6.1 IOT HARDWARE

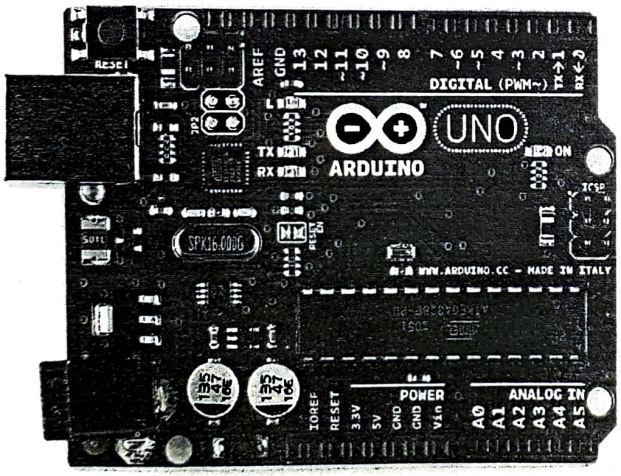
When it comes to IoT hardware, one can think of mobile phones as IoT devices, since smartphones have sensors, dis-plays, and a unique address and are connected to the Inter-net. Regarding IoT devices, Paul Jacobs, former chief executive officer of Qualcomm, has said, "In the future, almost all things will be linked on the web, and mobile phones will act as hubs for IoT. So, IoT is nothing but the Internet linkage of smart objects and embedded systems other than mobile phones, with mobiles phones acting as access centers for IoT". The term smart objects referred to by Jacobs can be described as things or objects that are responsible for providing useful information on their interac-

6.1.1 ARDUINO

GQ. Explain features of Arduino. **(2 Marks)**

Introduction

- Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

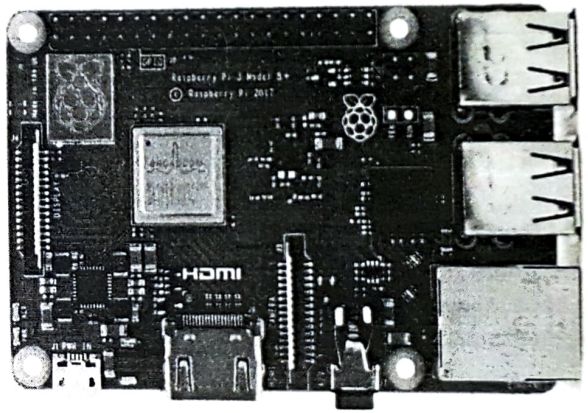


(1F1)**Fig. 6.1.1 : Arduino**

- The Arduino platform has become quite popular with people just starting out with electronics, and for good reason.
- Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.
- Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

6.1.2 What is a Raspberry Pi?

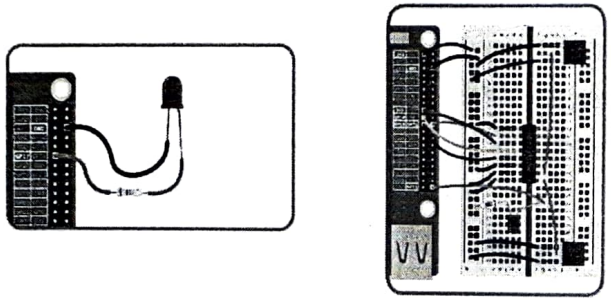
GQ. Explain Raspberry Pi. **(4 Marks)**



(1F2)**Fig. 6.1.2 : Raspberry Pi**

- Raspberry Pi is the name of a series of single-board computers made by the Raspberry Pi Foundation, a UK charity that aims to educate people in computing and create easier access to computing education.
- The Raspberry Pi launched in 2012, and there have been several iterations and variations released since then. The original Pi had a single-core 700MHz CPU and just 256MB RAM, and the latest model has a quad-core CPU clocking in at over 1.5GHz, and 4GB RAM. The price point for Raspberry Pi has always been under $100 (usually around $35 USD), most notably the Pi Zero, which costs just $5.

- All over the world, people use the Raspberry Pi to learn programming skills, build hardware projects, do home automation, implement Kubernetes clusters and Edge computing, and even use them in industrial applications.
- The Raspberry Pi is a very cheap computer that runs Linux, but it also provides a set of GPIO (general purpose input/output) pins, allowing you to control electronic components for physical computing and explore the Internet of Things (IoT).



(1F2)Fig. 6.1.3 : GIO Pins

See Getting started with Raspberry Pi and download the Raspberry Pi cheat sheet.

What Raspberry Pi models have been released?

There have been many generations of the Raspberry Pi line: from Pi 1 to 4, and even a Pi 400. There has generally been a Model A and a Model B of most generations. Model A has been a less expensive variant, and tends to have reduced RAM and fewer ports (such as USB and Ethernet). The Pi Zero is a spinoff of the original (Pi 1) generation, made even smaller and cheaper. Here's the lineup so far:

- Pi 1 Model B (2012)
- Pi 1 Model A (2013)
- Pi 1 Model B+ (2014)
- Pi 1 Model A+ (2014)
- Pi 2 Model B (2015)
- Pi Zero (2015)
- Pi 3 Model B (2016)
- Pi Zero W (2017)

- Pi 3 Model B+ (2018)
- Pi 3 Model A+ (2019)
- Pi 4 Model A (2019)
- Pi 4 Model B (2020)
- Pi 400 (2021)

6.1.3 ESP32

1.1 About

- ESP32 is a series of low cost, low power system on a chip microcontrollers with integrated Wi-Fi & dual-mode Bluetooth. The ESP32 series employs a Tensilica Xtensa LX6 microprocessor in both dual-core and single-core variations.
- ESP32 is created and developed by Espressif Systems, a Shanghai-based Chinese company, and is manufactured by TSMC using their 40 nm process. It is a successor to the ESP8266 micro controller.

6.2 FEATURES OF THE ESP32 INCLUDE THE FOLLOWING

GQ. Explain features of ESP 32.

- CPU: Xtensa Dual-Core 32-bit LX6 microprocessor, operating at 160 or 240 MHz and performing at up to 600 DMIPS
- Memory: 520 KiB SRAM
- Wireless connectivity:

  + Wi-Fi: 802.11 b/g/n/e/i

  + Bluetooth: v4.2 BR/EDR and BLE
- Peripheral interfaces:

  + 12-bit SAR ADC up to 18 channels

  + 2 × 8-bit DACs

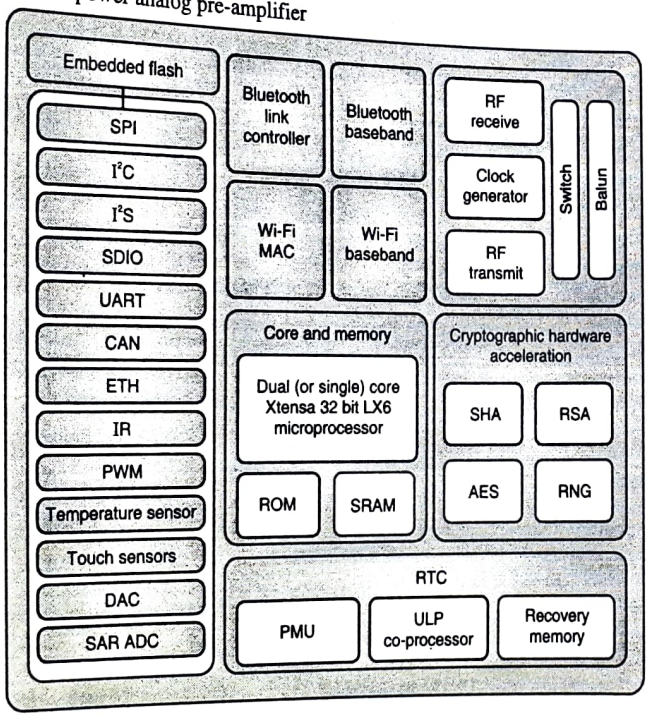
  + 10 × touch sensors

  + Temperature sensor

  + 4 × SPI

  + 2 × I²S

+ 2 × I²C
+ 3 × UART
+ SD/SDIO/MMC host
+ Slave (SDIO/SPI)
+ Ethernet MAC interface with dedicated DMA and IEEE 1588 support
+ CAN bus 2.0
+ IR (TX/RX)
+ Motor PWM
+ LED PWM up to 16 channels
+ Hall effect sensor
+ Ultra low power analog pre-amplifier



(1F4)**Fig. 6.2.1 : ESP32 block diagram**

- Security:
  + IEEE 802.11 standard security features all supported, including WFA, WPA/WPA2 and WAPI
  + Secure boot
  + Flash encryption
  + 1024-bit OTP, up to 768-bit for customers
  + Cryptographic hardware acceleration: AES, SHA-2, RSA, elliptic curve cryptography (ECC), random number generator (RNG)

6.3 ESP8266 VS ESP32

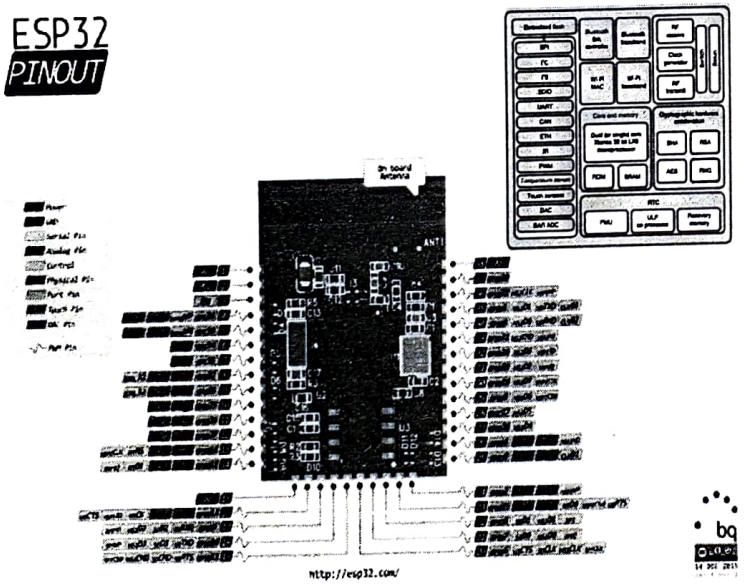
Specifications	ESP8266	ESP12
MCU	Xtensa@ Single-Core 32-bit L106	Xtensa@ Dual-Core 32-bit LX6 600 DMIPS
802.11 b/g/n Wi-Fi	Yes, HT20	Yes HT40
Bluetooth	None	Bluetooth 4.2 and below
Typical Frequency	80 MHz	160 MHz
SRAM	160 kBytes	512 kBytes
Flash	SPI flash. up to 16 Mbytes	SPI flash. up to 16 Mbytes
GINO	17	36
Hardware / Software PWM	None / 8 Channels	1/16 Channels
SPI/12C / 12S / UART	2/1/2/2	4/2/2/2
ADC	10 bit	12 bit
CAN	None	1
Ethernet MAC Interlace	None	1
Touch Sensor	None	Yes
Temperature Sensor	None	Yes
Working Temperature	– 40°C – 125°C	– 40°C – 125°C

Fig. 6.3.1 : ESP8266 vs ESP322

Module boards

ESP32 module boards are small PCBs which directly contain the ESP32 SoC and are designed to be easily used by other circuit boards. Meandered inverted-F antenna designs are used for the PCB trace antennas on the modules listed below. 2 popular module boards same features but different pinout and Vendor.

Vendor	Name	Antenna	Flash memory (DEB)	Description
Espressif	ESP-WR 00M-03	PCB trace	4	Limited distribution, pre7jai-odustiop module created by .Ew-essif for beta testing purposes; this module used the E5P31B, the beta resting chip for the ESP32 series
	ESP-ROOM-32	PCB trace	4	Flagship, public-release ESP32 module board created by Espressif.
Ai-Thinker	ESP-325	PCB trace	4	E5P32 module based On the farm factor of the Espressif ESP-WROOM-32 module. The ESP-325 module replaced the unreleased ESP3212 module

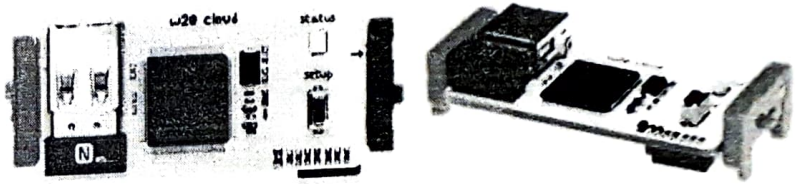


(1F5)**Fig. 6.3.2 : ESP32S module**

6.3.1 littleBits CloudBit Wi-Fi Module Simplifies DIY IoT Designs

GQ. Explain littleBits CloudBit Wi-Fi Module. **(4 Marks)**

- littleBits Electronics is a company selling tiny modules that snap together with tiny magnets for prototyping called littleBits. They do not require soldering, wiring, or programming, can be buttons, sensors, motors, etc., and are the electronics equivalent of LEGO, and are suitable to 8 years old and older kids.
- The company have recently launched a new product called CloudBit, a module based on Freescale i.MX233 with Wi-Fi connectivity meant to be used/snapped with existing littleBits.



(1F6)**Fig. 6.3.3 : little bits**

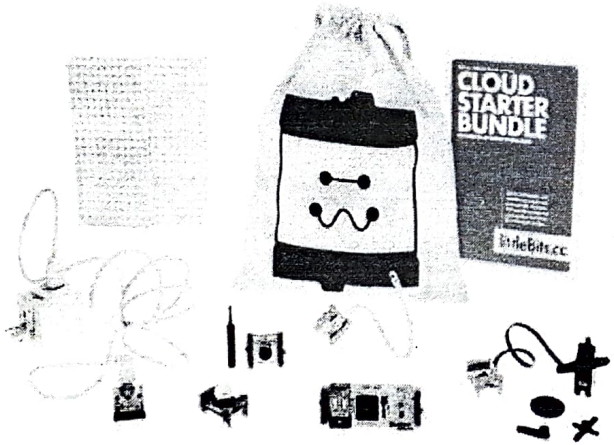
CloudBit hardware specifications

- Processor – Freescale i.MX233 ARM926EJ-S processor @ 454MHz
- System Memory – 64MB of RAM;
- Storage – microSD slot with included 4GB micro SD card pre-loaded with a customized Arch Linux ARM distribution
- Connectivity – 802.11b/g Wi-Fi via included USB dongle
- USB – micro USB port (for power only)
- Connectors – 2x BitSnap connectors for LittleBits connectivity using i.MX233 ADC/DAC signals
- Debugging – Pads for UART (3.3V, 8-N-1, 115,200 baud) to access the serial console (bottom of the board)
- Misc -Status LED, Setup button
- Power – via USB (power module, wall adapter, and cable included)

- Dimensions – 15 x 10 x 5mm
- Weight – 154 grams

CloudBit also includes a USB power module, and a wall adapter with cable. It runs Arch Linux ARM and leverages node.js technologies. The overall system diagram can be found here.

This little module allows you to connect virtually any device to the Internet, such as a thermostat that turns on when it's too hot or cold, a doorbell that send an SMS or an email, etc... All that "without programming, soldering or wiring required", the company claims. So how do you control it? You can use IFTTT "If this then that" app to connect to online services such as Facebook, Gmail and Twitter, as well as compatible hardware such as Nest and Philips HUE. Although programming CloudBit is not required, more advanced users can still do with via the Cloud API or littleBits Arduino module.



(IF7)Fig. 6.3.4 : Cloud bits

☞ **Cloud Starter Bundle**

- If you are new to littleBits, the CloudBit won't be useful by itself, and that's why the company also offers a Cloud Starter Bundle with CloudBit, the USB power module and wall adapter, but also several littleBits modules namely a "long" LED, a button, a servo, a sound trigger, as well as a mounting board, a sort of breadboard for the company's modules.

- The cloudBit and littleBits can interact with the web and each other in three ways :
  (1) **Bits to Web :** Using hardware to communicate with web services and software
  (2) **Web to Bits :** Communicating events in the web to the CloudBit, using for example, the company's Cloud Control or the third party IFTTT app.
  (3) **Bits to Bits :** Communicating from machine to machine
- The company features several demo projects with instructions including a chicken feed monitoring system, a remote fish/pet feeder, a baby monitor, an SMS doorbell, etc... and they also provide a few IFTTT samples, as tutorials.
- You can find all the documentation you need on CloudBit and Cloud Starter Bundles product pages, as well as purchase them respectively for $59 and $99, plus shipping.

6.3.2 Introduction To Particle

GQ. Write a short note on Particle. **(2 Marks)**

Particle

Fig. 6.3.5

Introduction

- Particle is an Internet of Things device platform which enables a developer to quickly and easily build, connect and manage their connected systems/applications.
- It provides ease for connecting things to the Internet/Web.
- Particle has come up with the different Internet of Things development kit which is mainly designed for creating IoT based applications.

- Particle's IoT based platform provides everything that is necessary to build a connected system/application like a smart home.
- All the Particle Devices come with free access to the Particle Cloud. The Cloud serves as the gateway between your devices and the web.
- The Particle Cloud has some great features for building connected projects, including Over-The-Air (OTA) firmware updates, an easy-to-use REST API, and firmware development supported by web and local IDEs.
- OTA (Over-The-Air) technique is useful for wirelessly updating firmware and configuration settings of Particle IoT devices remotely.
- Particle comes with access to a set of development tools - a Web IDE, Desktop IDE(Dev) and a CLI (Command Line Interface).

Why Particle?

- Particle provides many types of boards related to the Internet of Things (IoT) platforms which are useful for various applications.
- Particle provides its own web IDE, Dev IDE (Integrated Development Environment) and command line interface (CLI) which is free to download from Particle website and use.
- The main thing in Particle is you do not require a cable connection while flashing the code or programming, it can flash the code over the air (OTA).
- The program structure for Particle is built with Arduino compatible. This allows us to compile and run codes as it is from Arduino.
- The same libraries of Arduino can be useful for the particle photon. The functions implemented in these libraries can be used for quick development purposes.
- This is very useful for developers who are focused on building innovative applications and proof of concepts. The developers can spend more time on developing the applications rather than on developing the sub-modules used in building the applications.

Particle's Devices/Boards

Particle has designed various development boards which are useful for building IoT based systems/applications. The list of these boards is as follows,

☞ **Particle Photon (Wi-Fi)**

Particle Photon is a very small Wi-Fi development kit which is designed for creating connected projects and applications for the Internet of Things (IoT). It has a powerful 120Mhz ARM Cortex M3 microcontroller with an on-board Broadcom Wi-Fi chip.

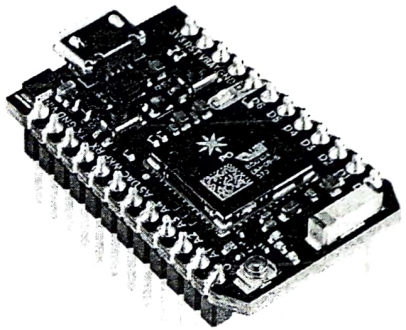


Fig. 6.3.6 particle Photon (Wi fi)

☞ **Particle Photon**

Particle Electron (Cellular)

- Particle Electron is a tiny Cellular(2G/3G) based development kit that can be used for creating connected projects and applications. It comes with a Particle SIM card with a data plan for low bandwidth applications.

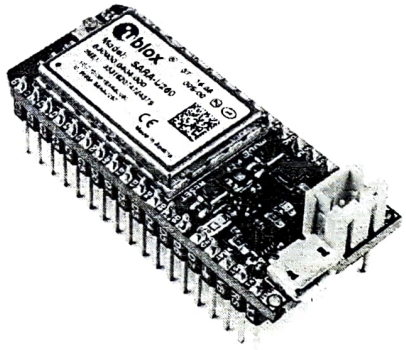


Fig. 6.3.7 Particle Electron (Cellular)

☞ **Particle Electron**

Particle's IDE

Particle has 3 different IDE platforms for developing applications.

Web IDE

- Particle's Web IDE is a powerful development environment which can run in your favorite browser. It doesn't require any setup to use.
- Using Web IDE, we can develop debug, compile and flash our devices from anywhere (OTA) in the world.
- We can access many sample code examples and hundreds of firmware libraries from any computer with an active Internet connection.

To know more about Web IDE, you can refer Web IDE (Build).

Desktop IDE (Dev)

- Particle's Desktop IDE is used as a local development environment. It is very easy to use the IDE.
- It provides advanced features that make managing large or complicated firmware projects fast, easy and efficient. But it requires internet access as the Desktop IDE is not an offline development tool. It uses the internet to push files to the cloud for compilation and returns binary.
- It is easy to download and install for Windows, Linux, and MacOS. To download Desktop IDE, you can visit Particle Desktop IDE.
- To know more about Particle Desktop IDE, you can refer Particle Desktop IDE.

Command Line Interface (CLI)

- Particle has CLI (Command Line Interface) which is a powerful tool for interacting with particle's devices and Particle Cloud. It uses node.js and can easily run on Windows, Linux, and MacOS.

6.3.3 BeagleBone

- BeagleBone is an "opensource hardware" which is having a credit card sized foam factor. It was the first in its kind, which becomes famous for its small size, but having high capability.
- Unlike its ancestors such as BeagleBoard or BeagleBoard-xM, it has plenty of input-output pins which can be used by hackers and hardware lovers for interfacing linux with their favorite sensors without any kernel level modifications.

- Moreover, its arduino like design can be used to attach plenty of capes, which will add wings to itscapabilities even further.

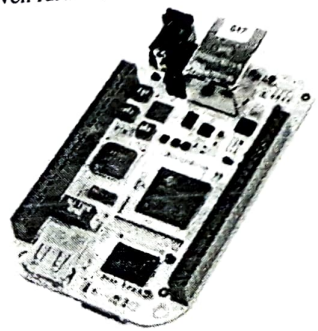


Fig. 6.3.8

- Major features are as follows
  (1) 256MB DDR2 RAM
  (2) 3D graphics accelerator
  (3) ARM Cortex-M3 for power management
  (4) 2x PRU 32-bit RISC CPUs
  (5) USB client: power, debug and device
  (6) USB host
  (7) Ethernet
  (8) 2x 46 pin headers
- Many opensource projects based on BeagleBone can be found under http://beagleboard.org/project. Getting Started page contains all the necessary informations and drivers for getting the board up out of the box in just few minutes.
- The SD Card which comes bundled with the pack contains Angstrom Linux Distribution, has all the necessary tools to begin development. This production version image can also be obtained from here. Apart from the Angstrom distribution users can easily switch to other distros like Ubuntu, ArchLinux or Android. More hacking tips on BeagleBone will be made available soon

- Internet of Things (IoT) concerned with the network of physical devices that are embedded with several technologies in order to connect, communicate and share the data with each other over the network.
- The major components of the IoT are connectivity, integration, cloud computing, sensing, & various others and the technology has its applications in multiple areas whether it be Smart IoT devices for Homes, Health Care, Automation, Retail, etc.
- There is no doubt about the fact that the Internet of Things is the next big thing in the Information Technology industry. Most of the developers and technical enthusiasts have already focused on learning the new skills required to pursue the career. In this whitepaper, we list down the 8 popular open source popular programming languages for IoT development.

6.4 IOT SOFTWARE - LANGUAGES FOR PROGRAMMING IOT HARDWARE

GQ. Explain different IOT software. **(4 Marks)**

1. **JAVA**
   - When it comes to IoT Development, JAVA stands out among the most popular programming languages.
   - One of the prominent features that make JAVA favorable for the Internet of Things (IoT) Development is Write Once, Run Anywhere concept which implies that the compiled JAVA code can run on any platform that supports the language without compiling it again.
   - In general, the JAVA codes are compiled to byte code that can run on any JAVA Virtual Machine conveniently. Moreover, the object-oriented language allows you to build the applications compatible for both – Edge nodes as well as Cloud.
   - Furthermore, the languages come up with various other renowned features such as an *extensive built-in library*, *highly interoperable*, etc. beneficial for the IoT Development.
2. **Python**
   - Python is another most-recommended programming language compatible for the IoT Development. It is an interpreted language that supports the programming standards of object-oriented programming as well as functional and structured programming.

   - The high-level programming language has an easier syntax and better code readability that makes it one of the most preferred languages for IoT by the developers.
   - Also, the language can work on various platforms such as Windows, Linux, etc. and can be integrated with other languages such as *C++*, *Java*, etc. conveniently.
   - Moreover, the language has *rich library support*, *large community support*, and various other features, and also it is much suitable for data-intensive applications.
3. **C**
   - How can we forget this much-acclaimed programming language!! C can be considered as one of the most widely used programming languages in the Internet of Things (IoT) world.
   - The middle-Level programming language allows you to understand the underlying architecture of programming that *provides the required flexibility to the IoT Developers*. Moreover, the language has several other prominent features also such as *portability*, *rich library*, and many more.
   - Furthermore, the language is pretty much compatible with the micro-controllers required for the IoT devices. However, it requires more effort and time as well to learn C Language effectively due to its not-so-easy syntax and layered architecture.
4. **LUA**
   - However, LUA is not one of the usual names in the computer programming word but when it comes to IoT Development, it has already made its strong presence among the developers.
   - LUA is a *general-purpose, high-level programming language* that is specifically designed for embedded purposes. The extensible procedural language is aimed to support data description facilities and it is required to be embedded in a host client for successful functioning.
   - Moreover, LUA comes up with its most preferred framework *Node.lua built on a lightweight LUA interpreter* that helps the developers to create IoT-based applications and various other enriching features such as better efficiency, portability, etc.
5. **Golang**
   - Golang, sometimes referred to as Go, is also one of those best languages that can be taken into consideration for IoT Development. In general, Golang is an open-source statically typed programming language, developed by Robert Griesemer, Rob Pike, and Ken Thompson at Google.

- The language offers several prominent features such as inbuilt concurrency (goroutines & channels) and the ability for the maximum usage of the hardware that makes it more compatible and relevant for the IoT development.
- Furthermore, the language provides several other crucial features also such as the rich standard library, dynamic-typing capability, etc. that can also be considered by the developers.

6. PHPoC

- If you're familiar with PHP (Hypertext Preprocessor) Language then to relate with the PHPoC (PHP on Chip) is not a big deal for you!! Meanwhile, PHPoC (PHP on Chip) is a programming language (based on PHP Language) and an IoT hardware platform. Even the syntax of PHPoC is almost similar to the PHP language.
- However, it can be considered that PHPoC is not only a web development language but also a general-purpose programming language compatible and suitable for IoT.
- Moreover, apart from the core PHP functions, PHPoC also includes several *additional functions like SPI, UART, RTC,* and various others beneficial for the IoT Development.

7. Swift

- Last but not least – Swift!! If we talk about the language introduction – Swift is a *general-purpose, multi-paradigm programming language* that is specifically designed to create applications for iOS, iPadOS, macOS, watchOS, and tvOS.
- The language comes up with several prominent features such as *powerful error handling, functional programming patterns, fast & secure,* and many more.
- However, as mentioned above that Swift is particularly concerned with the development of applications for Apple's devices so if you're looking forward to doing IoT development for these particular platforms such as iOS, macOS, etc. then you're strongly recommended to opt for Swift otherwise you can go with other languages as well.
- So these are several programming languages that you can take into consideration to learn IoT Development. Meanwhile, you can opt for a language (from the above-mentioned list or other than that) based on your own preferences, *for example – if you're looking for a language for IoT development with the easier syntax you can go with Python or if want to learn IoT for iOs & macOS devices you can opt for Swift, and so on.*

- However, whichever language you'd choose, you're required to do hard work with all the dedication and consistency to accomplish your goals!!

☞ Open-source Middleware Software Solutions

Open-source middleware application development tools are free to use and thus, they significantly reduce the overall project costs. Many of these platforms are used by a large number of small-to-medium enterprises as well as fortune 500 companies. Below are some examples of the most sought-after open-source middleware platforms that are also effective at building IoT middleware software solutions.

#1 Talend

- Talend, a market leader in cloud data integration, provides open-source middleware software solutions for enterprises to strengthen their software infrastructure.
- Talend's middleware application development platform enables enterprises to bridge the gap between disparate software components and heterogeneous enterprise IoT applications.
- Besides, it renders complete support to address a wide range of data integration and implementation requirements through middleware application development.
- Talend provides a unified software application suite that provides dedicated tools for diverse middleware project requirements. Other features include built-in data quality and data governance capabilities.
- Most importantly, developers don't need additional tools or switch between different software environments since they have all the required tools at their disposal.

#2 Apache Camel

- Apache Camel is a Java-based software integration framework that provides message-oriented middleware software solutions for varied business needs.
- It is an open-source framework with a rule-based routing and mediation engine that enables developers to implement enterprise integration patterns using custom APIs. In doing so, they can easily configure the given routing and mediation rules.
- Apache Camel supports several functions including Bean Binding (for Java objects) and JavaBeans. As a result, it makes it easy for developers to integrate many heterogeneous applications regardless of their software model or underlying technologies.

- Apache Camel is quite often used with several other software platforms like Apache ServiceMix, Apache CXF, and Apache ActiveMQ to address different project requirements.

#3 MuleSoft ESB

- Mule ESB (enterprise service bus) is an open-source software platform that incorporates Java-based programming for middleware application development.
- It is a lightweight integration platform by MuleSoft Inc. that enables developers to interconnect various applications, software components, and distributed systems regardless of their heterogeneity. Mule ESB is effective at handling a diverse range of applications built on top of the technologies like HTTP, JMS, JDBC, web services, and many others.
- Developers can deploy the ESB anywhere to integrate or orchestrate various events in several batches. Mule ESB offers universal connectivity and can also be deployed in real-time for software integration and orchestration.

☞ **Zetta**

Fig. 6.4.1

Zetta is API based IoT platform based on Node.js. It is considered as a complete toolkit to make HTTP APIs for devices. Zetta combines REST APIs, WebSockets to make data-intensive and real-time applications. The following are some notable features.

- It can run on the cloud, or a PC, or even modest development boards.
- Easy interface and necessary programming to control sensors, actuators, and controllers.
- Allows developers to assemble smartphone apps, device apps, and cloud apps.
- It is developed for data-intensive and real-time applications.
- Turns any machine into an API.

☞ **Arduino**

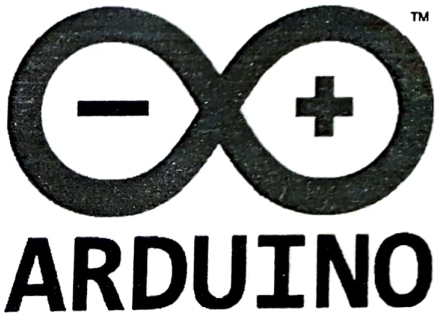


Fig. 6.4.2

- If you are seeking to make a computer that can perceive and exercise stronger control over the real world when related to your ordinary stand-alone computer, then Arduino can be your wise preference.
- Offering an appropriate blend of IoT hardware and software, Arduino is a simple-to-use IoT platform. It operates through an array of hardware specifications that can be given to interactive electronics. The software of Arduino comes in the plan of the Arduino programming language and Integrated Development Environment (IDE).

☞ **OpenRemote**

OpenRemote has introduced a new open-source IoT platform to create professional energy management, crowd management, or more generic asset management applications.

Summing up the most important features:

- Generic asset and attribute model with different asset types
- Protocol agents like HTTP REST or MQTT to connect your IoT devices, gateways, or data services or build a missing vendor-specific API.
- Flow editor for data processing, and a WHEN-THEN and a Groovy UI for event-based rules.
- Standard Dashboard for provisioning, automating, controlling, and monitoring your application as well as Web UI components to build project-specific apps.
- Android and iOS consoles which allow you to connect to your phone services, e.g., geofences, and push notifications.

- Edge Gateway solution to connect multiple instances with a central management instance.
- Multi-realms multi-tenant solution, combined with account management and identity service.

☞ **Node-RED**

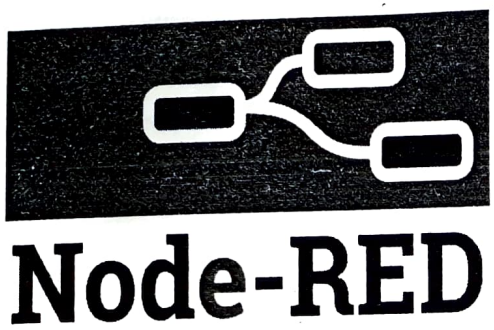


Fig. 6.4.3

- Node-RED is a visual tool for lining the Internet of Things, i.e., wiring together hardware devices, APIs, and online services in new ways. Built on Node.js, Node-RED describes itself as "a visual means for wiring the Internet of Things."
- It provides developers to connect devices, services, and APIs using a browser-based flow editor. It can run on Raspberry Pi, and further 60,000 modules are accessible to increase its facilities.

☞ **Flutter**

Fig. 6.4.4

- Flutter is a programmable processor core for electronics projects, designed for students, and engineers. Flutter's take to glory is it's long-range. This Arduino-based board includes a wireless transmitter that can show up to more than a half-mile. Plus, you don't require a router; flutter boards can interact with each other quickly.

- It consists of 256-bit AES encryption, and it's simple to use. Some of the other features are below.
  - Fast Performance
  - Expressive and Flexible UI
  - Native Performance
  - Visual finish and functionality of existing widgets.

☞ **M2MLabs Mainspring**

- M2MLabs Mainspring is an application framework for developing a machine to machines (M2M) applications such as remote control, fleet administration, or smart terminal. Its facilities include flexible design of devices, device structure, connection between machines and applications, validation and normalization of data, long-term data repository, and data retrieval functions.
- It's based on Java and the Apache Cassandra NoSQL database. M2M applications can be modeled in hours rather than weeks and subsequently passed on to a high-performance execution environment made on top of a standard J2EE server and the highly-scalable Apache Cassandra database.

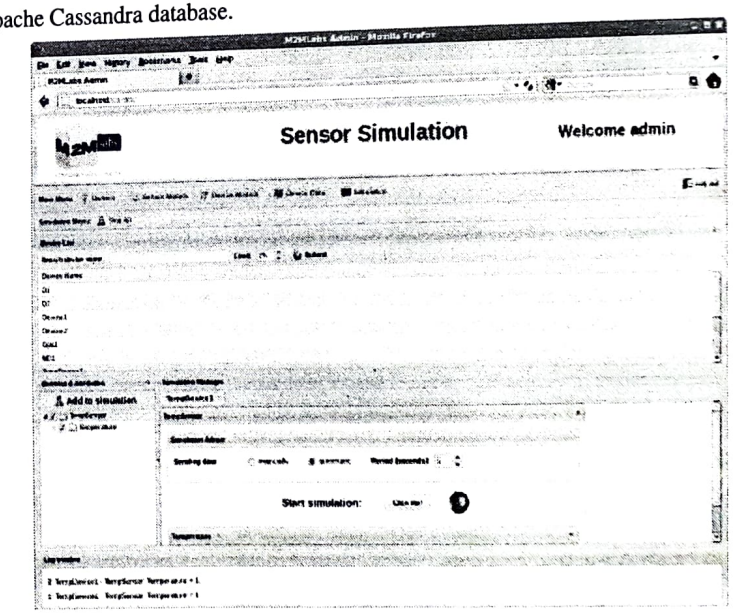


Fig. 6.4.5 M2MLabs

☞ **ThingsBoard**

ThingsBoard

Fig. 6.4.6

ThingsBoard is for data collection, processing, visualization, and device management. It upholds all standard IoT protocols like CoAP, MQTT, and HTTP as quickly as cloud and on-premise deployments. It builds workflows based on design life cycle events, REST API events, RPC requests.

Let's take a look at the following ThigsBoard features.

- A stable platform that is combining scalability, production, and fault-tolerance.
- Easy control of all connected devices in an exceptionally secure system
- Transforms and normalizes device inputs and facilitates alarms for generating alerts on all telemetry events, restores, and inactivity.
- Enables use-state specific features using customizable rule groups.
- Handles millions of devices at the same time.
- No single moment of failure, as every node in the bundle is exact.
- Multi-tenant installations out-of-the-wrap.
- Thirty highly customized dashboard widgets for successful user access.

☞ **Kinoma**

- Kinoma, a Marvell Semiconductor hardware prototyping platform, involves three different open source projects.
- Kinoma Create is a DIY construction kit for prototyping electronic devices.

Kinoma

Fig. 6.4.7

- Kinoma Studio is the development environment that functions with Set up and the Kinoma Platform Runtime.
- Kinoma Connect is a free iOS and Android app that links smartphones and stands with IoT devices.

☞ **Kaa IoT Platform**

Fig. 6.4.8 Kaa IOT

Kaa is a production-ready, flexible, multi-purpose middleware platform for establishing end-to-end IoT solutions, connected applications, and smart devices. It gives a comprehensive way of carrying out effective communication, deals with, and interoperation capabilities in connected and intelligent devices.

It mounts from tiny startups to a great enterprise and holds advanced deployment models for multi-cloud IoT solutions. It is primarily based on flexible microservices and readily conforms to virtually any need and application some other features as below.

- Facilitates cross-device interoperability.
- Performs real-time device control, remote device provisioning, and structure.
- Create cloud services for smart products
- Consists of topic-based warning systems to provide end-users to deliver messages of any predefined format to subscribed endpoints.
- Perform real-time device monitoring
- Manage an infinite quantity of connected devices
- Collect and analyze sensor data

☞ **SiteWhere**

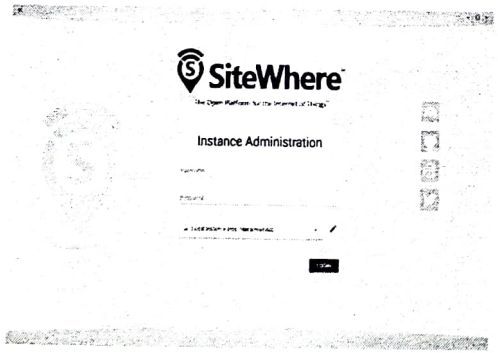


Fig. 6.4.9

SiteWhere platform offers the ingestion, repository, processing, and assimilation of device inputs. It runs on Apache Tomcat and provides highly tuned MongoDB and HBase implementations. You can deploy Site Where to cloud platforms like AWS, Azure, GCP, or on-premises. It also supports Kubernetes cluster provisioning.

The following are some of the other features.

- Run any estimate of IoT applications on a single SiteWhere instance
- Spring brings the root configuration framework.
- Add widgets through self-registration, REST services, or in batches.
- InfluxDB for event data storage

- Connect devices with MQTT, Stomp, AMQP and other protocols
- Integrates third-party integration frameworks
- Eclipse Californium for CoAP messaging
- HBase for the non-relational datastore
- Grafana to visualize SiteWhere data

☞ **DSA**

- Distributed Services Architecture (DSA) is for implementing inter-device communication, logic, and efforts at every turn of the IoT infrastructure. It allows cooperation between devices in a distributed manner and sets up a network engineer to share functionality between discrete computing systems.
- You can manage node attributes, permission, and links from DSLinks.

☞ **Thinger**

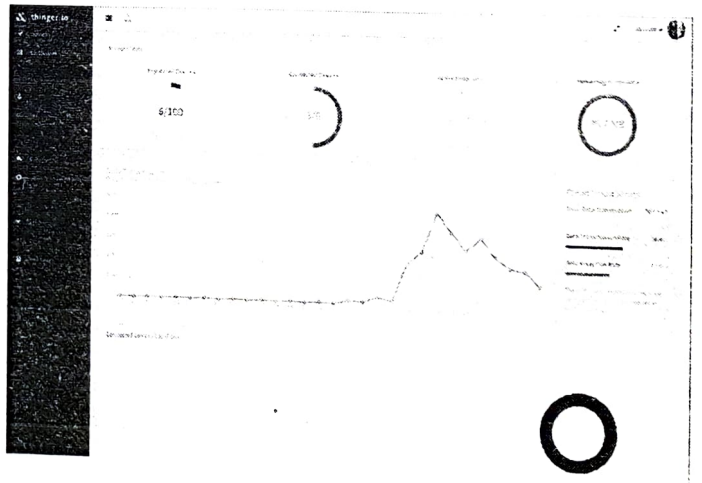


Fig. 6.4.10 Thinger

- Thinger.io provides a scalable cloud base for connecting devices. You can deal with them quickly by running the admin console or combine them into your project logic using their REST API. It supports all types of hackers boards such as Raspberry Pi, Intel Edison, ESP8266.
- Thinger can be integrated with IFTT, and it provides real-time data on a beautiful dashboard.

6.4.1 For Making front ends, REST and JSON-LD

What is REST API?

- REST stands for Representational State Transfer. It's an architectural style for developing web services. A lot of people believe that there is a REST protocol in IoT. However, REST itself is a concept, not an IoT protocol.
- REST is the basis for the most widely used form of API and is designed to be used over any protocol. However, it typically uses HTTP or COAP to work with components in a particular IoT device, such as:
  - Files
  - Objects
  - Media
- Web services are defined on the principles of REST and can be defined as a RESTful web service. RESTful web services can use normal POST, DELETE, PUT, and HTTP verbs of GET for working the components listed above.

What's the Difference between REST and RESTful?

A REST web service is a Representational State Transfer and an architectural pattern for creating web services.

On the other hand, the RESTful service is one that implements that pattern.

What's better in IoT? MQTT or REST?

Check out our guide here

☞ **Advantages of Using REST APIs**

1. **Scalability :** REST means that there's a clear separation between client and server. As a result, products can be scaled up by a development team without much difficulty.

2. **Familiarity and Usability :** REST APIs use constructs that are familiar to anyone who has used HTTP – i.e. the internet. Unless you're completely off the grid, you'll have used the internet before.

   On top of that, most IoT developers are already familiar with the REST architecture, such as SSL and TLS. This makes REST APIs the most easy-to-use API out there.

3. **Language-independent :** Developers can use any language that uses HTTP to make web-based requests. This is another reason why REST APIs are so popular with developers. They give you the power to program using a language you're comfortable and familiar with to develop your IoT app.

☞ **Disadvantages of Using REST APIs**

1. **Limited Architecture :** While the simple architecture of REST is a great entry point for budding IoT developers, those who want to do more or work with REST frequently may encounter limitations due to its architecture.

6.4.2 What is JSON-LD?

- JSON-LD stands for *JavaScript Object Notation for Linked Data*, which consists of multi-dimensional arrays (think: list of attribute-value pairs).
- It is an implementation format for structuring data analogous to Microdata and RDFa. Typically, in terms of SEO, JSON-LD is implemented leveraging the Schema.org vocabulary, a joint effort by Google, Bing, Yahoo!, and Yandex in 2011 to create a unified structured data vocabulary for the web. (However, Bing and other search engines have not officially stated their support of JSON-LD implementations of Schema.org.)
- JSON-LD is considered to be simpler to implement, due to the ability to simply paste the markup within the HTML document, versus having to wrap the markup around HTML elements (as one would do with Microdata).

What does JSON-LD do?

JSON-LD annotates elements on a page, structuring the data, which can then be used by search engines to disambiguate elements and establish facts surrounding entities, which is then associated with creating a more organized, better web overall.

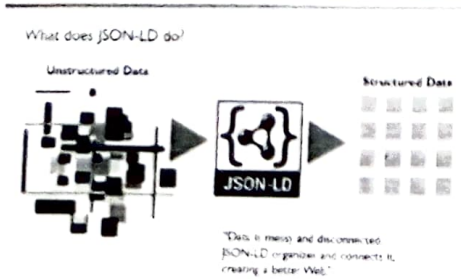


Fig. 6.4.11 : A conceptual visualization of JSON-LD taking the unstructured content on the web, annotating, and structuring the content to create an organized, structured result.

6.5 COMPARISON OF IOT BOARDS AND PLATFORMS

A comparison of IoT boards and platforms in terms of computing, IDE, Connectivity

Parameter	RASPBERRY PI	BEAGLEBONE BLACK
Model Tested	It uses Model B version.	It uses Rev A5 version.
Processor Type	It uses ARM11 processor.	It uses ARM Cortex-A8 processor.
RAM	For the functioning of raspberry pi, 512 MB SDRAM is used.	For the functioning of beaglebone black, 512 MB DDR3L is used.
Processor Speed	It uses 700 MHz for processing.	It uses 1 GHz for its processing.
Flash	It has dedicated SD Card socket for loading operating system.	It uses 4GB (micro SD) for loading OS and data storage.
Min Power	It requires a power supply of 700mA (3.5W).	It requires min power of 210mA (1.05W) for its functioning.
GPIO Pins	It has 12 GPIO pins.	It has 69 GPIO pins.
Dev IDE	It uses IDLE, Scratch, Squeak/Linux to perform tasks.	It uses Python, Scratch, Squeak, Cloud9/Linux to perform a particular task.

Parameter	RASPBERRY PI	BEAGLEBONE BLACK
USB Master	It has 2 USB 2.0 on board.	It has 1 USB 2.0 on its board.
Audio Output	Supports HDMI, Analog audio output	It uses Analog output for audio.
Video Output	It supports HDMI, Composite output for video.	No such specific video output.
UART	It uses 1 UART to transmit and receive serial data.	It uses 5 UART to transmit and receive serial data.
No. of I/O pins	It has 8 Digital, 0 Analog pins.	It has 65 Digital, 7 Analog pins.

6.6 A COMPARISON OF BOARDS AND PLATFORMS IN TERMS OF CONNECTIVITY

Platform	Connectivity	Microcontroller	Cost
Arduino Yun	WiFi & Ethernet	ATmega32u4 & AtherosAR9331	$75
Raspberry Pi	WiFin BLE & Ethernet	64 bit ARM Cortex-A53 Quad Core	$40
ESP8266	WiFi	Tensilica L106 32-bit	$3
Beaglebone Black	WiFi & BLE	OSD 3358ARM 1 GHz Cortex-AS	$70
Particle Photon	WiFi	STM32F205 120 MHz ARM Cortex M3	$20
Arduino Nano	Nil	ATmega328	$3

Arduino

- Arduino will be one of the first IoT hardware to come in mind when thinking about building a simple connected device. Arduino microcontrollers are open-source hardware which means that basically anyone can build it. There's a wide range of Arduino versions including the most popular Arduino Uno, Arduino YUN with enabled WiFi connectivity and Arduino MKR family that offers multiple wireless connectivity options such as WiFi, Bluetooth, LoRa, SigFox and Narrowband IoT.

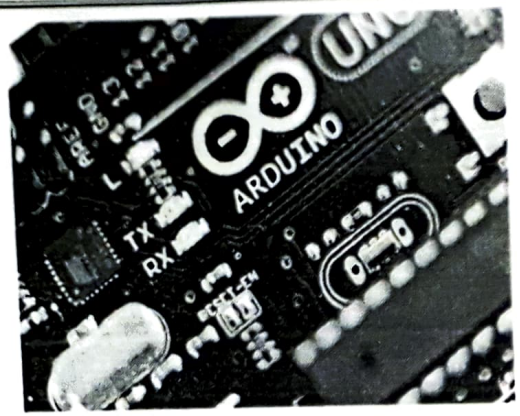


Fig. 6.6.1 : Arduino

- Apart from hardware, Arduino offers IDE (integrated development environment) and recently released Pro IDE for easier and faster coding. The platform has a well-established community, online software tools, various development kits, Arduino IoT Cloud and other resources for building connected devices.
- To learn more about Arduino hardware, check this article.

Why choose Arduino as an IoT hardware platform?

- Arduino hardware is an affordable and easy to set up option for building a basic IoT device that is supposed to perform one action, for example, read humidity sensor data.
- Arduino community is one of the oldest in this domain, so there won't be a lack of support or resources. On top of that, Arduino's functionality is easily expandable with on-top shields and multiple digital and analog general-purpose input/output pins.

☞ **Raspberry Pi**

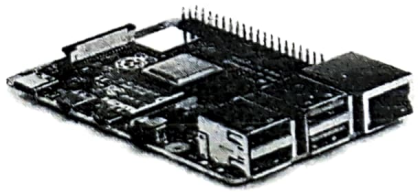


Fig. 6.6.2 : Raspberry Pi

- Raspberry Pi is another story. Pis are originally small fully-fledged computers with a range of connectivity options, a processor and up to 8GB of memory storage. It's much more powerful and speedy than other IoT boards and can handle complex functionality including data-heavy audio and video streaming.
- Just like Arduino, Raspberry Pi has its own community, accessory set, set-up and troubleshooting guides and multiple resources for developers. However, Raspberry Pi is closed-source hardware, so to build a Pi-based application, you'll need to use the boards, accessories and kits offered by the producer.

Why choose Raspberry Pi as an IoT hardware platform?

- Raspberry Pi is the best choice for data-heavy connected devices like hubs, gateways, datum collectors or personal cloud servers, however, it will also be a good fit for simpler IoT applications.
- There're several generations and various models of Pis with different componentry and price range starting from $5. Original models already have connectivity options, inputs and outputs on-board, so no on-top modules or soldering are needed for setting up basic functionality. As a rule, Pi-based solutions are low-powered, however they require more power than Arduino considering higher processing capabilities.

☞ **Particle**

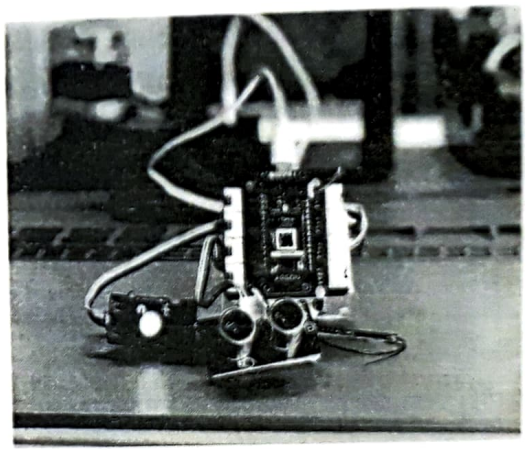


Fig. 6.6.3 : Particle application

- Particle is, probably, the most complete IoT hardware platform which offers Internet of Things hardware, connectivity, cloud and drag-and-drop IoT application builder. Apart from that, Particle has a serious developer community, its own IDE, developer tools, SDKs and numerous kits for different purposes and IoT projects.

- In terms of hardware, Particle provides boards with different types of connectivity, for example, Boron with cellular and mesh, Photon with WiFi or Xenon with mesh only. Additionally, there's a wide range of accessories, sensors and other add-ons with detailed specifications and instructions.

Why choose Particle as an IoT hardware platform?

- Particle is an all-inclusive platform that covers all bases not only for IoT prototyping but also for building a fleet of ready-to-go IoT devices. Basically, you have everything you need in one place hardware, development environment and tools, cloud and robust support from the community. Another benefit of the Particle platform is the mesh-ready hardware ,and connectivity which is getting more and more popular among IoT connectivity options.

☞ **BeagleBone**

- BeagleBone is an IoT hardware platform with open-source hardware, various daughter boards or capes to add functionality to main IoT development boards, strong Beagle community of developers and enthusiasts who promote open software and hardware in embedded computing.

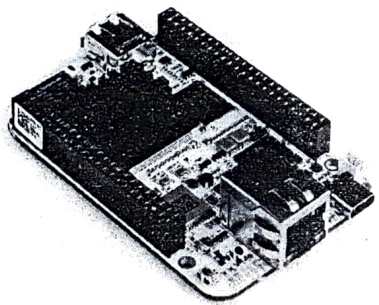


Fig. 6.6.4 : Begal bone black

- BeagleBone Linux-based boards are very diverse. They come in different sizes and feature sets, from the most basic PocketBeagle to AI-enabled BeagleBone AI with 1GB RAM and 16GB of flash memory, embedded vision engines and common connectivity options WiFi, Bluetooth and gigabit Ethernet.

Why choose BeagleBone as an IoT hardware platform?

- BeagleBone platform stands out with pretty powerful yet uncomplicated boards and great compatibility and connectivity options. For example, BeagleBone Green has 2 sets of 46 pin headers. This is a lot of I/O pins to connect to, so you can add multiple sensors and modules to your original board. Unlike Arduino or Particle, BeagleBone doesn't offer IDE but works with an open-source Cloud9 programming platform.

Let's discuss your project.

☞ **Adafruit**

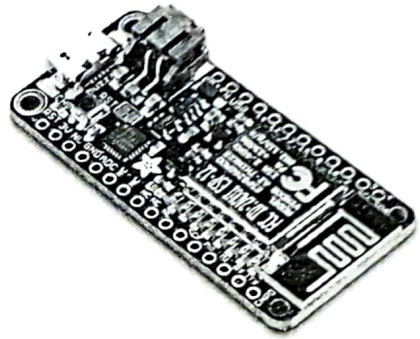


Fig. 6.6.5 : Adafruit

- Adafruit is a hardware platform and marketplace that has a huge community and can become the best place for the newbies in electronics and embedded computing.
- The platform both provides its own hardware and accessories and sells boards by other vendors like Raspberry Pi and Arduino.
- Adafruit Feather boards and extensions (wings) are extremely flexible extensions can work with any board. To connect devices to the Internet and handle all the data they create, Adafruit offers Adafruit IO cloud service that works both with Adafruit and Arduino hardware.

Why choose Adafruit as an IoT hardware platform?

- Adafruit may not have its own IDE or IoT software platform, but has one of the strongest communities, support and knowledgebase for building an IoT project and connecting physical objects to the Internet. Adafruit's hardware also has a competitive advantage. Feather boards are extremely light and simple, so they will become a great start of a small and uncomplicated IoT device like a soil sensor or a tracking wearable.
- *Read: What's hot on Internet of Things wearable technology market?*

☞ **Espressif**

- Espressif may not be the first in the IoT platform list, but can be the first option for an industrial IoT development. The thing is, one of the most catchy features of Espressif's hardware is longevity and robustness, which is a great perk for IoT devices that need to endure extreme conditions or be placed in remote locations. The most popular board series - ESP8266 and ESP32 have 12 years longevity guarantee, for example.

- Apart from reliable hardware and versatile development kits, Espressif IoT platform offers an IoT software development ecosystem, developer space for support and communication and multiple tools and apps for building an IoT prototype.

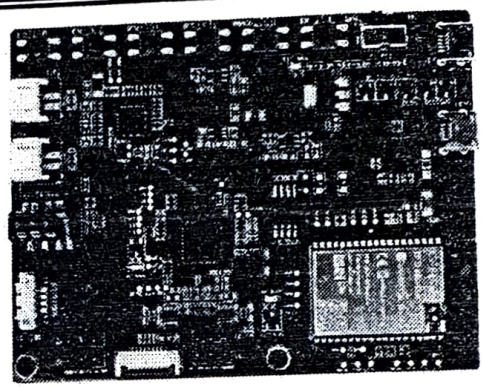


Fig. 6.6.6 Espressif

Why choose Espressif as an IoT hardware platform?

- Espressif offers diverse hardware options from coin-sized chips to full development kits for rapid prototyping and easy setup. Adafruit, for example, uses ESP microcontroller in its Feather development boards.
- As mentioned earlier, ESP boards are designed to withstand extreme conditions and can address the needs of a certain type of IoT projects. In terms of IoT connectivity, platform hardware works with WiFi, Bluetooth and mesh networks.

Chapter Ends...

❑❑❑