets are the group of the arguments, such as method, function, or a code statement. These are also used to group the math equations.

iii. Curly Brackets { }

The statements in the code are enclosed in the curly brackets. We always require closed curly brackets to match the open curly bracket in the code or sketch.

Open curly bracket- ' { '

Closed curly bracket - ' } '

iv. Line Comment

There are two types of line comments, which are listed below:

// Single line comment

The text that is written after the two forward slashes are considered as a single line comment. The compiler ignores the code written after the two forward slashes. The comment will not be displayed in the output. Such text is specified for a better understanding of the code or for the explanation of any code statement.

The // (two forward slashes) are also used to ignore some extra lines of code without deleting it.

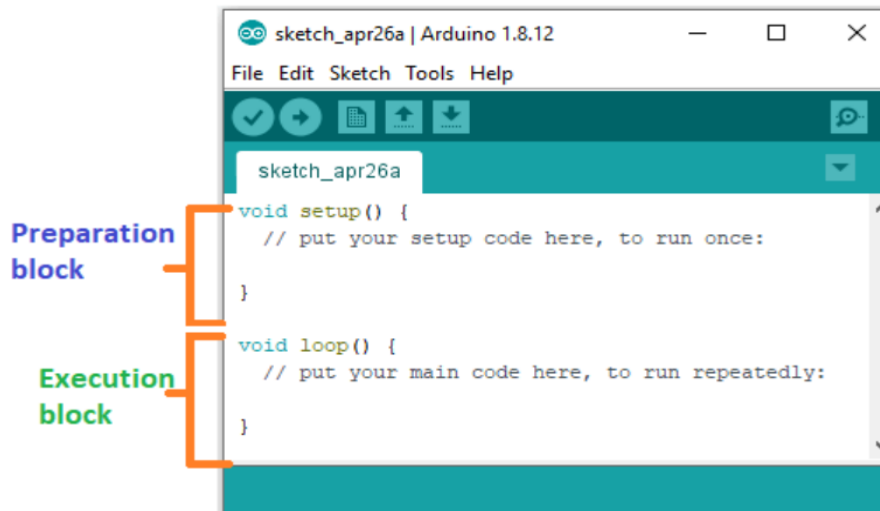
/* Multi - line comment */

The Multi-line comment is written to group the information for clear understanding. It starts with the single forward slash and an asterisk symbol (/ *). It also ends with the / *. It is commonly used to write the larger text. It is a comment, which is also ignored by the compiler.

v. Coding Screen

The coding screen is divided into two blocks. The **setup** is considered as the preparation block, while the **loop** is considered as the execution block. It is shown below as shown in Figure 2:

For example:



```

sketch_apr26a
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}

```

Fig 2: Coding screen.

The set of statements in the setup and loop blocks are enclosed with the curly brackets. We can write multiple statements depending on the coding requirements for a particular project.

1. **void** setup ()
2. {
3. Coding statement 1;
4. Coding statement 2;
5. .
6. .
7. .
8. Coding statement n;
9. }
10. **void** loop ()
11. {
12. Coding statement 1;
13. Coding statement 2;
14. .
15. .
16. .



17. Coding statement n;

18. }

vi. **PinMode()**

The specific pin number is set as the INPUT or OUTPUT in the pinMode () function.

The Syntax is: **pinMode (pin, mode)**

Where,

pin: It is the pin number. We can select the pin number according to the requirements.

Mode: We can set the mode as INPUT or OUTPUT according to the corresponding pin number.

Let' understand the pinMode with an example.

vii. **digitalWrite()**

The digitalWrite() function is used to set the value of a pin as HIGH or LOW.

Where,

HIGH: It sets the value of the voltage. For the 5V board, it will set the value of 5V, while for 3.3V, it will set the value of 3.3V.

LOW: It sets the value = 0 (GND).

If we do not set the pinMode as OUTPUT, the LED may light dim.

The syntax is: **digitalWrite(pin, value HIGH/LOW)**

Arduino Syntax and Program Flow

2. Syntax

Syntax in Arduino signifies the rules need to be followed for the successful uploading of the Arduino program to the board. The syntax of Arduino is similar to the grammar in English. It means that the rules must be followed in order to compile and run our code successfully. If we break those rules, our computer program may compile and run, but with some bugs.

Functions:

The functions in Arduino combine many pieces of lines of code into one.

- a. The functions usually return a value after finishing execution. But here, the function does not return any value due to the presence of void.
- b. The setup and loop function have void keyword present in front of their function name.



- c. The multiple lines of code that a function encapsulates are written inside curly bracket.
- d. Every closing curly bracket ' } ' must match the opening curly bracket ' { ' in the code.
- e. We can also write our own functions, which will be discussed later in this tutorial.

2.1 Spaces

- a. Arduino ignores the white spaces and tabs before the coding statements.
- b. The coding statements in the code are intent (empty spacing at the starting) for the easy reading.
- c. In the function definition, loop, and conditional statements, 1 intent = 2 spaces.
- d. The compiler of Arduino also ignores the spaces in the parentheses, commas, blank lines, etc.

2.2 Tools Tab

- a. The verify icon present on the tool tab only compiles the code. It is a quick method to check that whether the syntax of our program is correct or not.
- b. To compile, run, and upload the code to the board, we need to click on the Upload button.

4.4. Uses of Parentheses ()

- a. It denotes the function like void setup () and void loop ().
- b. The parameter's inputs to the function are enclosed within the parentheses.
- c. It is also used to change the order of operations in mathematical operations.

4.5. Semicolon ;

- a. It is the statement terminator in the C as well as C++.
- b. A statement is a command given to the Arduino, which instructs it to take some kind of action. Hence, the terminator is essential to signify the end of a statement.
- c. We can write one or more statements in a single line, but with semicolon indicating the end of each statement.
- d. The compiler will indicate an error if a semicolon is absent in any of the statements.
- e. It is recommended to write each statement with semicolon in a different line, which makes the code easier to read.
- f. We are not required to place a semicolon after the curly braces of the setup and loop function.

Arduino processes each statement sequentially. It executes one statement at a time before moving to the next statement.

4.6. Program Flow

The program flow in Arduino is similar to the flowcharts. It represents the execution of a program in order.

We recommend to draw the flowchart before writing the code. It helps us to understand the concept of code, which makes it the coding simpler and easier.

4.7. Flow Charts

A flowchart uses shapes and arrows to represent the information or sequence of actions.

An oval ellipse shows the Start of the sequence, and a square shows the action or processes that need to be performed.

The Arduino coding process in the form of the flowchart is shown below:

Setup () : Acts as an entry point

Here, the processor enters our code, and the execution of code begins. After the setup, the execution of the statement in the loop begins.

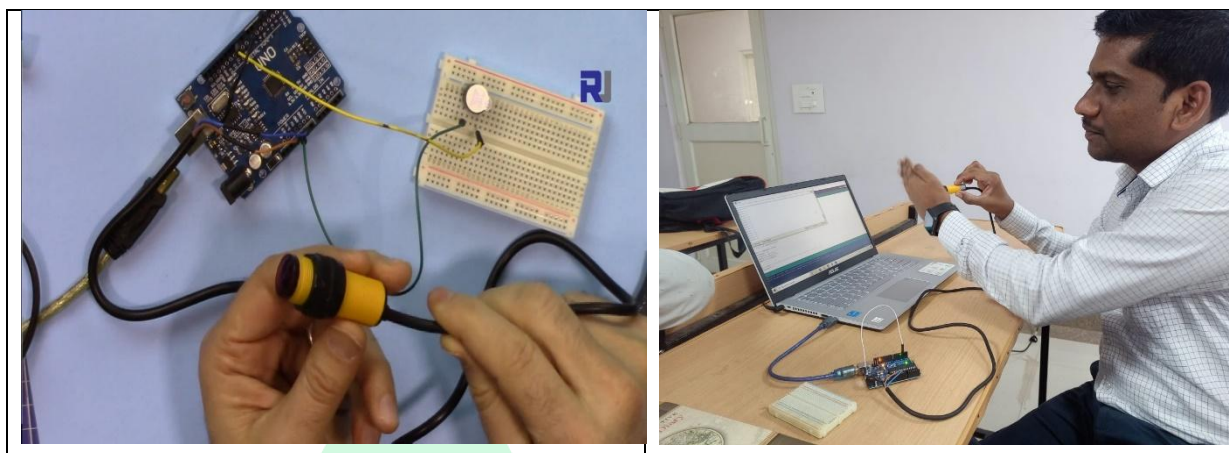
loop () : runs over and over again

Example. Calculate RPM of wheel using Ultrasonic sensor through Arduino.

```

uint32_t startTime;
volatile int rpmcountr1 = 0;
int rpmr1 = 0;
const byte interruptPinr1 = 2;
unsigned long lastmillis = 0;
#include <Wire.h>
void setup() {
  Serial.begin(9600);
  Serial.println("speed = ");
  pinMode(interruptPinr1, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(interruptPinr1),
    rpm, FALLING);
}
void loop() {
  if(millis()-lastmillis==1000)
  {
    rpmr1 = rpmcountr1 * 60/4;
  }
}

```


Result:**Fig. 4: Arduino, Ultrasonic sensor.****3. Arduino applications:**

- i. Smart homes:** With Arduino boards, we may regulate household activities using control systems like motion sensors, outlet controls, temperature controls, blower controls, garage door controls, air flow controls, sprinkler controls, and bill of materials controls (David et al., 2015).
- ii. Defence:** An object's range, altitude, direction, or speed can be determined using RADAR (Radio Detection and Ranging), a technology that uses radio waves to detect things. Different sizes and performance requirements are possible for radars. It may be used for long-range surveillance, early-warning systems in ships, and air traffic control at airports. This system is the brain of a missile guidance system. In times of conflict, both large-room systems and a variety of compact, portable radars are maintained and controlled (Bhor et al., 2016).
- iii. Industries:** Because of its simple programming environment, variety of signal types, and ease of adaptability to new setups, Arduino is used in a wide range of industries. For adding remote control and monitoring features to tiny legacy industrial systems, Arduino boards provide a versatile, low-cost alternative to the typical industrial gadgets. The proliferation of WiFi and other wireless technologies over the past few years has made wireless systems boring in our daily lives (Teja et al., 2017).
- iv. Traffic Signal Control:** Today, Arduino is used to control traffic lights. It may also be used to operate real-time systems with configurable timings and illumination for pedestrians, among other things. In a traffic control system, junction timing is

automatically adjusted to facilitate smooth vehicle movement and prevent waiting at junctions.

- v. **Medical:** The number of heartbeats in a minute is counted using an Arduino-based heartbeat monitor. This has an integrated heartbeat sensor module that detects the heartbeat when a finger is placed on the sensor. The open-source EEG, ECG, and EMG, the thermometer, the WiFi body scale with Arduino Board, and many more pieces of medical equipment are designed using the Arduino platform (Rákay et al., 2015).
- vi. **Laboratories:** Arduino offers a useful platform in the lab for developing and studying circuit design. Beginning users run the risk of breaking something or doing something incorrectly, and using new electronic components may be expensive for them. The Arduino Simulator provides a solution to these issues with no cable clutter, no hardware expense, faster circuit development, and no harm to your components. The automatic slide movement microscope built on Arduino is a relatively affordable lab tool (Rubio et al., 2015).
- vii. **Body Control with Arduino:** Arduino can operate a variety of bodily parts, including handSight gloves, breathalyzer microphones, heart rate monitors, and other medical devices. A heart rate monitor powered by Arduino is more sophisticated than one that only measures the user's heart rate. Our heart rate sensors are speaking! Each button verbally explains what it does while also displaying the data on the screen. This monitor will average the most recent four readings, show them, and provide some motivational sayings (Mallick and Patro, 2016).
- viii. **Aerospace:** Classical control theory to an airplane flap model and integration of RC vehicles in a Robotic Arena (Alvarez et al., 2015).
- ix. **Automatic Vehicle Control:** It is used to control various system in machinery like wheel speed, rpm, distance, torque, load etc (Alvarez et al., 2015).

Conclusion:

Successful development and control of five smart home apps using an Arduino platform and an Android smartphone In a mock smart house, the apps regulate the temperature and humidity display, the fan speed, the lighting and electrical outlets, the fire alarm system, and the poisonous gas alert system. Following the purchase of the necessary parts for these systems, the wiring circuits and controllers were put into place. In a later

study, Arduino components may be wirelessly linked to Android cell phones. Different controllers, including PLCs, might also be employed and studied.

References:

- Abdulkareem, M. M., Mohammed, Q. A. and Shakir, M. M. 2016. A Short Range Radar System : Rangefinder, Technical Report TR-EEE351, Electrical and Electronics Department, University of Turkish Aeronautical Association Electric and Electronic Engineering Department.
- Alvarez, E.C., Holguino, J. and Restrepo, V. 2015. Applying Classical Control Theory to an Airplane Flap Model on Real Physical Hardware. Proceedings of the 13th Latin American and Caribbean Conference for Engineering and Technology.
- ARDUINO.CC, "Arduino – Introduction", 2015 [Online] Available: <http://arduino.cc/en/Guide/Introduction>. [Accessed: 5- Dec - 2022].
- Bhor, G., Bhandari, P., Ghodekar, R. and Deshmukh, S. 2016. MINI RADAR. International Journal of Technical Research and Applications. 39: 68-71.
- David, N., Chima, A., Ugochukwu, A. and Obinna, E. 2015. Design of a Home Automation System Using Arduino. International Journal of Scientific & Engineering Research. 6(6): 795-801.
- Louis, L. 2016. Working principle of Arduino and using it as a tool for study and research. International Journal of Control, Automation, Communication and Systems (IJCACS). 1(2): 21-29. DOI: 10.5121/ijcacs.2016.1203.
- Mallick, B. and Patro, A. K. 2016. Heart Rate Monitoring System Using Finger Tip Through Arduino And Processing Software. International Journal of Science, Engineering and Technology Research (IJSETR). 5(1): 82-89.
- Rákay, R., Višňovský, M., Galajdová, A. and Šimšík, D. 2015. Testing Properties of E-health System Based on Arduino. Journal of Automation and Control. 3(3): 122-126.
- Rubio, H., Soriano, E. and Barber, R. 2015. A Low-Cost Lab Monitoring System Based on Arduino Microcontroller and Android, Proceedings of ICERI2015 Conference. 8014-8022.
- Swetha, B.R., Yuvasri, D., Karthiga, M. and Padma, S. 2017. Density Based Traffic Control System Using Arduino Uno, SSRG International Journal of Industrial Engineering. 5-7.



Teja,P.S., Krishna, V. M. and Raja,D.C. 2017. Industry Supervising System By Using Arduino &IOT,|| International Journal of Advanced Research in Electronics and Communication Engineering , vol. 6(3): 93-95.

