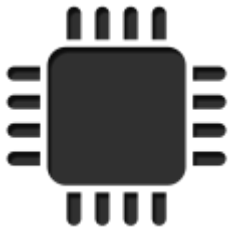
Lecture 3

Smart Devices

IoT Technology Stack

Device Hardware

- Physical objects such as sensors, modules, and gateways

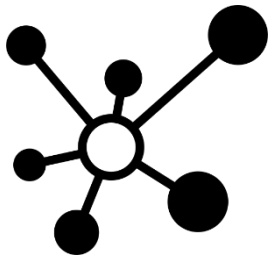


Device Software

- Runs on the device's processor and controls its functions

Connectivity

- How devices connect to the internet and transfer data

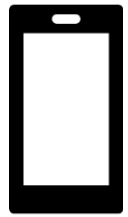


Platform

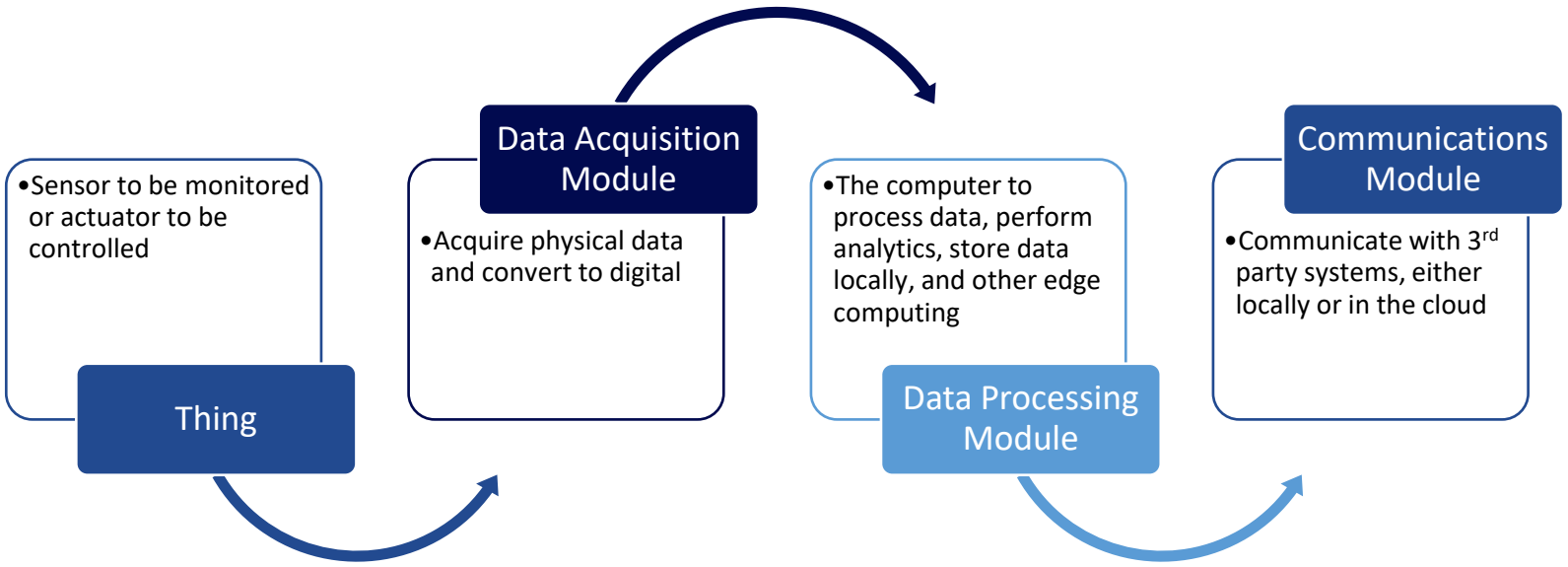
- Where data from IoT devices is captured, processed and stored

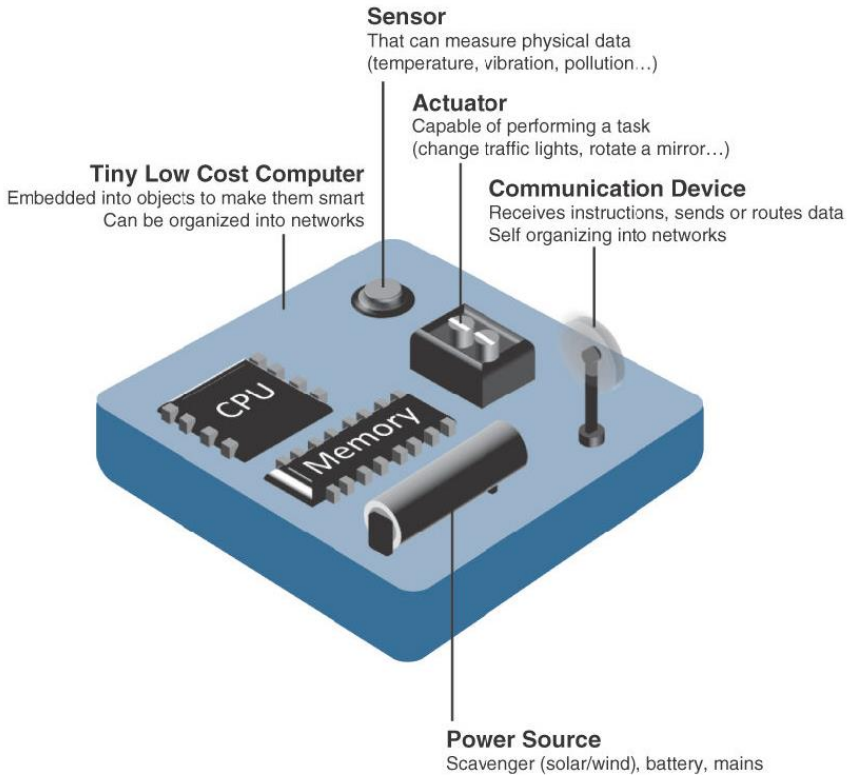
Application

- Dashboards and data visualization, data analytics, alerts, AL/ML

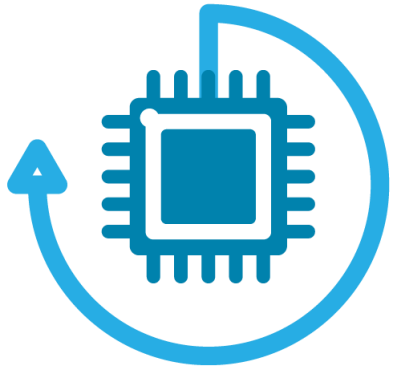


Smart Device





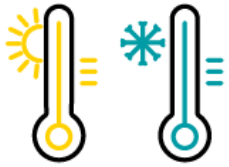
IoT Sensors



Additional Requirements for IoT sensors

- **Low cost**
  To make IoT devices more economical for use in market
- **Small formfactor**
  For easy mounting in any environment
- **Wireless**
  For easy installation
- **Self-identification and self-validation**
  To generate alarm for it's own failure
- **Low power**
  For long lasting battery operation or manage with energy harvesting
- **Robust**
  To minimize or eliminate maintenance
- **Self-diagnostic and self-healing**
  To detect own health
- **Self-calibrating**
  Or accepts calibration commands via wireless link, for accurate results
- **Data pre-processing**
  To reduce load on gateways, PLCs and cloud resources

IoT Sensors



Temperature Sensor
- Thermistor
- Resistance temperature detectors
- Thermocouples

Water Level Sensor
- Hydrostatic pressure
- Optical sensor

Moisture Sensor
- Hair tension
- Psychrometer

Presence & Proximity Sensor
- Doppler radar
- Infrared light

Chemical Sensor
- Electrochemical breathalyzer
- Electronic nose

Light Sensor
- Photoresistor
- Photodiode

Motion Sensor
- Active, ultrasonic
- Passive, infrared
- Active, radar

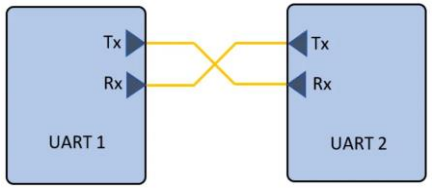
Image Sensor
- Active-pixel
- Charge-coupled device

Acoustic & Noise Sensor
- Hydrophone
- Geophone

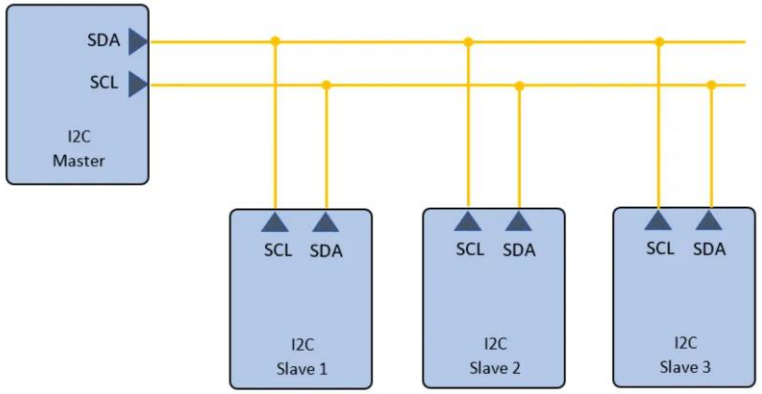
Gyroscope Sensor
- Accelerometer
- Heading indicator

IoT Sensor Interfaces

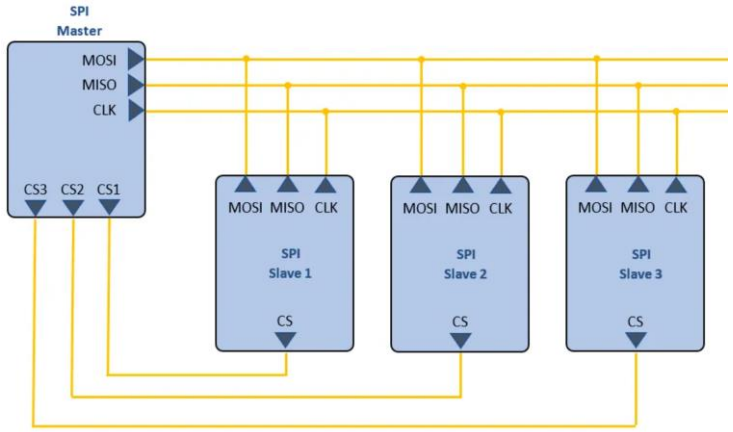
UART
Universal Asynchronous Receiver/ Transmitter



I2C
Inter-Integrated Circuit

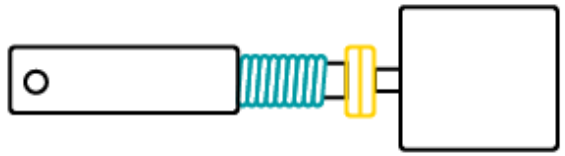


SPI
Serial Peripheral Interface



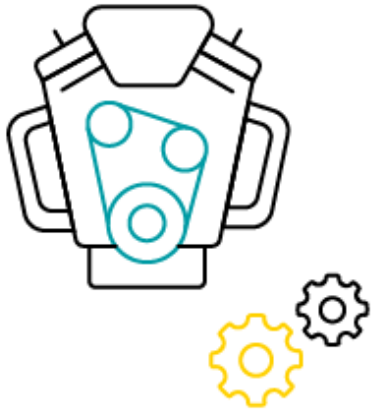
	UART	I2C	SPI
Number of Pins	2	2	>3
Baud Rate (b/s)	Up to 115,200	Up to 400,000	Up to a few megabits
Communication type	Point to Point	Multi Master - Multi Slave	One Master - Multi Slave
Half and Full duplex	Full duplex	Half Duplex	Full Duplex
Synchronous or Asynchronous	Asynchronous	Synchronous	Synchronous
Maximum Number of Devices on the Bus	2	Up to 128	Theoretically Infinite (Limited by the Number of I/O Pins of the Master)
Complexity	Low	High	Medium
Cost	Low	Medium	High

IoT Actuators



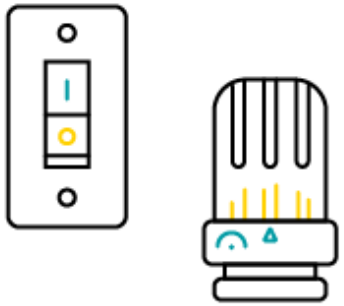
Linear actuators

Used to enable motion of objects or elements in a straight line.



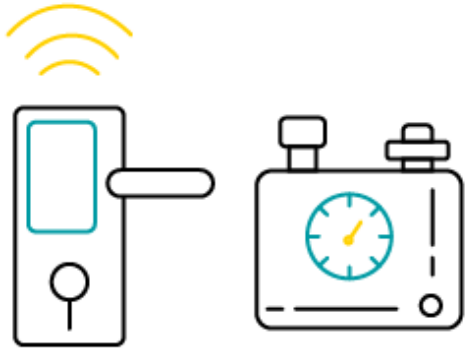
Motors

Enable precise rotational movements of device components or whole objects.



Relays

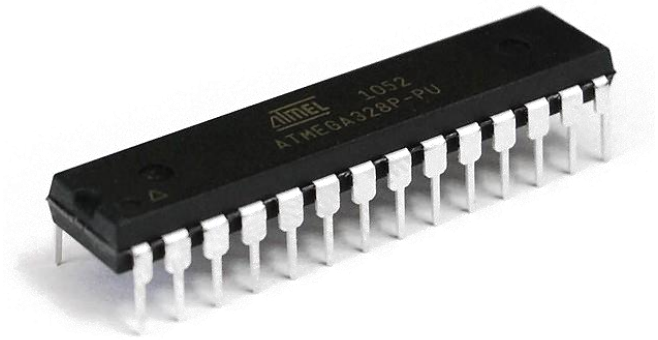
Electromagnet-based actuators to operate power switches in lamps, heaters or even smart vehicles.

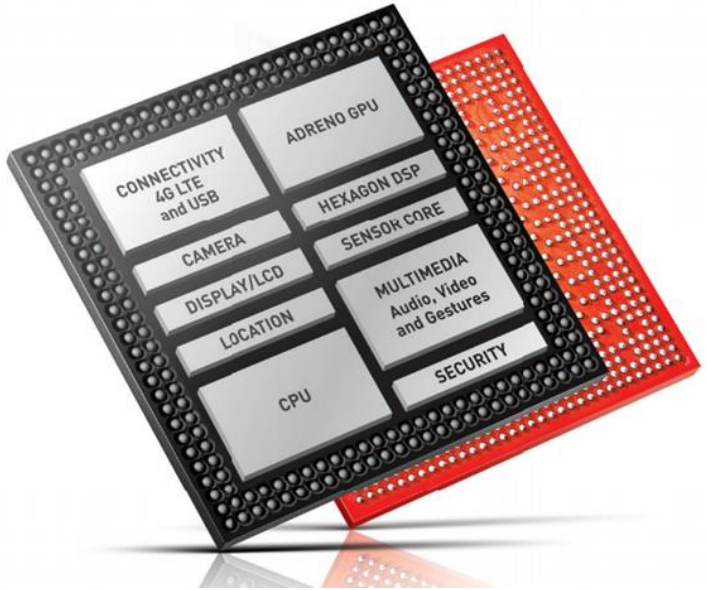


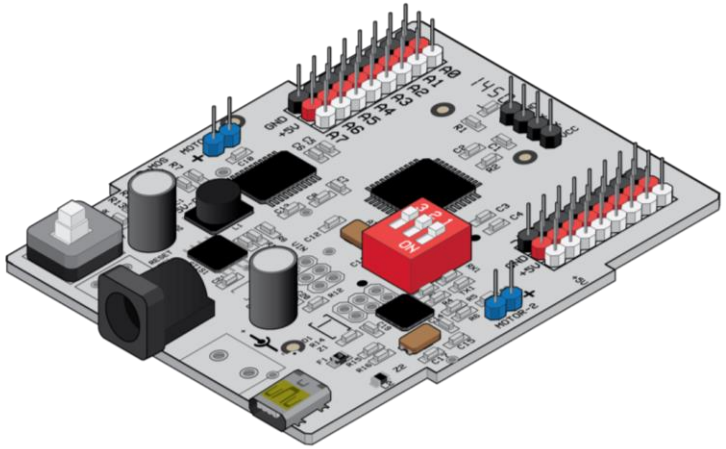
Solenoids

Most widely used in home appliances as part of locking or triggering mechanisms, they also act as controllers in IoT-based gas and water leak monitoring systems.

IoT Hardware Platforms







Microcontrollers

Small self-contained and low-cost computers embedded in a single IC chip that contains one or multiple CPUs, memory elements, and programmable peripherals.

System on Chip

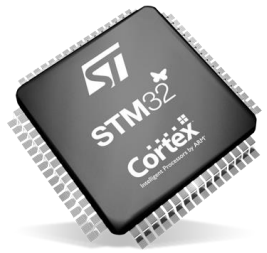
An entire computer including the processing unit, internal memory, navigation system, wireless connection, graphic cards, and analog I/O ports integrated into a single chip that can perform a variety of tasks from signal processing to artificial intelligence.

Single-board Computers

All devices of an entire embedded computing system including the microprocessors, I/O peripherals and memory elements are integrated and built on a single PCB.

Microcontroller

arm



High Performance	STM32F2 398 CoreMark 120 MHz Cortex-M3	STM32F4 608 CoreMark 180 MHz Cortex-M4	STM32F7 1082 CoreMark 216 MHz Cortex-M7	STM32H7 Up to 3224 CoreMark Up to 550 MHz Cortex-M7 240 MHz Cortex-M4
Mainstream	STM32G0 142 CoreMark 64 MHz Cortex-M0+	STM32G4 ● 569 CoreMark 170 MHz Cortex-M4		
	STM32F0 106 CoreMark 48 MHz Cortex-M0	STM32F1 177 CoreMark 72 MHz Cortex-M3	STM32F3 ● 245 CoreMark 72 MHz Cortex-M4	● Optimized for mixed-signal applications
Ultra-low-power	STM32L4+ 409 CoreMark 120 MHz Cortex-M4	STM32U5 651 CoreMark 160 MHz Cortex-M33		
	STM32L0 75 CoreMark 32 MHz Cortex-M0+	STM32L1 93 CoreMark 32 MHz Cortex-M3	STM32L4 273 CoreMark 80 MHz Cortex-M4	STM32L5 443 CoreMark 110 MHz Cortex-M33
Wireless	STM32WL 162 CoreMark 48 MHz Cortex-M4 48 MHz Cortex-M0+	STM32WB ● 216 CoreMark 64 MHz Cortex-M4 32 MHz Cortex-M0+		● Cortex-M0+ Radio co-processor

Cortex - A

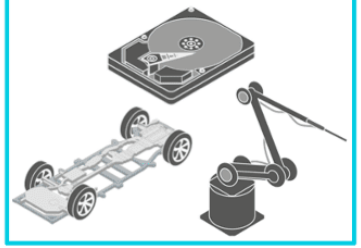
Highest performance

Optimised for rich operating systems

Cortex - R

Fast response

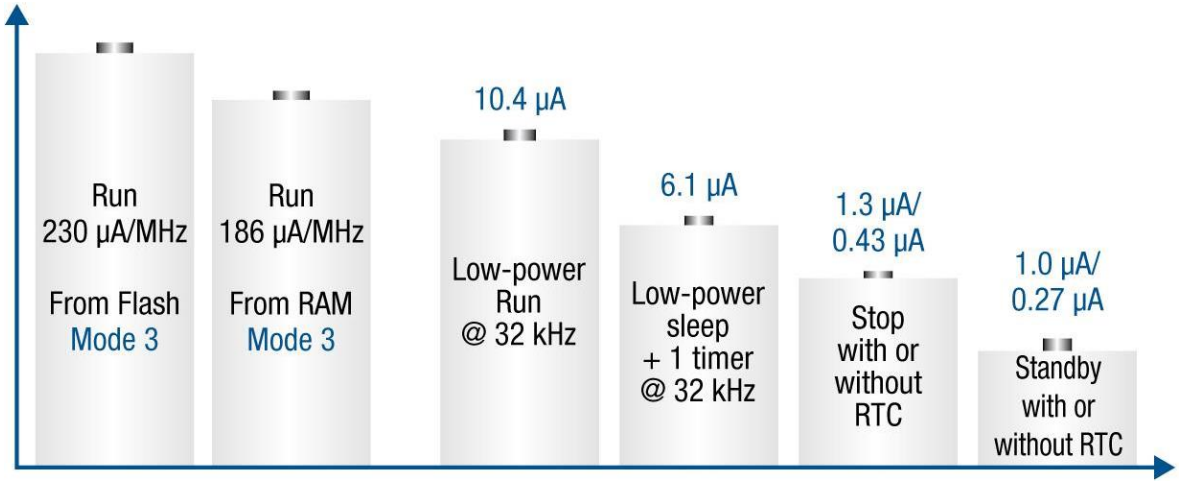
Optimised for high performance, hard real-time applications



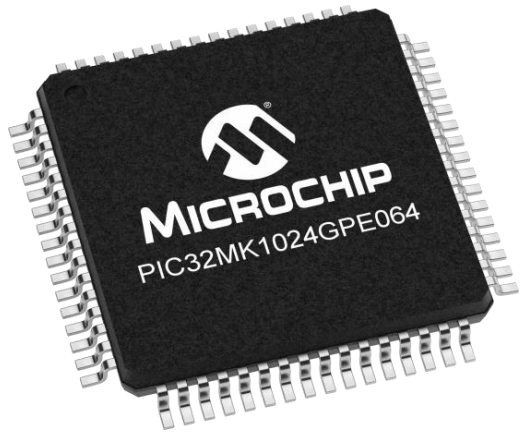
Cortex - M

Smallest/lowest power

Optimised for discrete processing and microcontrollers



Microcontroller



PERFORMANCE

	Memory Flash/SRAM	Automotive	Connectivity	Functional Safety	Graphics	Motor Control	Security	Ultra-Low Power	Touch
MIPS32® M-Class, 252 MHz	512-2048 KB/ 128-512 KB	●	●	●	●		●		●
MIPS32 microAptiv™, 200 MHz	1024-2048 KB/ 256-640 KB		●	●	●		●		●
Arm® Cortex®-M4F, 120 MHz	1024 KB/ 256 KB	●	●	●	●	●	●		●
MIPS32 microAptiv, 120 MHz	256-1024 KB/ 128-256 KB	●	●	●	●	●			●
MIPS32 M4K®, 80-120 MHz	32-512 KB/ 8-128 KB		●	●					
MIPS32 M4K, 80 MHz	64-512 KB/ 16-128 KB		●	●					
MIPS32 M4K, 72 MHz	128-256 KB/ 32-64 KB		●	●				●	
MIPS32 M4K, 50 MHz	16-512 KB/ 4-64 KB		●	●					
Arm® Cortex®-M0+, 48 MHz	128-512 KB/ 16-64 KB	●	●	●		●	●		●
Arm® Cortex®-M23, 48 MHz	256-512 KB/ 32-16 KB		●				●	●	●
Arm® Cortex®-M0+, 48 MHz	64-128 KB/ 8-16 KB			●		●			
MIPS32 microAptiv UC, 25 MHz	16-256 KB/ 4-32 KB	●						●	

Single-Board Computers

Built with the Particle Platform

Particle offers a range of development kits designed to connect to the Internet over Wi-Fi, cellular, or BLE. Particle is the best platform to build a connected project from prototype to production.

ESPRESSIF

Develops Wi-Fi and Bluetooth low-power IoT hardware solutions. Most well-known for ESP8266 and ESP32 chips, modules, and development boards.

adafruit

Adafruit Feather is a line of open-source development boards that are designed for prototyping on the fly. The Adafruit Feather line comes with a large suite of accessories that rapidly accelerate development.

ARDUINO

Is the ubiquitous name in the electronic development space. The company offers a range of open-source development kits, microcontrollers, and software tools for building connected products.

sparkfun ELECTRONICS

Is a large retail store that sells everything from development kits, breakout boards to sensors. They offer more than 2,000+ open source components and offer their own training and online tutorials that explain how to build embedded electronics.

Raspberry Pi

Is a single-board-based computer that runs on Linux and is designed for prototyping small computing applications. Raspberry Pi products are perfect for people of all ages and are a good way to get into electronic development

.

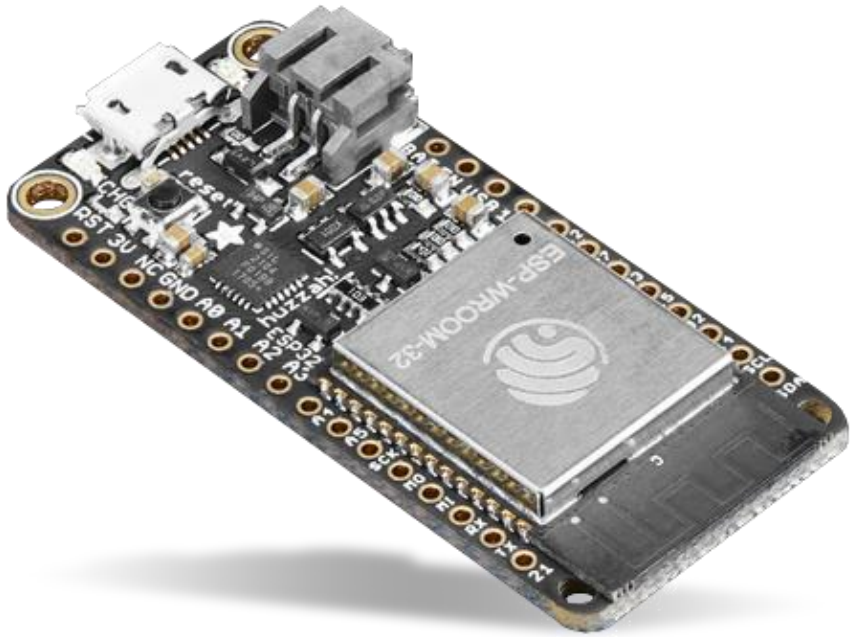
Single-Board Computers

Built with the Particle Platform

Particle Gen 3 Hardware
The 3rd Generation of Particle products enables users to connect projects to the Internet over Wi-Fi, Cellular, or BLE. Models are based on the Nordic nRF52840 and have built-in battery charging circuitry so it's easy to connect a Li-Po and deploy your local network in minutes.

- Photon
- Argon
- Boron

Single-Board Computers



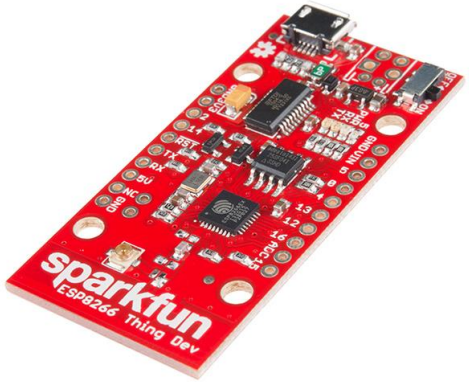
Adafruit Feather Huzzah32
This feather development board is packed with everything you need to rapidly prototype a connected project: USB-to-Serial converter, USB-to-Serial converter, automatic bootloader reset, Lithium Ion/Polymer charger, and a dual-core ESP32 chip, which means it has both WiFi and Bluetooth Classic/LE support.

Adafruit Feather 32u4 Bluefruit LE
A dev kit designed around BLE

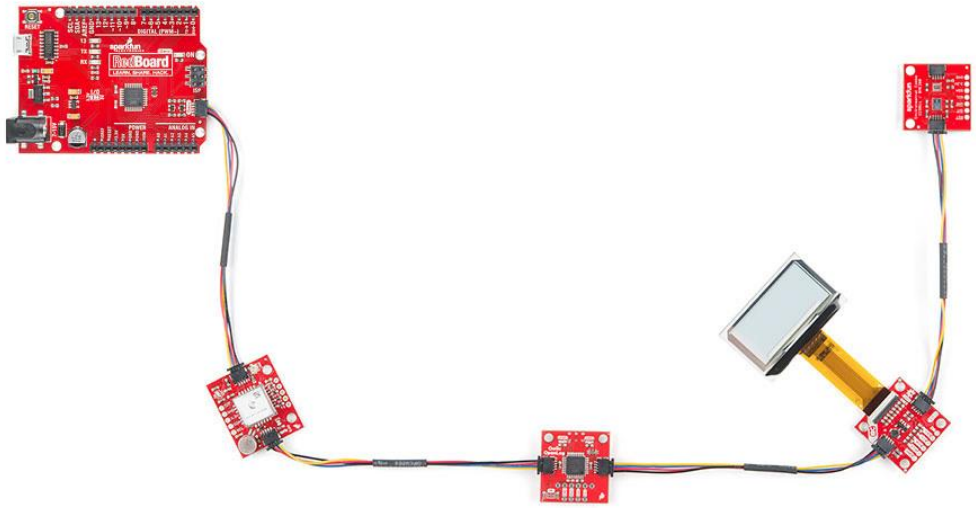
Adafruit Feather 32u4 Basic Proto
Feather development board designed around power.

Single-Board Computers

SparkFun ESP8266 Thing - Dev Board
A development board that has been solely designed around the ESP8266, with an integrated FTDI USB-to-Serial chip. The ESP8266 is a cost-effective, and very capable WiFi-enabled microcontroller.



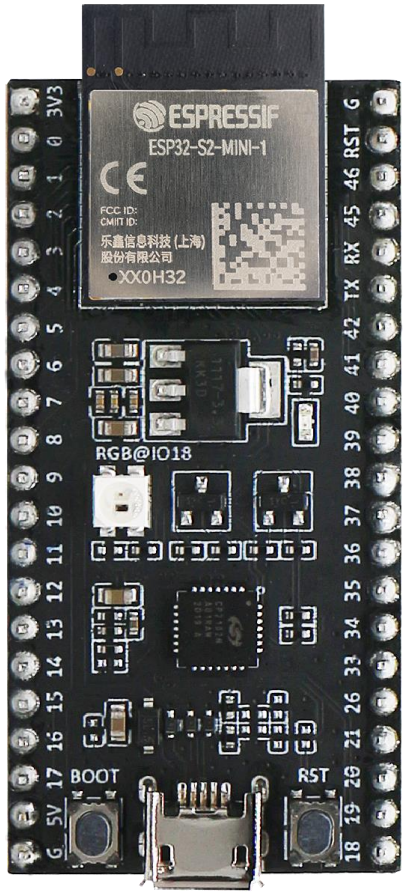
Qwicc Connect System
uses 4-pin JST connectors to quickly interface development boards with sensors, LCDs, relays and more.



Single-Board Computers

ESPRESSIF

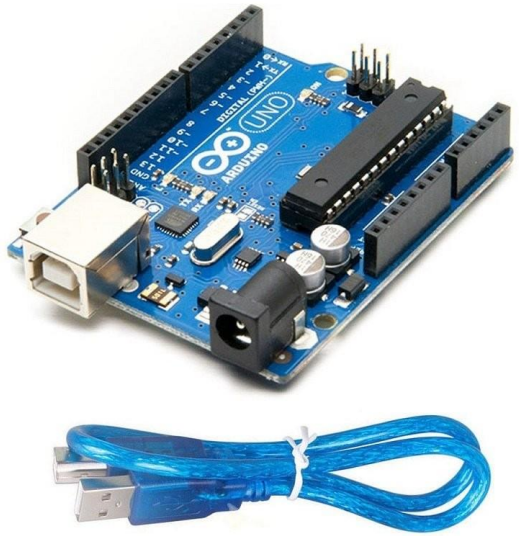
2.4 GHz Wi-Fi & BT/BLE Development Boards
These boards provide PC connectivity, 5V/GND header pins, or 3V3/GND header pins ESP-IDF source code and example applications. These boards support everything from image transmission, voice recognition and come with a variety of possible features, such as onboard LCD, JTAG, camera header, RGB LEDs, etc.

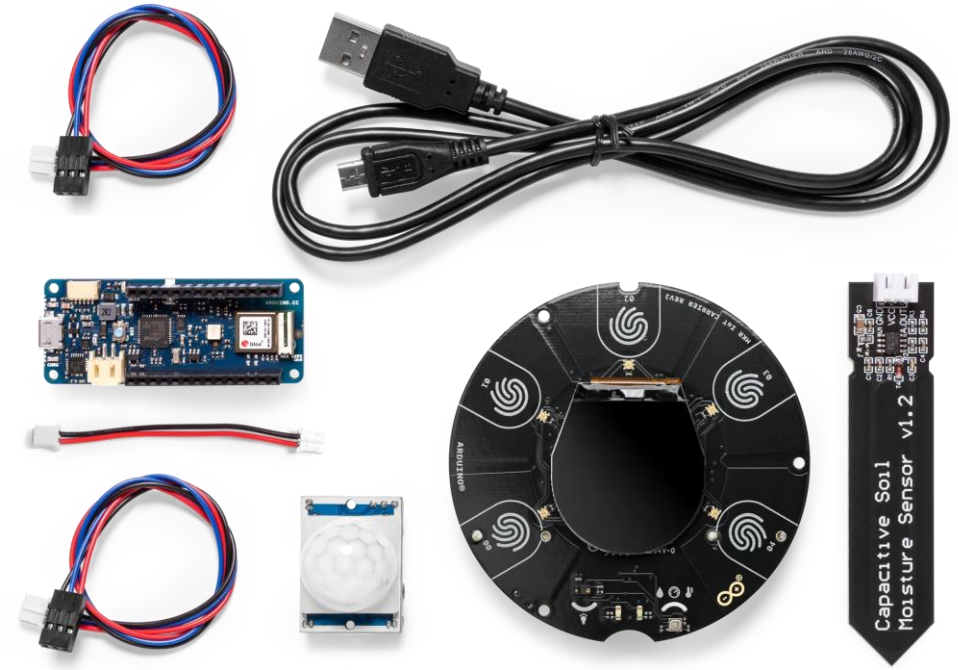


Single-Board Computers

Explore IoT Kit Rev2

Create, collaborate, impact: teach students how to use IoT technologies and design thinking to solve real-world challenges.





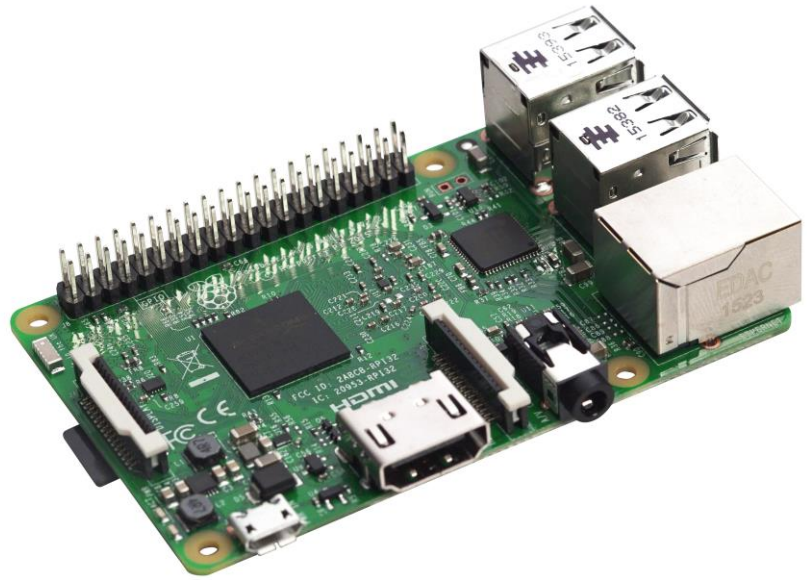
Single-Board Computers

Raspberry Pi

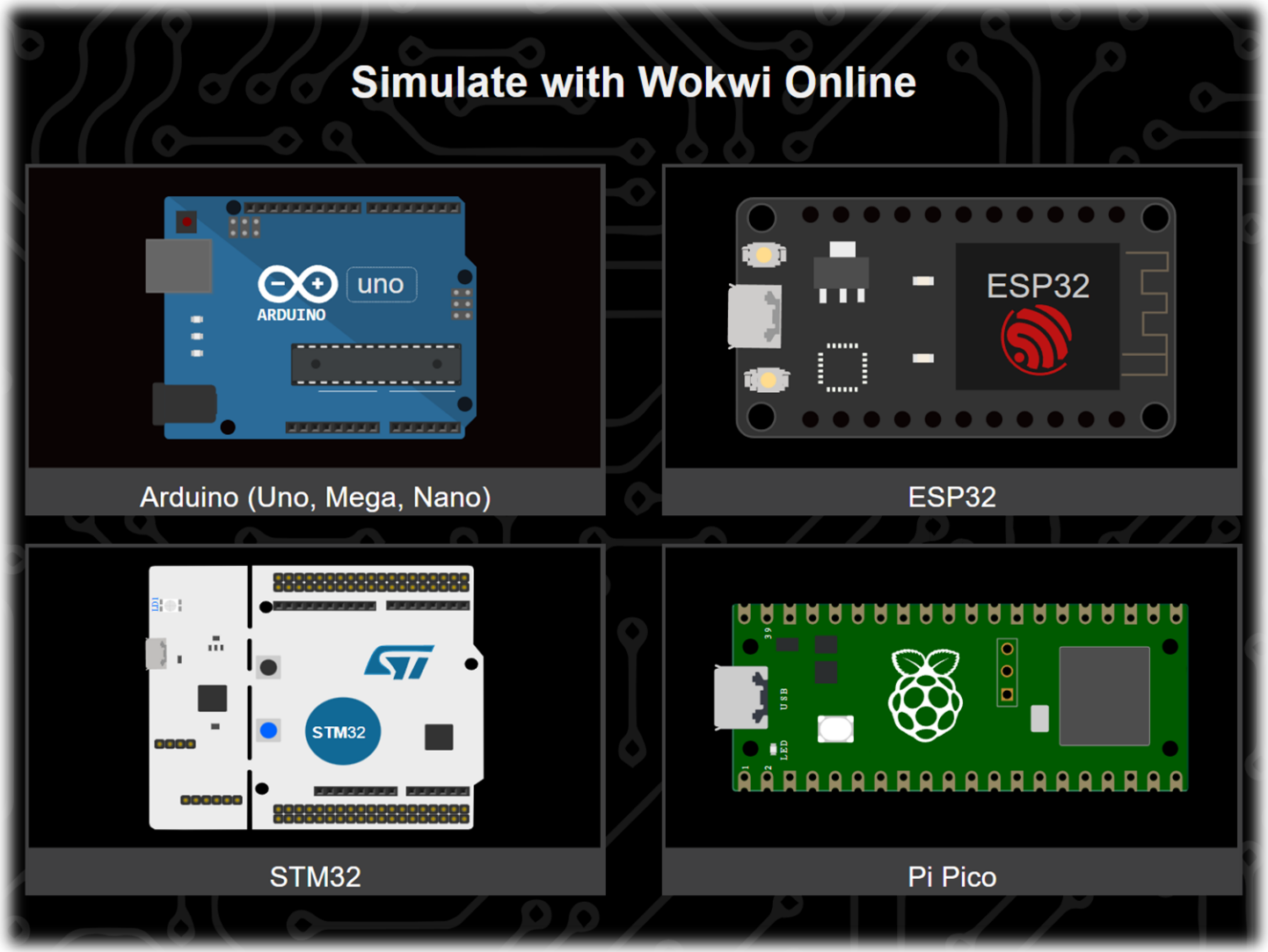
Raspberry Pi 3 Model B+
The Raspberry Pi 3 Model B+ is the newest product in the 3 range that comes with a 1.4GHz 64-bit quad-core processor, dual-band wireless LAN, Bluetooth 4.2/BLE, faster Ethernet, and Power-over-Ethernet support (with separate PoE HAT).

Raspberry Pi 1 Model B+
The Model B+ is the final revision of the original Raspberry Pi that comes with more USB ports, more GPIO pins, Micro SD, better audio, and a neater form factor.

Compute Model 4
The Compute Model 4 is a Raspberry Pi designed for deeply embedded applications. This board is great for those who intend to make a serious enterprise application.



Online Simulators



wokwi.com

System-on-Chip

NORDIC®
SEMICONDUCTOR

nRF52805 SoC

Baseline nRF52 Series multiprotocol SoC supporting Bluetooth Low Energy

64 MHz Cortex-M4
192 KB Flash, 24 KB RAM
2.4 GHz transceiver
2 Mbps, 1 Mbps
Bluetooth Low Energy
+4 dBm TX power
128-bit AES
UART, SPI, TWI
12-bit ADC

nRF52810 SoC

Baseline nRF52 Series multiprotocol SoC optimized for wide market appeal and cost-constrained applications.

64 MHz Cortex-M4
192 KB Flash, 24 KB RAM
2.4 GHz Transceiver
2 Mbps, 1 Mbps
Bluetooth 5
ANT
+4 dBm TX Power
128-bit AES CCM
UART, SPI, TWI, PDM
PWM
12-bit ADC

nRF52811 SoC

Baseline nRF52 Series SoC with comprehensive protocol support, including Bluetooth 5.1 Direction Finding.

64 MHz Cortex-M4
192 KB Flash, 24 KB RAM
2.4 GHz Transceiver
2 Mbps, 1 Mbps, Long Range
Bluetooth 5.1 Direction Finding
ANT, 802.15.4, Thread, Zigbee
+4 dBm TX Power
128-bit AES CCM
UART, SPI, TWI, PDM
PWM
12-bit ADC

nRF52820 SoC

Baseline nRF52 SoC supporting Bluetooth mesh, Thread and Zigbee, qualified for up to 105°C ambient temperature with built-in USB

64 MHz Arm Cortex-M4
256 KB Flash, 32 KB RAM
2 Mbps, 1 Mbps, Long Range
Bluetooth Low Energy
Bluetooth Direction Finding
Bluetooth mesh
Thread, Zigbee
+8 dBm TX power
128-bit AES
UART, SPI, TWI, QDEC
Full Speed USB 2.0
-40 to 105 °C extended temperature range
1.7 to 5.5 V supply voltage

nRF52832 SoC

General-purpose nRF52 Series multiprotocol SoC with high performance and broad peripheral set.

64 MHz Cortex-M4F
512/256 KB Flash, 64/32 KB RAM
2.4 GHz Transceiver
2 Mbps, 1 Mbps
Bluetooth 5, Bluetooth mesh
ANT
+4 dBm TX Power
128-bit AES CCM
UART, SPI, TWI, PDM, I2S
PWM
12-bit ADC
NFC-A

nRF52833 SoC

Bluetooth 5.1 SoC supporting Bluetooth mesh, Thread and Zigbee, qualified for up to 105°C ambient temperature

64 MHz Arm Cortex-M4 with FPU
512 KB Flash, 128 KB RAM
2 Mbps, 1 Mbps, Long Range
Bluetooth 5.1 Direction Finding
Bluetooth mesh
Thread, Zigbee
+8 dBm TX Power
128-bit AES CCM
UART, SPI, TWI, PDM
HS-SPI, I2S, PWM
12-bit ADC
NFC
USB 2.0

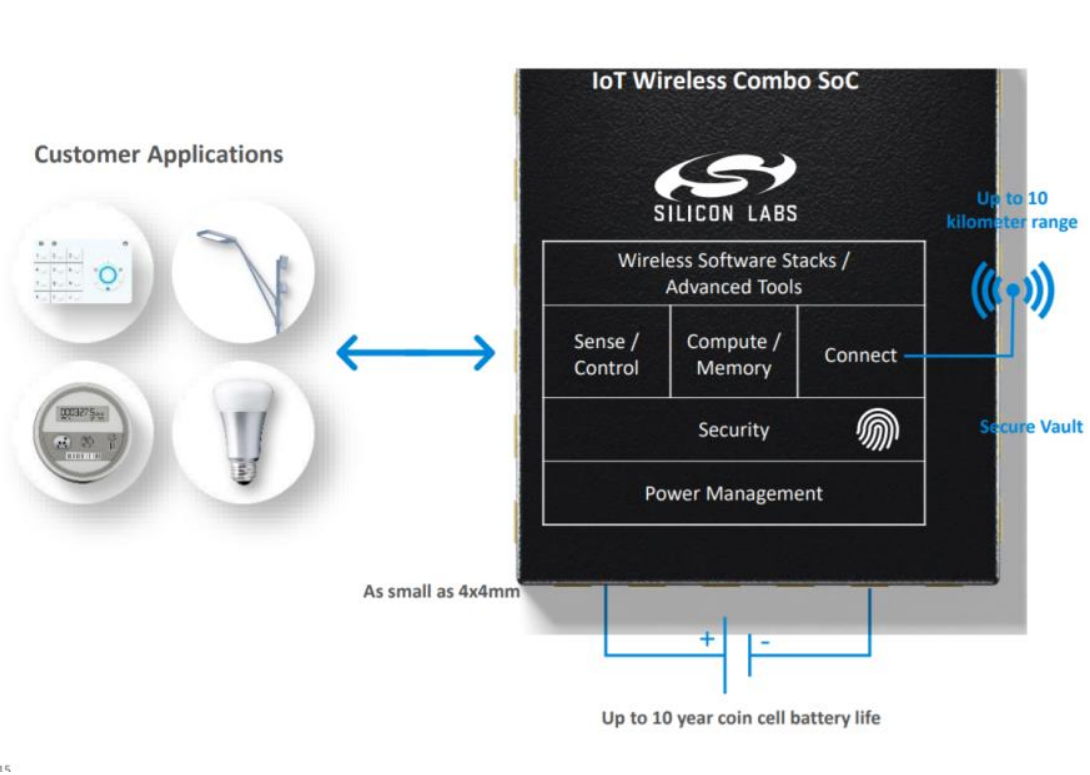
nRF52840 SoC

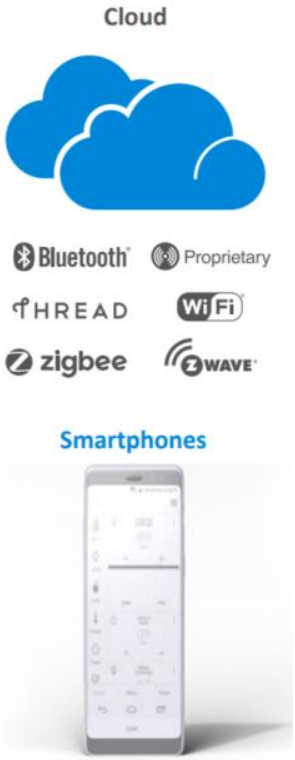
High-end nRF52 Series multiprotocol SoC for high-performance, feature-rich applications with best-in-class security.

64 MHz Cortex-M4F
1 MB Flash, 256 KB RAM
2.4 GHz Transceiver
2 Mbps, 1 Mbps, Long Range
Bluetooth 5, Bluetooth mesh
ANT, 802.15.4, Thread, Zigbee
+8 dBm TX Power
128-bit AES CCM, ARM CryptoCell
UART, SPI, TWI, PDM, I2S, QSPI
PWM
12-bit ADC
NFC-A
USB 2.0

System-on-Chip

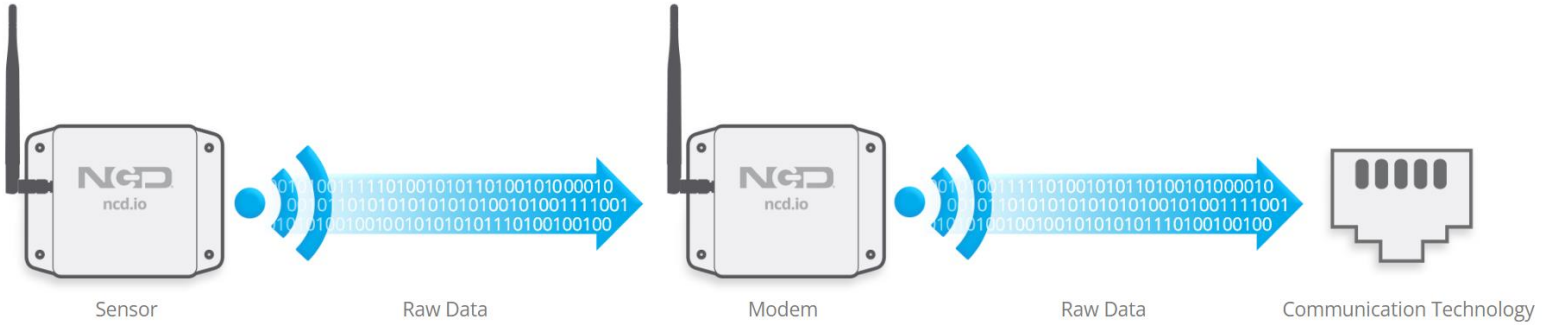
SILICON LABS





IoT Modems vs. Gateways

Modems receive wireless data from remote sensors and forward these data to a different communications format, such as Bluetooth, Wi-Fi, or other popular communication technologies.



Gateways receive wireless data from remote sensors, translate these data into meaningful real-world values, and communicate these data to popular cloud platforms. They require frequent firmware updates to include new translator profiles.

Benefits of IoT Gateways

Device-to-cloud and inter-device communications
Directs communication from several devices to the cloud

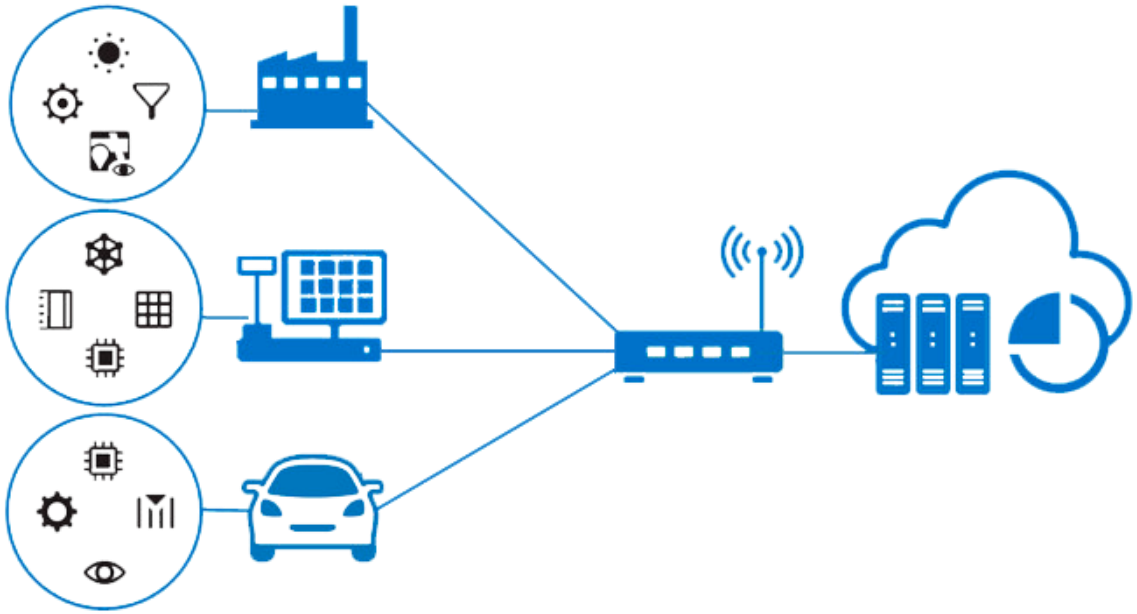
- Devices from different vendors communicate amongst themselves.

Edge computing
Filters and pre-processes data

- Saves bandwidth, reduces latency and connectivity costs, saves energy

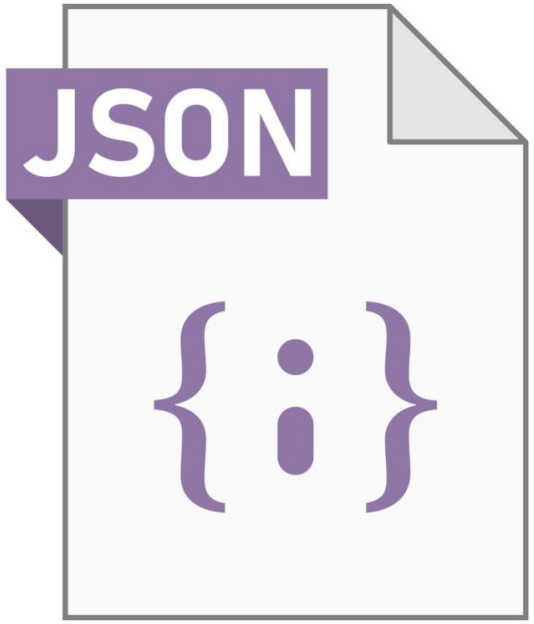
Security
Offers integrated security functionality

- Reduces the connection points and implements full data encryption to protect IoT devices from the threats of the public Internet.



IoT Data Format

JSON or **JavaScript Object Notation** is a lightweight text-based open standard designed for human-readable data interchange. It is used to transmit structured data over network connections.



- Data is in name/value pairs
- Data is separated by commas
- Curly braces hold objects
- Square brackets hold arrays

```
{ "name": value}
```

JSON Values

JSON Strings:

```
{"name":"John"}
```

JSON Numbers:

```
{"age":30}
```

JSON Objects:

```
{
"employee":{"name":"John", "age":30, "city":"New York"}
}
```

JSON Arrays:

```
{
"employees":["John", "Anna", "Peter"]
}
```

JSON Booleans:

```
{"sale":true}
```

JSON Null:

```
{"middlename":null}
```

IoT Data Format

JSON string:

```
'{"name":"John", "age":30, "car":null}'
```

JSON object literal:

```
{"name":"John", "age":30, "car":null}
```

JavaScript Object:

```
myObj = {"name":"John", "age":30, "car":null};
```

```
const myJSON = '{"name":"John", "age":30, "car":null}';
const myObj = JSON.parse(myJSON);
x = myObj.name;
```

JavaScript Object ↔ JSON

JSON String → JavaScript Object: JSON.parse()

```
jsontext = '{"name":"John", "age":30, "city":"New York"}'
obj = JSON.parse(jsontext);
console.log(obj)
console.log(obj.name)
```

JavaScript Object → JSON String: JSON.stringify()

```
Temp=23.8;
Speed=70;
obj={"Name":"John","Temperature":Temp,"Odometer":Speed};
console.log(obj);
console.log(obj.Temperature);
json_string=JSON.stringify(obj);
console.log(json_string);
```

Online JavaScript Compiler: **playcode.io/javascript**

Python Dictionary ↔ JSON

Python Dictionary → JSON String: json.dumps()

```
import json

Temp = 28.3
Speed = 80
Car_info = {
   "Driver": "John",
   "Temperature": Temp,
   "Odometer": Speed
}

json_message = json.dumps(Car_info)
print(Car_info["Temperature"])
print("Here is the JSON message " + json_message)
```

JSON String → Python Dictionary: json.loads()

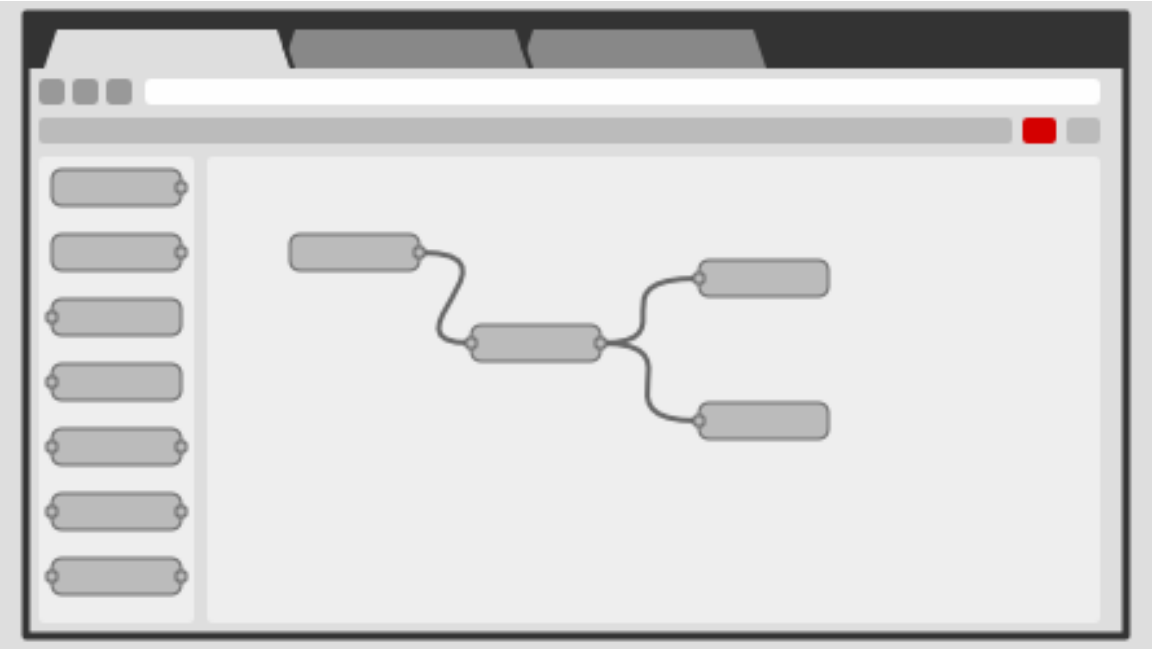
```
import json

json_message = '{"Driver":"John","Temperature": 28.3}';
print(json_message)
Car_info = json.loads(json_message)
print(Car_info["Temperature"])
```

Online Python Compiler: **www.online-python.com**

Node-RED

- Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.

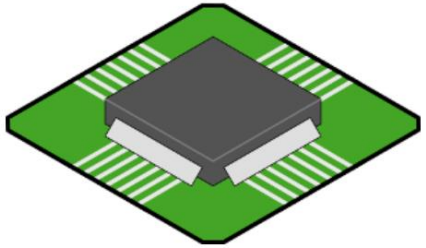
- Node-Red allows logical flow-building, so reactions and events may be generated based on actual sensor values.



Node-RED

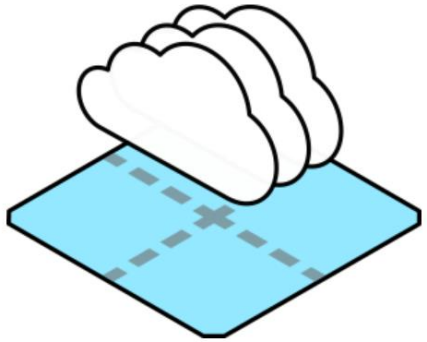
- It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.
- Node-RED is built on Node.js, taking full advantage of its event-driven, non-blocking model. This makes it ideal to run at the edge of the network on low-cost hardware such as the Raspberry Pi as well as in the cloud.

Run locally



On a device

- Raspberry Pi
- BeagleBone Black
- Interacting with Arduino
- Android



In the cloud

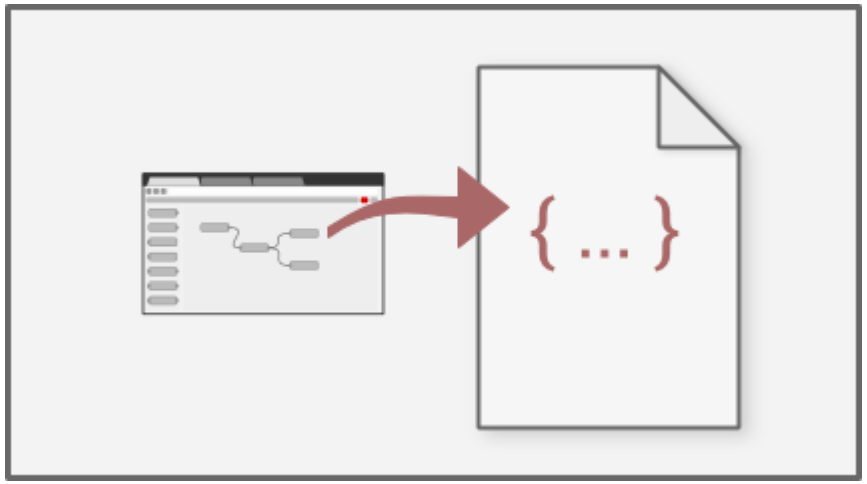
- IBM Cloud
- SenseTecnic FRED
- Amazon Web Services
- Microsoft Azure

Node-RED

Social development

- The flows created in Node-RED are stored using JSON which can be easily imported and exported for sharing with others.

- An online flow library allows you to share your best flows with the world.



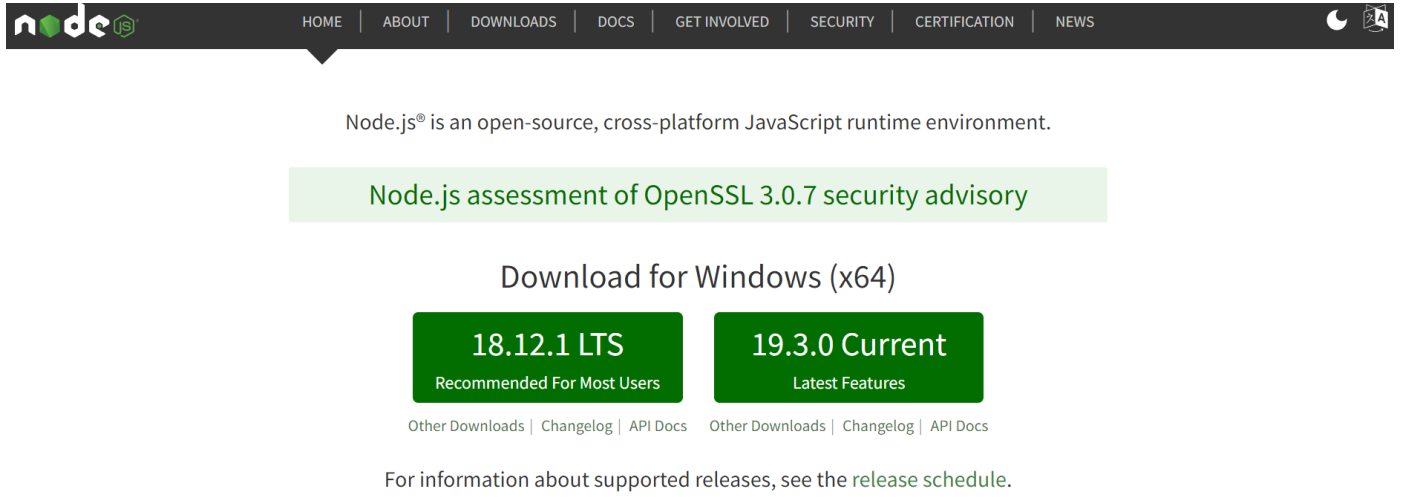
28/01/2018, 12:14:41 node: e9bfcf86.03984

msg.payload : Object

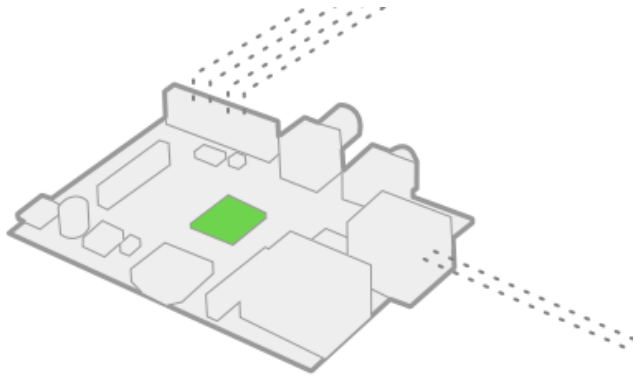

```
▾object
  FirstName: "Fred"
  Surname: "Smith"
  Age: 28
▸ Address: object
▾ Phone: array[4]
  ▸ 0: object
  ▸ 1: object
  ▾ 2: object
      type: "office"
      number: "01962 001235"
  ▸ 3: object
```

Installing Node-RED

1. Download and install the recommended version of the JavaScript runtime Node.js: https://nodejs.org/en/#home-downloadhead



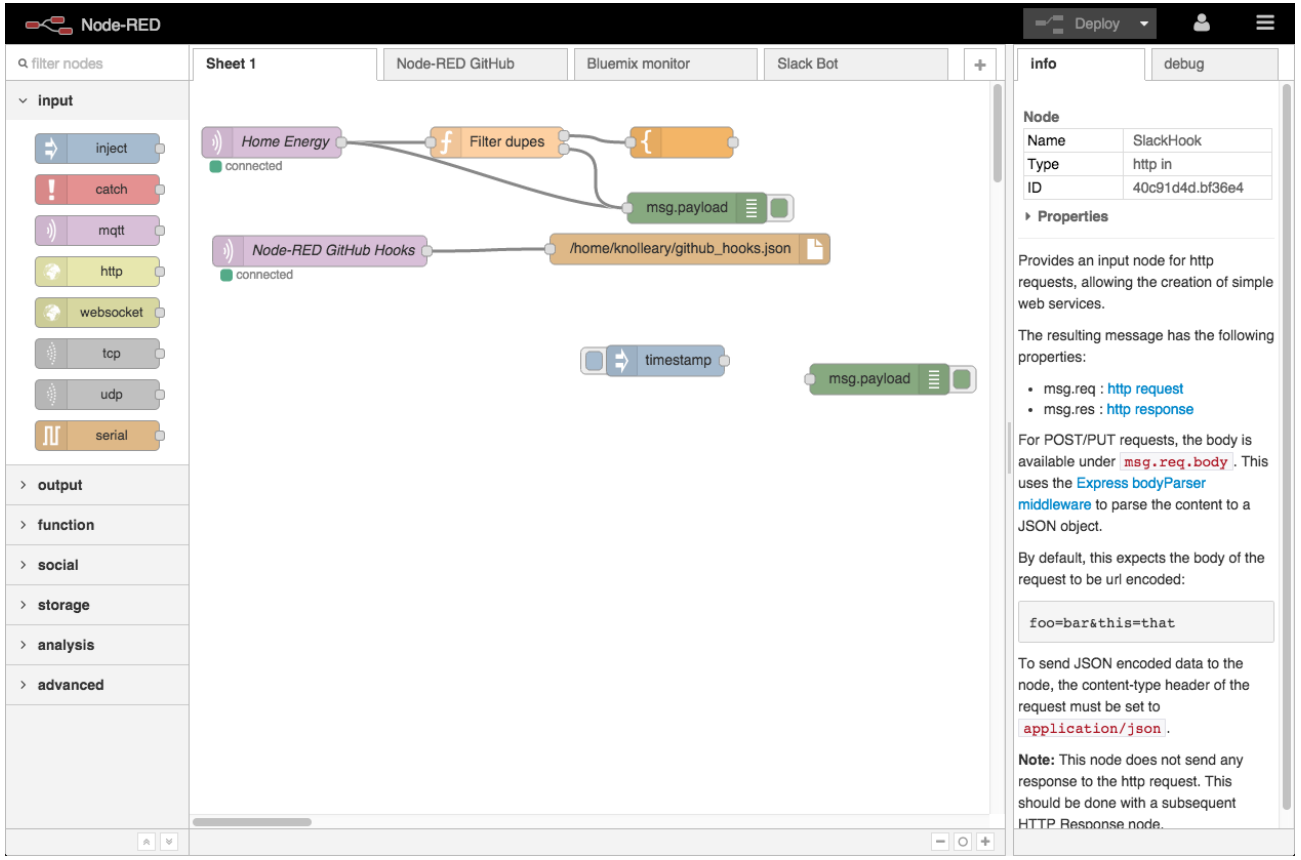
2. Open Command Prompt and type in the following code to install Node-RED as a global module including its all dependencies:
   npm install -g –unsafe-perm node-red



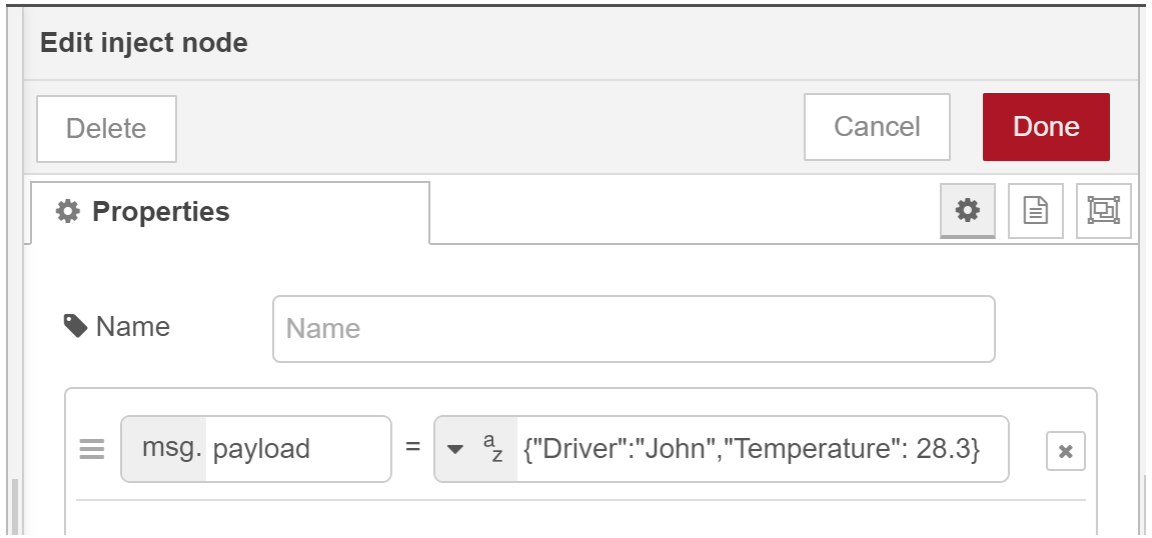
Running Node-RED

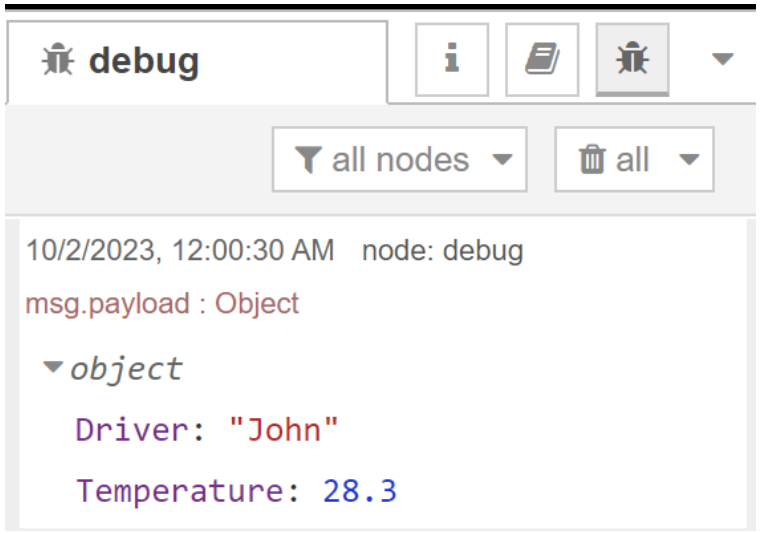
1. Once Node-RED is installed, in the Command Prompt type in **Node-red** to start Node-RED. If you want to stop Node-RED you can press Ctrl-C or close the terminal window.

2. Access the Node-RED editor by entering **http://localhost:1880** at your browser's address bar.



Data Type Conversion





Node-RED on RPi

Installing Node-RED on Raspberry Pi

1. Install and run Node-RED on a Raspberry Pi.
   https://nodered.org/docs/getting-started/raspberrypi

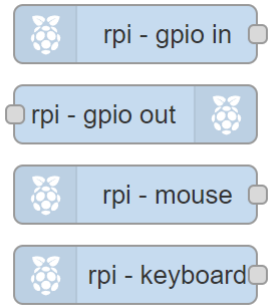
2. Enable the service to auto start Node-RED on boot.

```
sudo systemctl enable nodered.service
```

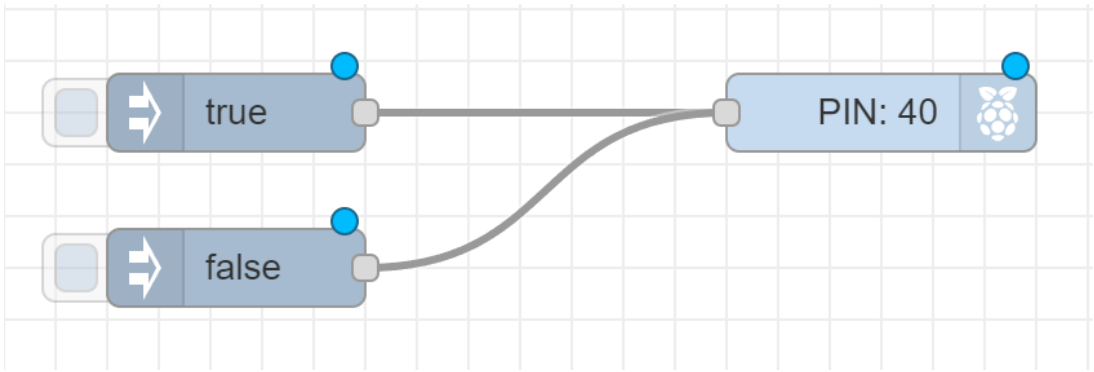
Node-RED on RPi

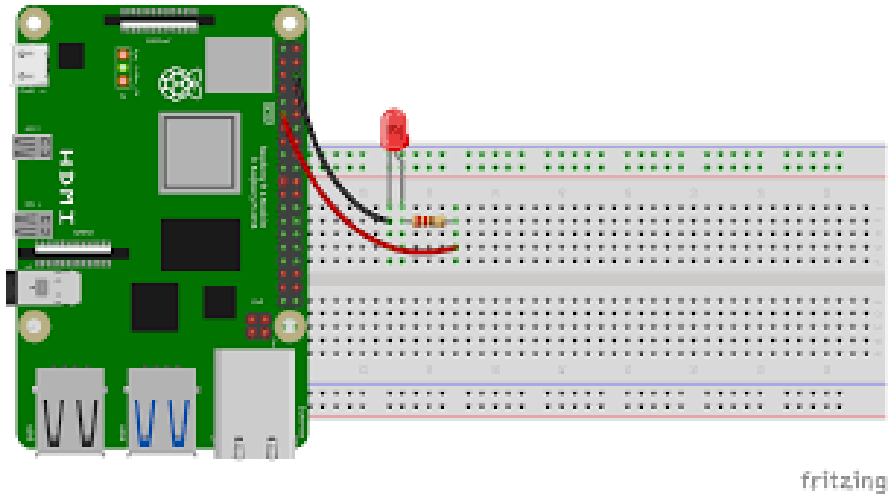
Interact with GPIO: Toggle LED

1. Make sure the **node-red-node-pi-gpio** is installed.



2. Use inject nodes and the **rpi-gpio out** node to control the LED.





Node-RED on RPi

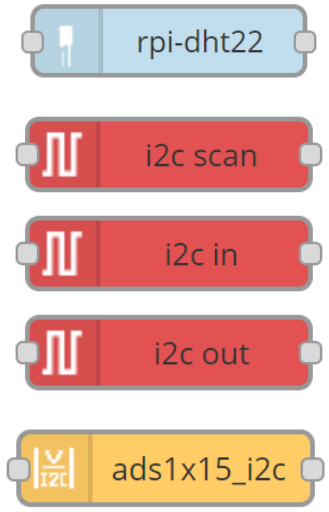
Interact with Sensors: DHT11/22 or any I2C device

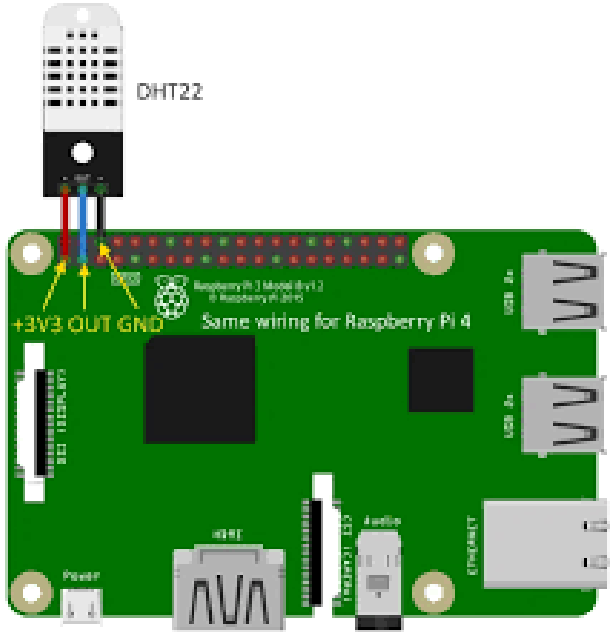
1. Install one of these nodes:

   node-red-contrib-dht-sensor

   node-red-contrib-i2c

   node-red-contrib-ads1x15_i2c





1. Make the flow to read the sensor data.


Thank You