UNIT III
IOT AND ARDUINO PROGRAMMING

Introduction to the Concept of IoT Devices – IoT Devices Versus Computers – IoT Configurations – Basic Components – Introduction to Arduino – Types of Arduino – Arduino Toolchain – Arduino Programming Structure – Sketches – Pins – Input/Output From Pins Using Sketches – Introduction to Arduino Shields – Integration of Sensors and Actuators with Arduino.

What Is Iot:

IoT stands for Internet of Things. It refers to the interconnectedness of physical devices, such as appliances and vehicles, that are embedded with software, sensors, and connectivity which enables these objects to connect and exchange data. This technology allows for the collection and sharing of data from a vast network of devices, creating opportunities for more efficient and automated systems.

Internet of Things (IoT) is the networking of physical objects that contain electronics embedded within their architecture in order to communicate and sense interactions amongst each other or with respect to the external environment. In the upcoming years, IoT-based technology will offer advanced levels of services and practically change the way people lead their daily lives. Advancements in medicine, power, gene therapies, agriculture, smart cities, and smart homes are just a very few of the categorical examples where IoT is strongly established.

IoT is network of interconnected computing devices which are embedded in everyday objects, enabling them to send and receive data. Over 9 billion 'Things' (physical objects) are currently connected to the Internet, as of now. In the near future, this number is expected to rise to a whopping 20 billion.

There are two ways of building IoT:

1. Form a separate internetwork including only physical objects.
2. Make the Internet ever more expansive, but this requires hard-core technologies such as rigorous cloud computing and rapid big data storage (expensive).

In the near future, IoT will become broader and more complex in terms of scope. It will change the world in terms of "anytime, anyplace, anything in connectivity."

IoT Enablers:

- **RFIDs:** uses radio waves in order to electronically track the tags attached to each physical object.
- **Sensors:** devices that are able to detect changes in an environment (ex: motion detectors).

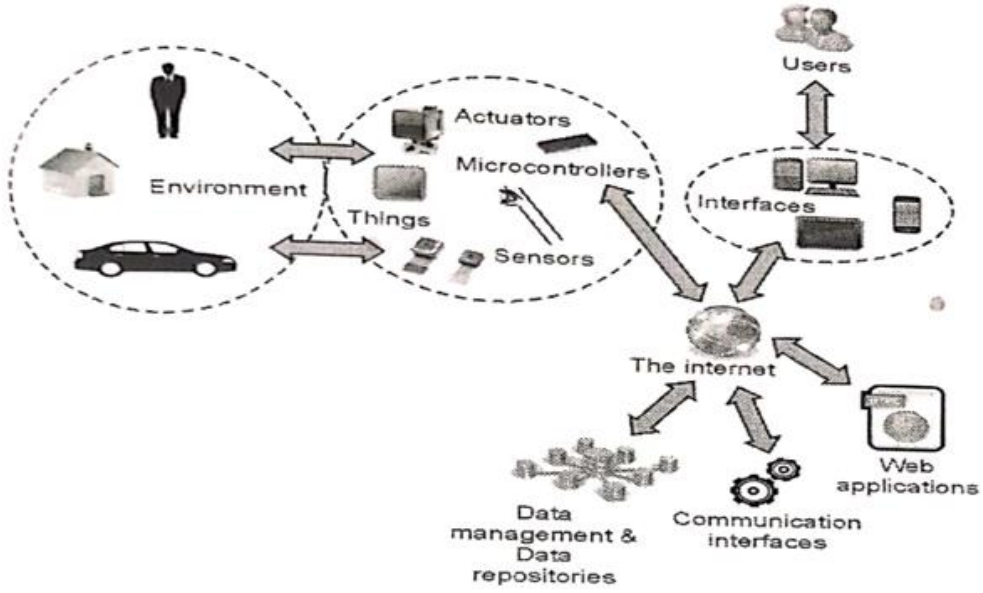
Different types of Sensors:

1. Temperature Sensors
2. Image Sensors
3. Gyro Sensors
4. Obstacle Sensors
5. RF Sensor
6. IR Sensor
7. MQ-02/05 Gas Sensor
8. LDR Sensor
9. Ultrasonic Distance Sensor

- **Nanotechnology:** as the name suggests, these are extremely small devices with dimensions usually less than a hundred nanometers.
- **Smart networks:** (ex: mesh topology).

Working with IoT Devices :

- Collect and Transmit Data : For this purpose sensors are widely used they are used as per requirements in different application areas.
- Actuate device based on triggers produced by sensors or processing devices : If certain condition is satisfied or according to user's requirements if certain trigger is activated then which action to performed that is shown by Actuator devices.
- Receive Information : From network devices user or device can take certain information also for their analysis and processing purposes.
- Communication Assistance : Communication assistance is the phenomena of communication between 2 network or communication between 2 or more IoT devices of same or different Networks. This can be achieved by different communication protocols like : MQTT , Constrained Application Protocol, ZigBee, FTP, HTTP etc.



Working of IoT

Characteristics of the Internet of Things

The Internet of Things (IoT) is characterized by the following key features that are mentioned below.

1. **Connectivity**

Connectivity is an important requirement of the IoT infrastructure. Things of IoT should be connected to the IoT infrastructure. Anyone, anywhere, anytime can connect, this should be guaranteed at all times. For example, the connection between people through Internet devices like mobile phones, and other gadgets, also a connection between Internet devices such as routers, gateways, sensors, etc.

2. Intelligence and Identity

The extraction of knowledge from the generated data is very important. For example, a sensor generates data, but that data will only be useful if it is interpreted properly. Each IoT device has a unique identity. This identification is helpful in tracking the equipment and at times for querying its status.

3. Scalability

The number of elements connected to the IoT zone is increasing day by day. Hence, an IoT setup should be capable of handling the massive expansion. The data generated as an outcome is enormous, and it should be handled appropriately.

4. Dynamic and Self-Adapting (Complexity)

IoT devices should dynamically adapt themselves to changing contexts and scenarios. Assume a camera meant for surveillance. It should be adaptable to work in different conditions and different light situations (morning, afternoon, and night).

5. Architecture

IoT Architecture cannot be homogeneous in nature. It should be hybrid, supporting different manufacturers ‘ products to function in the IoT network. IoT is not owned by anyone engineering branch. IoT is a reality when multiple domains come together.

6. Safety

There is a danger of the sensitive personal details of the users getting compromised when all his/her devices are connected to the internet. This can cause a loss to the user. Hence, data security is the major challenge. Besides, the equipment involved is huge. IoT networks may also be at risk. Therefore, equipment safety is also critical.

7. Self Configuring

This is one of the most important characteristics of IoT. IoT devices are able to upgrade their software in accordance with requirements with a minimum of user participation. Additionally, they can set up the network, allowing for the addition of new devices to an already-existing network.

8. Interoperability

IoT devices use standardized protocols and technologies to ensure they can communicate with each other and other systems. Interoperability is one of the key characteristics of the Internet of Things (IoT). It refers to the ability of different IoT devices and systems to communicate and exchange data with each other, regardless of the underlying technology or manufacturer.

9. Embedded Sensors and Actuators

Embedded sensors and actuators are critical components of the Internet of Things (IoT). They allow IoT devices to interact with their environment and collect and transmit data.

10. Autonomous operation

Autonomous operation refers to the ability of IoT devices and systems to operate independently and make decisions without human intervention. This is a crucial characteristic of the Internet of Things (IoT) and enables a wide range of new applications and services.

Configuring an Internet of Things (IoT)

Identify your IoT components: Determine the devices and sensors you will be using in your IoT network. These could include sensors, actuators, gateways, and edge devices.

Choose a Communication Protocol: Decide on the communication protocol you'll use for your devices to exchange data. Common protocols include MQTT, CoAP, HTTP, and AMQP. Choose a protocol based on factors like the type of data you'll be transmitting, the reliability needed, and the network environment.

Set Up Network Infrastructure: Ensure you have a stable and secure network infrastructure in place. This might include Wi-Fi, Ethernet, cellular, or even LoRaWAN for long-range low-power communication.

Device Registration and Onboarding: Register each IoT device to your network. This might involve provisioning them with necessary credentials (such as certificates) to ensure secure communication.

Security Measures: Implement security mechanisms to protect your IoT ecosystem from unauthorized access, data breaches, and cyberattacks. This could involve using encryption, secure bootstrapping, device authentication, and firewalls.

Data Collection and Transmission: Configure devices to collect the required data from sensors and other sources. Set up rules for data transmission frequency, conditions, and thresholds. Ensure that data is sent efficiently to minimize network usage.

Data Processing at the Edge: If you're using edge computing, configure devices or gateways to process data locally before sending it to the cloud. This reduces latency and conserves bandwidth.

Cloud Integration: Integrate your IoT system with cloud platforms like AWS IoT, Microsoft Azure IoT, or Google Cloud IoT. Configure endpoints, authentication, and data routing to the cloud services.

Data Storage and Analytics: Set up data storage and analytics pipelines to process and analyze the incoming data. This might involve using databases, data lakes, and analytics tools to gain insights from the collected data.

Remote Device Management: Implement mechanisms for remote device management, such as firmware updates, configuration changes, and diagnostics. Over-the-air (OTA) updates can be critical for maintaining device security and functionality.

Scalability Planning: Consider how your IoT system will scale as you add more devices and sensors. Design the architecture in a way that allows easy scalability without major disruptions.

Monitoring and Maintenance: Set up monitoring tools to keep track of the health, performance, and security of your IoT devices and network. Implement automated alerts for abnormal conditions.

Compliance and Regulations: Ensure that your IoT system complies with relevant regulations and standards, especially if it involves sensitive or personal data.

Testing and Iteration: Thoroughly test your IoT configuration in a controlled environment before deploying it at scale. Make necessary adjustments based on testing results and user feedback.

Documentation: Maintain comprehensive documentation of your IoT configuration, including device specifications, network settings, security measures, and communication protocols. This documentation will be valuable for troubleshooting and future enhancements.

Types of IoT Applications:

IoT devices span a wide range of industries and applications, including:

- **Home Automation:** Smart thermostats, doorbell cameras, smart lights, and connected appliances.
- **Healthcare:** Wearable fitness trackers, remote patient monitoring devices, and medical implants.
- **Industrial IoT (IIoT):** Sensors and monitors used in manufacturing, supply chain management, and predictive maintenance.

- **Agriculture:** Soil moisture sensors, GPS trackers for livestock, and automated irrigation systems.
- **Smart Cities:** Connected streetlights, waste management systems, and traffic monitoring.
- **Automotive:** Connected vehicles, vehicle-to-vehicle (V2V) communication, and self-driving cars.

Benefits of IoT Devices:

- **Efficiency:** IoT devices can optimize processes, reduce waste, and enhance resource utilization.
- **Automation:** Tasks can be automated based on real-time data, leading to improved productivity.
- **Data-Driven Insights:** IoT devices provide data that can be analyzed to gain insights, make informed decisions, and identify trends.
- **Remote Monitoring and Control:** Devices can be monitored and controlled remotely, enabling real-time adjustments.
- **Enhanced User Experience:** IoT devices create personalized experiences and convenience for users.

Challenges and Considerations:

- **Security and Privacy:** IoT devices can be vulnerable to cyberattacks if not properly secured.
- **Interoperability:** Ensuring that devices from different manufacturers can communicate seamlessly.
- **Scalability:** Handling a large number of devices and data points can be challenging.
- **Power Management:** Ensuring devices have adequate power and optimizing energy consumption.
- **Data Management:** Efficiently handling and processing the vast amount of data generated by IoT devices.

Basic components of IoT:

1. **Devices/Things:** These are physical objects embedded with sensors, actuators, and communication modules that enable them to collect data, perform actions, and communicate over the internet. Devices can range from simple sensors to complex machinery.

2. **Sensors:** Sensors are components that detect changes in the environment and convert physical phenomena (like temperature, light, pressure, humidity, etc.) into electrical signals that can be processed by IoT devices.
3. **Actuators:** Actuators are devices that perform actions based on the data received from sensors or commands from remote systems. Examples include motors, servos, solenoids, and relays.
4. **Microcontrollers/Microprocessors:** These are the computing units within IoT devices. They process data from sensors, control actuators, and execute programmed instructions. They also manage communication with other devices and systems.
5. **Communication Modules:** IoT devices need communication capabilities to send and receive data over the internet. Communication modules can include Wi-Fi, Bluetooth, Zigbee, LoRa, cellular, Ethernet, and more.
6. **Network Infrastructure:** This includes the underlying network that enables devices to connect to the internet. It can be wired (like Ethernet) or wireless (like Wi-Fi, cellular, or satellite).
7. **Internet Connectivity:** The devices need access to the internet to communicate with other devices and central systems. This connectivity can be through Wi-Fi, cellular networks, satellite connections, or other means.
8. **Cloud Services:** Cloud platforms provide storage, computing power, and services that enable data processing, analysis, storage, and remote control of IoT devices. Cloud services facilitate the scalability and management of IoT solutions.
9. **Data Processing and Storage:** IoT devices generate a vast amount of data. Data processing involves analyzing and interpreting this data to extract meaningful insights. Data storage involves storing the collected data for future reference and analysis.
10. **User Interfaces:** User interfaces can be web applications, mobile apps, or dashboards that allow users to monitor and control IoT devices remotely. These interfaces provide real-time information and enable users to set preferences or receive alerts.
11. **Security Measures:** Security is critical in IoT to protect data, devices, and networks from unauthorized access and cyberattacks. Security measures include encryption, authentication, access controls, and regular software updates.
12. **Power Management:** IoT devices often operate on limited power sources such as batteries. Power management strategies are essential to optimize energy consumption and extend the device's operational lifespan.
13. **Data Analytics and Insights:** Analyzing the data collected from IoT devices can yield valuable insights for making informed decisions, predicting trends, and improving operations.

14. **Machine Learning and Artificial Intelligence (AI):** IoT systems can leverage machine learning and AI to improve automation, anomaly detection, and decision-making based on real-time data.

Modern Applications:

1. Smart Grids and energy saving
2. Smart cities
3. Smart homes/Home automation
4. Healthcare
5. Earthquake detection
6. Radiation detection/hazardous gas detection
7. Smartphone detection
8. Water flow monitoring
9. Traffic monitoring
10. Wearables
11. Smart door lock protection system
12. Robots and Drones
13. Healthcare and Hospitals, Telemedicine applications
14. Security
15. Biochip Transponders(For animals in farms)
16. Heart monitoring implants(Example Pacemaker, ECG real time tracking)

Advantages of IoT :

1. Improved efficiency and automation of tasks.
2. Increased convenience and accessibility of information.
3. Better monitoring and control of devices and systems.
4. Greater ability to gather and analyze data.
5. Improved decision-making.
6. Cost savings.

Disadvantages of IoT :

1. Security concerns and potential for hacking or data breaches.
2. Privacy issues related to the collection and use of personal data.
3. Dependence on technology and potential for system failures.
4. Limited standardization and interoperability among devices.
5. Complexity and increased maintenance requirements.
6. High initial investment costs.
7. Limited battery life on some devices.
8. Concerns about job displacement due to automation.
9. Limited regulation and legal framework for IoT, which can lead to confusion and uncertainty.

Overview of IoT Vs Computers:

One big difference between IoT devices and computers is that the main function of IoT devices is not to compute(not to be a computer) and the main function of a computer is to compute functions and to run programs. But on IoT devices that is not its main point, it has some other function besides that. As an example like in cars, the function of IoT devices are not to compute anti-lock breaking or to do fuel injection, their main function from the point of view of a user is to be driven and to move you from place to place and the computer is just to help that function. For example, The main function of the car is not to compute like anti-lock breaking or to do fuel injection their main function from the point of view of a user is to drive, to move you from place to place. But when we embed software in it then the software can be able for fuel limit detection.

Difference between IoT devices and Computers:

IOT Devices	Computers
IoT devices are special-purpose devices.	Computers are general-purpose devices.
IoT devices can do only a particular task for which it is designed.	Computers can do so many tasks.
The hardware and software built-in in the IoT devices are streamlined for that particular task.	The hardware and software built-in in the computers are streamlined to do many tasks(such as calculation, gaming, music player, etc.)
IoT devices can be cheaper and faster at a particular task than computers, as IoT devices are made to do that particular task.	A computer can be expensive and slower at a particular task than an IoT device.
Examples: Music Player- iPod, Alexa, smart cars, etc.	Examples: Desktop computers, Laptops, etc.

Introduction to Arduino

Arduino is a project, open-source hardware, and software platform used to design and build electronic devices. It designs and manufactures microcontroller kits and single-board interfaces for building electronics projects.

The Arduino boards were initially created to help the students with the non-technical background. The designs of Arduino boards use a variety of controllers and microprocessors.

The Arduino board consists of sets of analog and digital I/O (Input / Output) pins, which are further interfaced to **breadboard, expansion boards,** and other **circuits**. Such boards feature the model, Universal Serial Bus (**USB**), and **serial communication interfaces**, which are used for loading programs from the computers.

Types of Arduino

The flexibility of the Arduino board is enormous so that one can do anything they imagine. This board can be connected very easily to different modules such as obstacle sensors, presence detectors, fire sensors, GSM Modules GPS modules, etc.

The main function of the Arduino board is to control electronics through reading inputs & changing it into outputs because this board works like a tool. This board is also used to make different electronics projects in the field of electronics, electrical, robotics, etc.

Features of Different Types of Arduino Boards

The features of different types of Arduino boards are listed in the tabular form.

Arduino Board	Processor	Memory	Digital I/O	Analogue I/O
Arduino Uno	16Mhz ATmega328	2KB SRAM, 32KB flash	14	6 input, 0 output
Arduino Due	84MHz AT91SAM3X8E	96KB SRAM, 512KB flash	54	12 input, 2 output
Arduino Mega	16MHz ATmega2560	8KB SRAM, 256KB flash	54	16 input, 0 output
Arduino Leonardo	16MHz ATmega32u4	2.5KB SRAM, 32KB flash	20	12 input, 0 output

Different Types Of Arduino Boards

The list of Arduino boards includes the following such as

- Arduino Uno (R3)
- Arduino Nano
- Arduino Micro

- Arduino Due
- LilyPad Arduino Board
- Arduino Bluetooth
- Arduino Diecimila
- RedBoard Arduino Board
- Arduino Mega (R3) Board
- Arduino Leonardo Board
- Arduino Robot
- Arduino Esplora
- Arduino Pro Mic
- Arduino Ethernet
- Arduino Zero
- Fastest Arduino Board

Arduino Uno (R3)

The Uno is a huge option for your initial Arduino. This Arduino board depends on an ATmega328P based microcontroller. As compared with other types of arduino boards, it is very simple to use like the Arduino Mega type board. .It consists of 14-digital I/O pins, where 6-pins can be used as PWM(pulse width modulation outputs), 6-analog inputs, a reset button, a power jack, a USB connection, an In-Circuit Serial Programming header (ICSP), etc.

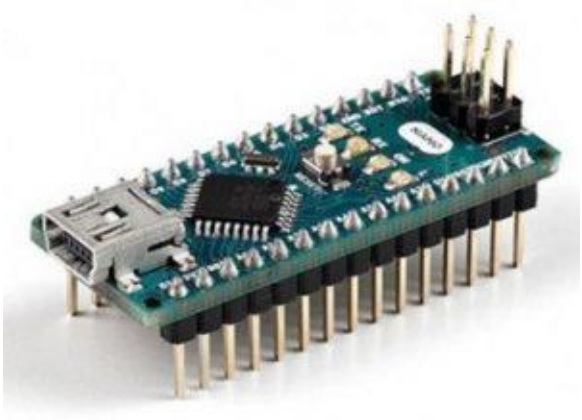
It includes everything required to hold up the microcontroller; simply attach it to a PC with the help of a USB cable and give the supply to get started with an AC-to-DC adapter or battery.

Arduino Uno (R3)

Arduino Uno is the most frequently used board and it is the standard form apart from all the existing Arduino Boards. This board is very useful for beginners. Please refer to this link to know more about Arduino Uno Board

Arduino Nano

This is a small board based on the microcontrollers like ATmega328P otherwise ATmega628 but the connection of this board is the same as to the Arduino UNO board. This kind of microcontroller board is very small in size, sustainable, flexible, and reliable.



Arduino Nano

As compared with the Arduino Uno board, it is small in size. The devices like mini USB and Arduino IDE are necessary to build the projects. This board mainly includes analog pins-8, digital pins-14 with the set of an I/O pin, power pins-6 & RST (reset) pins-2. Please refer to this link to know more about Arduino Nano Board.

Arduino Micro

The Arduino Micro board mainly depends on the ATmega32U4 based Microcontroller that includes 20-sets of pins where the 7-pins are PWM pins, 12-analog input pins. This board includes different components like an ICSP header, RST button, small USB connection, crystal oscillator-16MHz. The USB connection is inbuilt and this board is the shrunk version of the Leonardo board.

Arduino Micro

Arduino Due

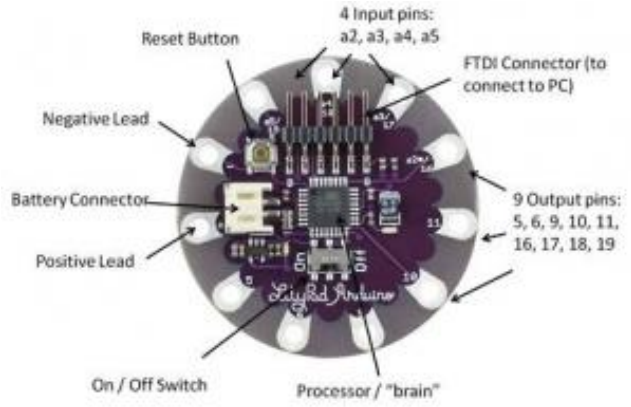
This Arduino board depends on the ARM Cortex-M3 and it is the first Arduino microcontroller board. This board includes digital I/O pins-54 where 12-pins are PWM o/p pins, analog pins -12, UARTs-4, a CLK with 84 MHz, an USB OTG, DAC-2, a power jack, TWI-2, a JTAG header, an SPI header, two buttons for reset & erase.

Arduino Due

This board works with 3.3V where the highest voltage that the pins of input/output can stand is 3.3V because providing a high voltage to any I/O pin can lead to damage the board This board is simply connected to a computer through a small USB cable otherwise it can be powered through an AC to DC adapter. This Arduino Due board is suitable with all shields of Arduino at 3.3V.

LilyPad Arduino Board

The Lily Pad Arduino board is a wearable e-textile technology expanded by Leah " Buechley" and considerately designed by "Leah and SparkFun". Each board was imaginatively designed with huge connecting pads & a smooth back to let them to be sewn into clothing using conductive thread. This Arduino also comprises of I/O, power, and also sensor boards which are built especially for e-textiles. These are even washable!



LilyPad Arduino Boards

Arduino Bluetooth

This Bluetooth mainly depends on the microcontroller like ATmega168 and this board is also called Arduino BT. This kind of board includes different components like digital pins-16, analog pins-6, crystal oscillator-16MHz, reset button, screw terminals, ICSP header. In this board, the screw terminals are mainly used for power. The programming of this Bluetooth microcontroller can be done with Bluetooth like a wireless connection.

Arduino Diecimila

The microcontroller board like Arduino Diecimila mainly depends on the ATmega168. This board includes digital I/O pins -14 where 6-pins can be used like PWM outputs & analog inputs-6, a USB connection, a crystal oscillator-16 MHz, an ICSP header, a reset button & a power jack. This board can be connected to a computer through a USB cable and it can be activated using a battery and an AC-DC adapter.

Arduino Diecimila

As the name suggests, the meaning of Diecimila in Italian is 10,000 which means that marks the truth that above 10k Arduino boards have been designed. In a set of USB Arduino boards, it is the latest one as compared with other versions.

RedBoard Arduino Board

The RedBoard Arduino board can be programmed using a Mini-B USB cable using the Arduino IDE. It will work on Windows 8 without having to modify your security settings. It is more constant due to the USB or FTDI chip we used and also it is entirely flat on the back. Creating it is very simple to utilize in the project design. Just plug the board, select the menu option to choose an Arduino UNO and you are ready to upload the program. You can control the RedBoard over a USB cable using the barrel jack.

RedBoard Arduino Boards

Arduino Mega (R3) Board

The Arduino Mega is similar to the UNO's big brother. It includes lots of digital I/O pins (from that, 14-pins can be used as PWM o/ps), 6-analog inputs, a reset button, a power jack, a USB connection, and a reset button. It includes everything required to hold up the microcontroller; simply attach it to a PC with the help of a USB cable and give the supply to get started with an AC-to-DC adapter or battery. The huge number of pins make this Arduino board very helpful for designing projects that need a bunch of digital i/ps or o/ps like lots of buttons. Please refer to this link to know more about Arduino Mega (R3) Board

Arduino Mega (R3) Board

Arduino Leonardo Board

The first development board of an Arduino is the Leonardo board. This board uses one microcontroller along with the USB. That means, it can be very simple and cheap also. Because this board handles USB directly, program libraries are obtainable which let the Arduino board to follow a keyboard of the computer, mouse, etc.

Arduino Leonardo Board

Arduino Robot

This kind of board is the first Arduino over wheels. This Arduino robot includes two processors on each of its boards. The two boards are the motor board and control board where the motor board controls the motors & the control board is used to read the sensors for operating. Every board is a complete Arduino board and its programming can be done through the Arduino IDE. These are microcontroller boards that depend on the ATmega32u4.

The pins of this Robot are mapped to actuators and sensors onboard. The process of programming the robot is the same as the Arduino Leonardo board. It is also named a small computer and it is extensively used in robotics.

This board includes the speaker, color screen, buttons-5, motors-2, a digital compass, an SD card reader, potentiometers-2 & floor sensors-5. The library of this robot can be used for controlling the sensors as well as the actuators.

Arduino Esplora

The Arduino Esplora includes a small computer known as a microcontroller including a number of inputs & outputs. The inputs of this board are a light sensor, four buttons, a microphone, an accelerometer, joystick, a slider, a temperature sensor, etc whereas the outputs are a 3 color LED, a buzzer. This kind of Arduino board looks like a videogame controller.

The programming of this board can be done using Arduino Software like IDE which takes the data from the inputs and controls the output like a keyboard or a mouse. As compared with all other types of Arduino boards, this esplora is totally different because the inputs, as well as outputs, are connected to the board already.

Arduino Esplora

So connecting the components like actuators or sensors is very simple. Thus, programming is somewhat different as compared with other types of Arduino boards. This esplora board includes its own library so that the data from the sensors & actuators are very easy to read and write.

Arduino Pro Mic

The Arduino Pro Micro board is the same as the Arduino Mini board apart from the ATmega32U4 Microcontroller. This pro mic board includes digital I/O pins-12, pulse width modulation (PWM) pins-5, serial connections of Tx & Rx &10-bit ADC.

Arduino Ethernet

The Arduino Ethernet board depends on the microcontroller like ATmega328. This kind of microcontroller board includes analog pins-5, digital I/O pins-14, RST button, an RJ45 connection, crystal oscillator, a power jack, ICSP header, etc. The connection of the Arduino board can be done through the Ethernet shield to the internet.

Arduino Zero

This is a powerful as well as simple 32-bit board and it provides the best platform for innovative projects like wearable technology, smart IoT devices, crazy robotics, high-tech automation, etc. This board expands by providing improved performance, permitting a range of project opportunities & performs like a great educational tool.

Arduino Zero

This board includes analog input pins-6, digital I/O pins-14, a power jack, AREF button, UART port pins, a USB connector & an In-Circuit Serial Programming (ICSP) header, a power header, etc.

This board is power-driven through the SAMD21 microcontroller based on Atmel. The main feature of this is EDBG (Embedded Debugger) based on Atmel and it provides a complete debug interface without using extra hardware.

Fastest Arduino Board

Designing one of the best Arduino development boards that are familiar with Arduino MEGA & UNO is the hifive1 board that includes a 320 MHz RISC-V microcontroller unit. This kind of fastest board has Cortex M-7 with a 400 MHz microcontroller unit.

- Flash memory – upto 2Mbytes
- RAM – 1 Mbyte

- DMA controllers -4
- Communication peripherals- Up to 35
- 16-bit Max Resolution with 3× ADCs
- D/A converters with 2× 12-bit
- Hardware with JPEG Codec
- Timers -22 & Watchdogs – 200Mhz
- HW Calendar & RTC with Sub-second Accuracy
- Cryptographic Acceleration

Toolchain (IDE)

A toolchain is a set of programming tools that is used to perform a complex set of operations. In the Arduino Software (IDE) the toolchain is hidden from the user, but it is used to compile and upload the user Sketch. It includes compiler, assembler, linker and Standard C & math libraries.

The source code for the Java environment is released under the GPL and the C/C++ microcontroller libraries are under the LGPL.

Sketch − Program coded in Arduino IDE is called a SKETCH

1. To create a new sketchFile -> New

-To open an existing sketch File -> open ->

-There are some basic ready-to-use sketches available in the EXAMPLES section

-File -> Examples -> select any program

2. Verify: Checks the code for compilation errors

3. Upload: Uploads the final code to the controller board

4. New: Creates a new blank sketch with basic structure

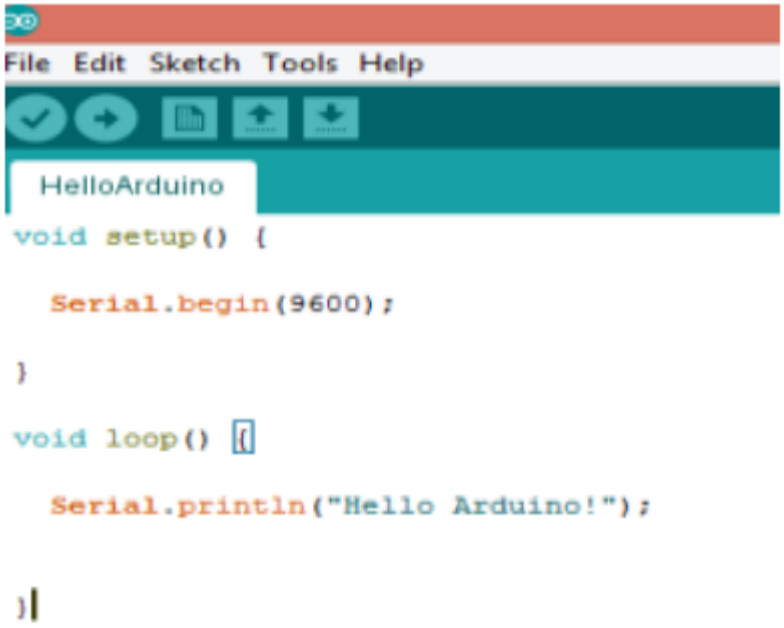
5. Open: Opens an existing sketch

6. Save: Saves the current sketch

Compilation and Execution

Serial Monitor: Opens the serial console

All the data printed to the console are displayed here



Structure of Sketch

A sketch can be divided into two parts:

- Setup ()
- Loop()

The function setup() is the point where the code starts, just like the main() function in C and C++

I/O Variables, pin modes are initialized in the Setup() function

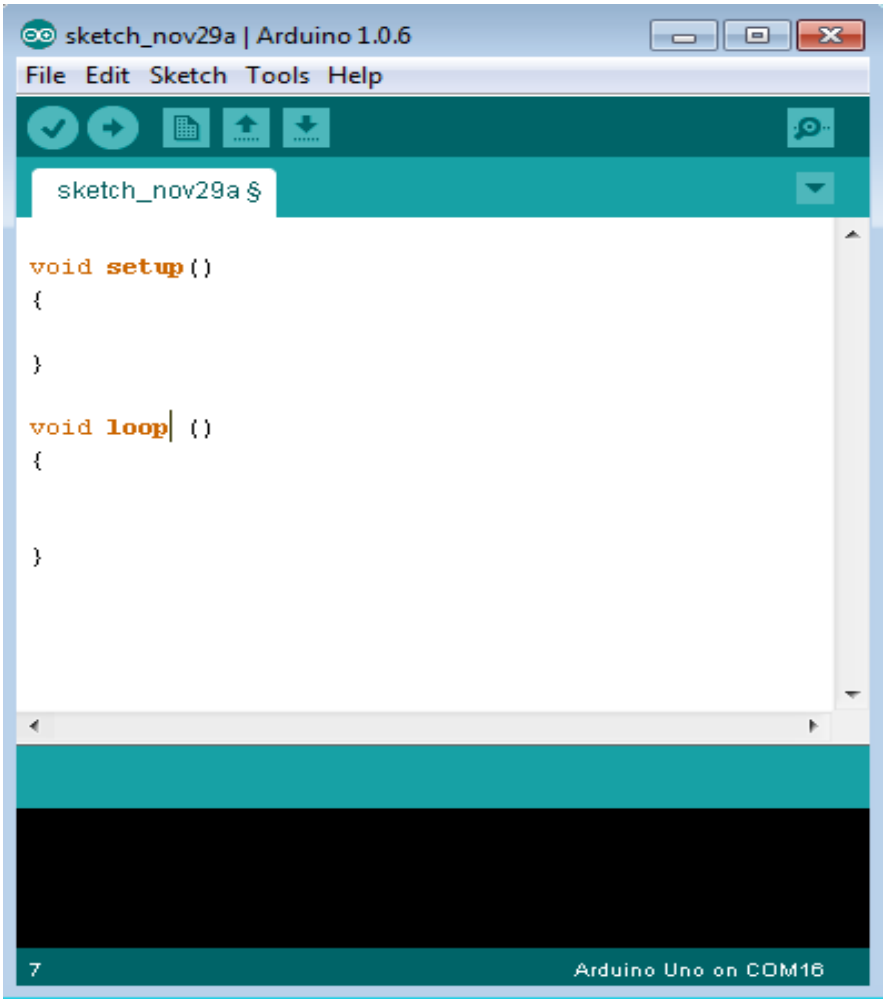
Loop() function, as the name suggests, iterates the specified task in the program

Structure

Arduino programs can be divided in three main parts: **Structure, Values** (variables and constants), and **Functions**. In this tutorial, we will learn about the Arduino software program, step by step, and how we can write the program without any syntax or compilation error.

Let us start with the **Structure**. Software structure consist of two main functions −

- Setup( ) function - The setup() function is called when a sketch starts. Use it to initialize the variables (not declaration), pin modes, start using libraries, etc. The setup function will only run once, after each power up or reset of the Arduino board.
- Loop( ) function - After creating a setup() function, which initializes and sets the initial values, the loop() function does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the Arduino board



Void setup ( ) {

}

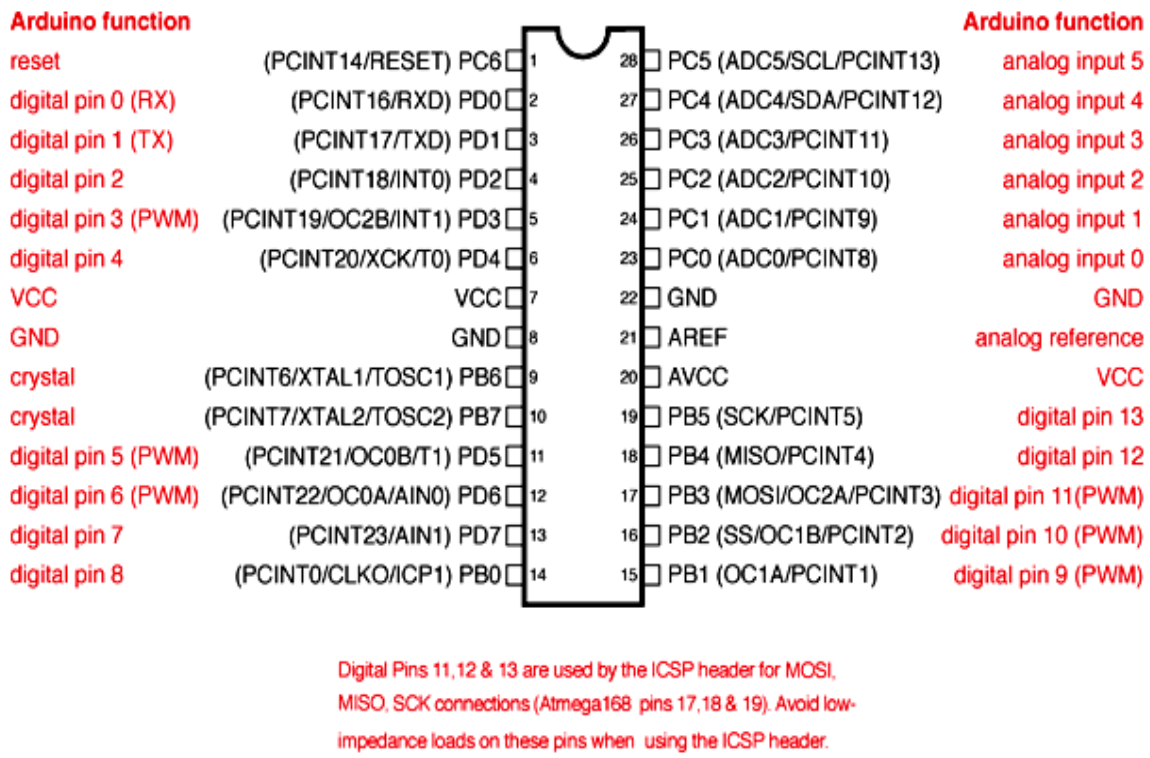
- **PURPOSE** − The **setup()** function is called when a sketch starts. Use it to initialize the variables, pin modes, start using libraries, etc. The setup function will only run once, after each power up or reset of the Arduino board.
- **INPUT** − -
- **OUTPUT** − -
- **RETURN** − -

Void Loop ( ) {

}

- **PURPOSE** − After creating a **setup()** function, which initializes and sets the initial values, the **loop()** function does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the Arduino board.
- **INPUT** − -
- **OUTPUT** − -
- **RETURN** − -

Arduino Uno Pins Diagram



Digital Pins 11,12 & 13 are used by the ICSP header for MOSI, MISO, SCK connections (Atmega168 pins 17,18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.

- Pins are wires connected to the microcontroller
- Pins are the interface of the microcontroller
- Pin voltages are *controlled* by a sketch
- Pin voltages can be *read* by a sketch

ARDUINO I/O (Input, Output) Functions

1. pinMode() -The pinMode() function is used to configure a specific pin to behave either as an input or an output.

pinMode() Function Syntax void setup () { pinMode (*pin , mode*); }	• ***pin*** – the number of the pin whose mode you wish to set • ***mode*** – INPUT, OUTPUT, or INPUT_PULLUP.

2. **digitalWrite()** - The digitalWrite() function is used to write a HIGH or a LOW value to a digital pin.

digitalWrite() Function Syntax digitalWrite (*pin ,value*) **Example** - void loop() { digitalWrite (4 ,HIGH); }	• ***pin*** – the number of the pin whose mode you wish to set • ***value*** – HIGH, or LOW.

3. digitalRead() -Reads the value from a specified digital pin, either HIGH or LOW.

digitalRead() Function Syntax
digitalRead(*pin*);
Example -

```
void loop()
{
  val = digitalRead(4);   // read the input pin
  digitalWrite(7, val);    // sets the LED to the button's
value
}
```

- ***pin*** - the number of the digital pin you want to read (*int*).

Return -
- **val** – HIGH, or LOW.

4. analogRead( ) function - we can read the voltage applied to one of the pins. This function returns a number between 0 and 1023, which represents voltages between 0 and 5 volts.

analogRead() function Syntax
analogRead(*analogPin*);
Example -

```
void loop()
{          val = analogRead(analogPin); }
```

- ***analogPin*** – the number of the analog input pin to read from (0 to 5 on most boards, 0 to 7 on the Mini and Nano, 0 to 15 on the Mega)

Return -
- int (0 to 1023)

5. analogWrite( ) function - Writes an analog value to a pin.

analogWrite() function Syntax

analogWrite(*Pin, value*)

Example -

```
void loop()
{

  val = analogRead(Pin);   // read the input pin

  analogWrite(ledPin, val / 4);  // analogRead
values go from 0 to 1023, analogWrite values
from 0 to 255

}
```

- ***Pin*** - the pin to write to.
- ***value*** - the duty cycle: between 0 (always off) and 255 (always on).

Input/Ouput (I/O) in Arduino

- These functions allow access to the pins

void pinMode(pin, Mode)

- Sets a pin to act as either an input or an output
- pin is the number of the pin
    - 1 -13 for the digital pins
    - A0 - A5 for the analog input pins
- Mode is the I/O mode the pin is set to
    - INPUT, OUTPUT

Digital Input

int digitalRead(pin)

- Returns the state of an input pin
- Returns either LOW (0 volts) or HIGH (5 volts)

int pinval;

pinval = digitalRead(3);

- pinval is set to the state of digital pin3

Digital Output

void digitalWrite(pin, value)

- Assigns the state of an output pin
- Assigns either LOW (0 volts) or HIGH (5 volts)

digitalWrite(3, HIGH);

- Digital pin 3 is set HIGH (5 volts)

Analog Input

int analogRead(pin)

- Returns the state of an analog input pin
- Returns an integer from 0 to 1023
- 0 for 0 volts, 1023 for 5 volts

int pinval;

pinval = analogRead(A3);

- Pin must be an analog pin

Analog Output

- No pins can generate a true analog output
- But PWM can be used

ARDUINO SHIELDS

The advantages of using Arduino shields are listed below:

- It adds new functionalities to the Arduino projects.
- The shields can be attached and detached easily from the Arduino board. It does not require any complex wiring.
- It is easy to connect the shields by mounting them over the Arduino board.
- The hardware components on the shields can be easily implemented.

Types of Shields

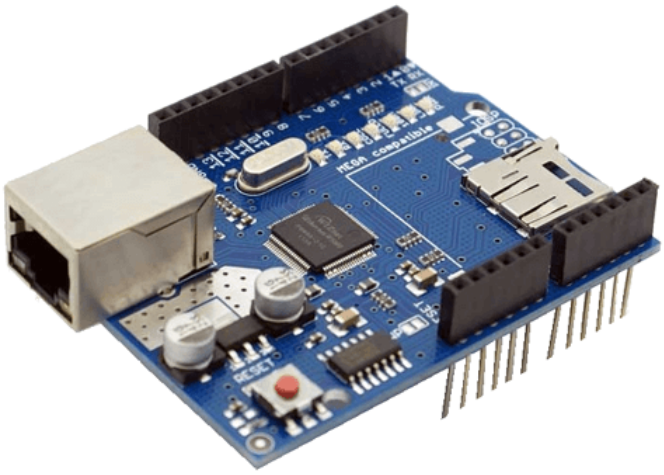
The popular Arduino shields are listed below:

- Ethernet shield
- Xbee Shield
- Proto shield
- Relay shield
- Motor shield
- LCD shield
- Bluetooth shield
- Capacitive Touchpad Shield

Ethernet shield

- The Ethernet shields are used to connect the Arduino board to the Internet. We need to mount the shield on the top of the specified Arduino board.
- The USB port will play the usual role to upload sketches on the board.
- The latest version of Ethernet shields consists of a micro SD card slot. The micro SD card slot can be interfaced with the help of the SD card library.

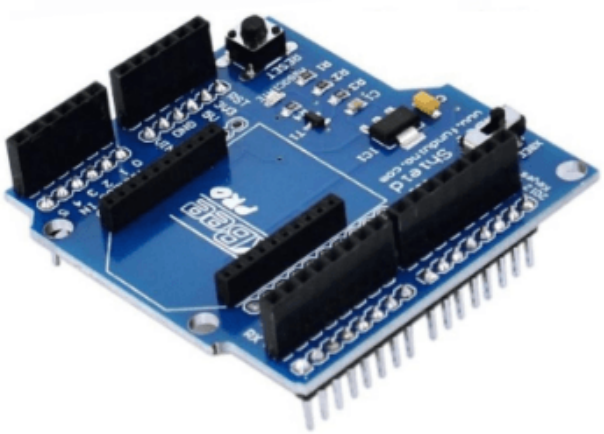
The Ethernet shield is shown below:



- We can also connect another shield on the top of the Ethernet shield. It means that we can also mount two shields on the top of the Arduino board.

Xbee Shield

- We can communicate wirelessly with the Arduino board by using the Xbee Shield with Zigbee.
- It reduces the hassle of the cable, which makes Xbee a wireless communication model.
- The Xbee wireless module allows us to communicate outdoor upto 300 feet and indoor upto 100 feet.
- The Xbee shield is shown below:



- It can also be used with different models of Xbee.

Proto shield

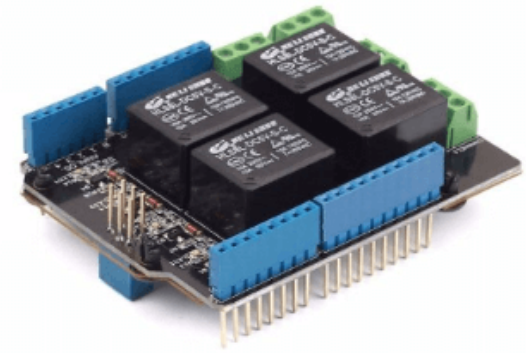
- Proto shields are designed for custom circuits.
- We can solder electronic circuits directly on the shield.
- The shield consists of two LED pads, two power lines, and SPI signal pads.
- The IOREF (Input Output voltage REFerence) and GND (Ground) are the two power lines on the board.

The proto shield is shown below:

- We can also solder the SMD (Surface Mount Device) ICs on the prototyping area. A maximum of 24 pins can be integrated onto the SMD area.

Relay shield

- The Arduino digital I/O pins cannot bear the high current due to its voltage and current limits. The relay shield is used to overcome such situation. It provides a solution for controlling the devices carrying high current and voltage.
- The shield consists of four relays and four LED indicators.
- It also provides NO/NC interfaces and a shield form factor for the simple connection to the Arduino board.
- The LED indicators depicts the ON/OFF condition of each relay.
- The relay used in the structure is of high quality.
- The NO (Normally Open), NC (Normally Closed), and COM pins are present on each relay.
- The relay shield is shown below:



- The applications of the Relay shield include remote control, etc.

Motor shield

- The motor shield helps us to control the motor using the Arduino board.
- It controls the direction and working speed of the motor. We can power the motor shield either by the external power supply through the input terminal or directly by the Arduino.
- We can also measure the absorption current of each motor with the help of the motor shield.

- The motor shield is based on the L298 chip that can drive a step motor or two DC motors. L298 chip is a full bridge IC. It also consists of the heat sinker, which increases the performance of the motor shield.
- It can drive inductive loads, such as solenoids, etc.
- The operating voltage is from 5V to 12V.

The Motor shield is shown below:

- The applications of the motor shield are intelligent vehicles, micro-robots, etc.

LCD shield

- The keypad of LCD (Liquid Crystal Display) shield includes five buttons called as up, down, left, right, and select.
- There are 6 push buttons present on the shield that can be used as a custom menu control panel.
- It consists of the 1602 white characters, which are displayed on the blue backlight LCD.
- The LED present on the board indicates the power ON.
- The five keys present on the board helps us to make the selection on menus and from board to our project.

The LCD shield is shown below:

- The LCD shield is popularly designed for the classic boards such as Duemilanove, UNO, etc.

Bluetooth shield

- The Bluetooth shield can be used as a wireless module for transparent serial communication.
- It includes a serial Bluetooth module. D0 and D1 are the serial hardware ports in the Bluetooth shield, which can be used to communicate with the two serial ports (from D0 to D7) of the Arduino board.
- We can install Groves through the two serial ports of the Bluetooth shield called a Grove connector. One Grove connector is digital, while the other is analog.

The Bluetooth shield is shown below:

- The communication distance of the Bluetooth shield is upto 10m at home without any obstacle in between.

Capacitive Touchpad shield

- It has a touchpad interface that allows to integrate the Arduino board with the touch shield.
- The Capacitive touchpad shield consists of 12 sensitive touch buttons, which includes 3 electrode connections and 9 capacitive touch pads.
- The capacitive shield is shown below:

- The board can work with the logic level of 3.3V or 5V.
- We can establish a connection to the Arduino project by touching the shield.

INTEGRATION OF SENSORS AND ACTUATORS WITH ARDUINO.

Arduino is a popular microcontroller platform that allows you to easily interface with various sensors and actuators to create interactive projects. Integrating sensors and actuators with Arduino involves the following steps:

Selecting the appropriate sensors and actuators: Identify the specific sensors and actuators you need for your project. Arduino supports a wide range of sensors such as temperature sensors, motion sensors, light sensors, and actuators such as motors, servos, and LEDs.

Actuators are basically mechanical or electro mechanical devices. They convert energy or signals into motion. And mainly use to provide controlled motion to other components of various mechanical structures or devices.

Power supply: Ensure that your sensors and actuators have a suitable power supply. Arduino boards can usually provide power to low-power sensors and actuators directly from their digital or analog pins. However, high-power devices like motors may require an external power source.

Wiring connections: Connect the sensors and actuators to the Arduino board using jumper wires or appropriate connectors. Arduino boards typically have digital input/output (I/O) pins, analog input pins, and power pins that you can use for wiring.

For digital sensors and actuators, you can connect them to the digital I/O pins. These pins can be configured as either input or output. Analog sensors can be connected to the analog input pins, which can read continuous voltage values.

Power and ground connections should be made to provide appropriate voltage and common reference for the sensors and actuators.

Library installation: Some sensors and actuators require specific libraries to be installed in the Arduino Integrated Development Environment (IDE). These libraries provide pre-written code and functions that make it easier to interface with the devices. You can install libraries by navigating to Sketch -> Include Library -> Manage Libraries in the Arduino IDE.

Code development: Write the Arduino code to read sensor data and control the actuators. The code should include appropriate functions and commands to initialize the sensors, read their values, and control the actuators based on the sensor inputs. The specific code will depend on the sensors and actuators you are using, so refer to their respective documentation and examples.

Uploading and testing: Upload the code to the Arduino board and test the integration. Use the Serial Monitor in the Arduino IDE to view sensor readings or debug any issues. Make sure the sensors and actuators are functioning as expected.

By following these steps, you can integrate sensors and actuators with Arduino to create interactive projects and prototypes. The possibilities are vast, ranging from simple projects like temperature monitoring and LED control to complex robotics and automation systems.