ROHINI COLLEGE OF ENGINEERING AND TECHNOLOGY

Kanyakumari Main Road, near Anjugramam, Palkulam, Anjugramam, Tamil Nadu 629401

Department of Electronics and Communication Engineering

&

Masha Innovation Centre
Kamaraj Building,
Nagercoil-629001

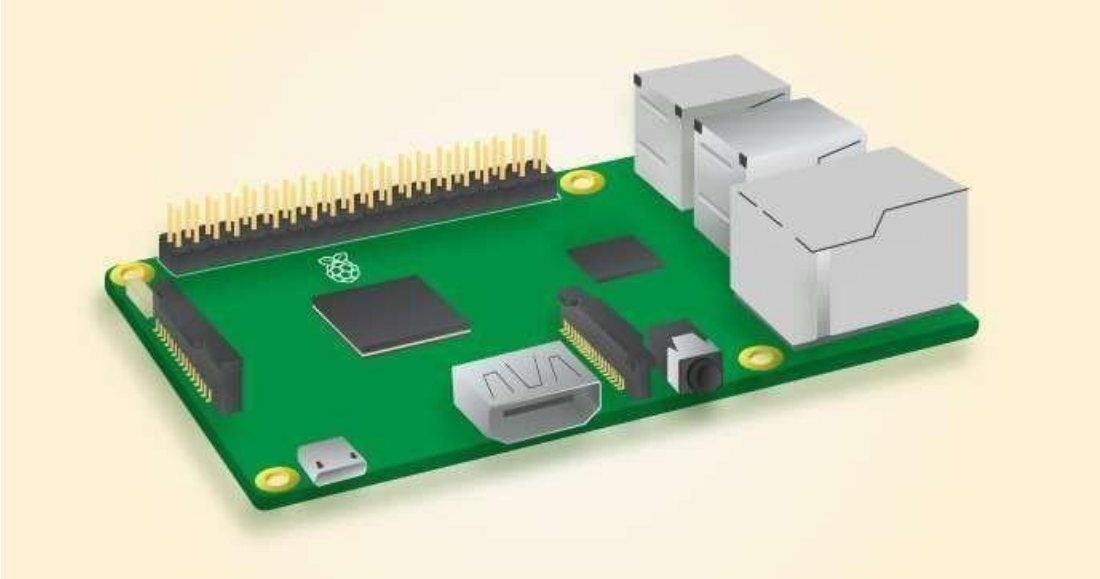
VALUE ADDED COURSE ON EMBEDDED USING RASPBEERY PI

COURSE MATERIAL

PHYSICAL DEVICE RASPBERRY PI

The Raspberry Pi Computer:

- It may not look much like the computers you are used to, but this is because we are accustomed to seeing a computer in a case, with a monitor, keyboard and mouse attached.
- The Raspberry Pi comes without any of these peripheral input, output and storage devices. It is known as a single board computer, and the fact that it comes without any additional peripherals, and uses hardware components more usually found in mobile phones and tablets, means it can be sold for as



little as $35.

- The Raspberry Pi has several ports that enable you to connect a variety of devices.
- Input devices let you send data to a computer. The two most common input devicesare a keyboard and a mouse. You can plug a USB keyboard and mouse into two of the four USB ports on the Raspberry Pi.
- Output devices let the computer send data to a user. Two of the most common output devices are a monitor and speakers. You can connect an HDMI monitor or television to the Raspberry Pi using the single HDMI port. If you don't have an HDMI monitor, then you'll need to use an adaptor. You can connect speakers or headphones to the Raspberry Pi using the 3.5mm headphone port.
- Storage devices are used to store data. On most computers this would be handled bya hard drive. Most modern computers, tablets and mobile phones now use Solid State

  storage devices. The Raspberry Pi uses a type of Solid State storage device

called a microSD card. This will be used to store the Operating System, your software, and all the files you create.

- The last thing you'll need to do is provide your Raspberry Pi with power. For this, we use a micro-USB power supply.

RASPBERRY PI INTERFACES

- There are many peripherals that can be added to a microprocessor over the I2C and SPI serial interfaces. These include atmospheric sensors, EEPROMS, and several types of display.

The Pi Wedge helps access the I2C and SPI signals.

- This tutorial will walk you through getting the I2C and SPI interfaces of your Raspberry Pi working. These interfaces aren't enabled by default, and need some extra configuration before you can use them.

Background & Software Setup

- The Raspberry Pi has three types of serial interface on the GPIO header. You're probably already familiar with the UART serial port, which allows you to open a login session from a serial terminal application, such as PuTTY.
- The other two serial interfaces are the Serial Peripheral Interface (SPI) and Inter-Integrated-Circuit bus (I2C). SPI on the Pi allows for up to two attached devices, while I2C potentially allows for many devices, as long as their addresses don't conflict.

Software Details

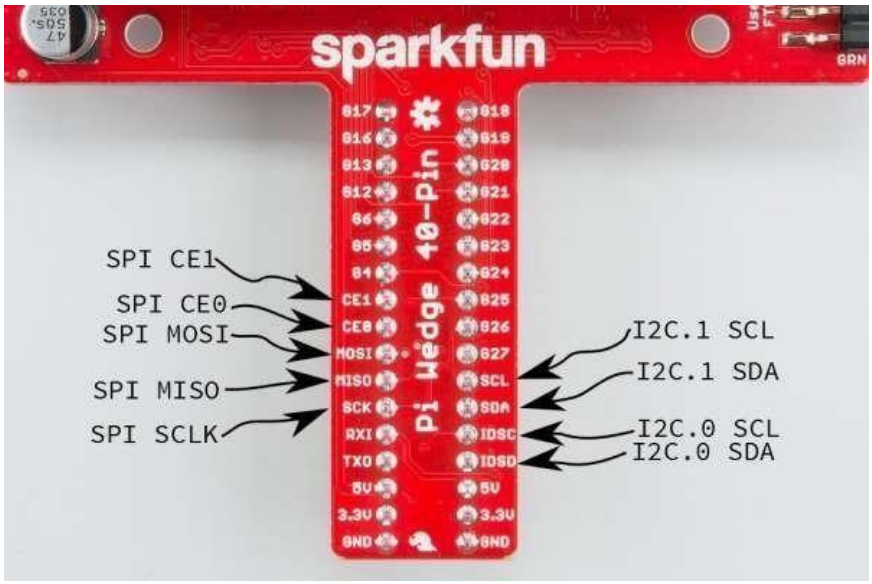
- The software landscape for the Raspberry Pi has evolved considerably since the introduction of the Pi. Many different operating systems have been ported to the Pi, and the device driver infrastructure has also changed quite a bit.
- For this tutorial, we'll be using a recent version of Raspbian (installed via NOOBS), and the wiringPi I/O library for C/C++ (or spidev/smbus for Python).
- With the implementation of device tree overlays in Raspbian, some of the specific interface enablement details have changed. If you're working with an older install, it might be worth backing up your SD card, and starting with a fresh install.

Connecting To the Ports

- Before we get into the configuration and software examples, let's locate the pins used by each of these interfaces.
- If you're directly connecting to the pins on the Pi, they're a little disorganized. I2C.1 is near one end, while SPI and I2C.0 are in the middle of the header. If you're connecting to these pins, be sure to count carefully.

Pi Serial Bus Pins

- The Pi Wedge adapter PCB rearranges the pins, and labels them clearly. We'll be using the Wedge for the following examples.

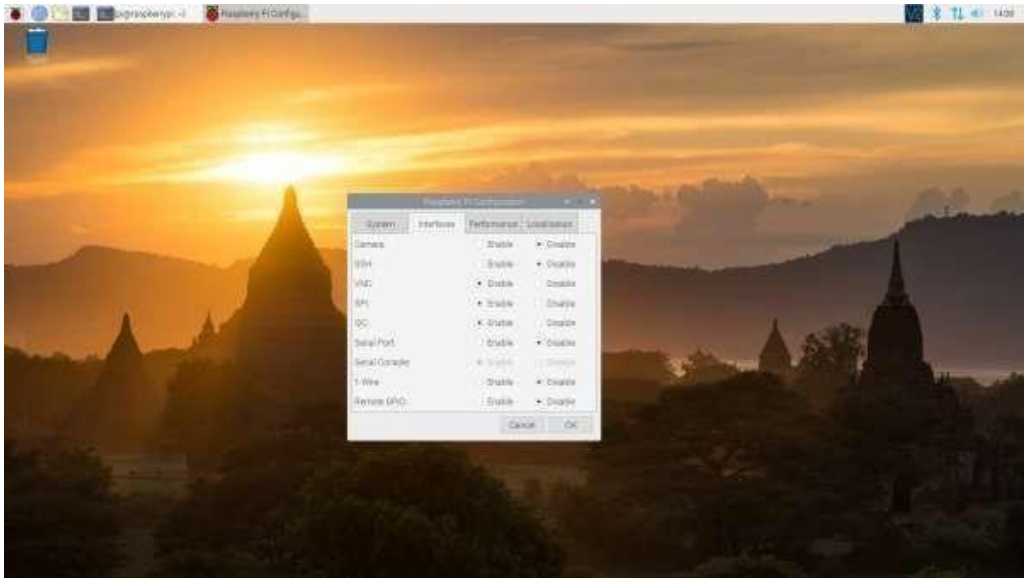


Wedge Serial Bus Pins

SPI on Pi connfiguration

- The SPI peripheral is not turned on by default. There are two methods to adjust the settings. To enable it, do the following.
- Raspberry Pi Configuration via Desktop GUI
- You can use the Desktop GUI by heading to the Pi Start Menu > Preferences > Raspberry Pi Configuration.

- A window will pop up with different tabs to adjust settings. What we are interested is the Interfaces tab. Click on the tab and select Enable for SPI. At this point, you can enable additional interfaces depending on your project needs. Click on the OK button to same.



- We recommend restarting your Pi to ensure that the changes to take effect. Click on the Pi Start Menu > Preferences > Shutdown. Since we just need to restart, click on the Restart button.

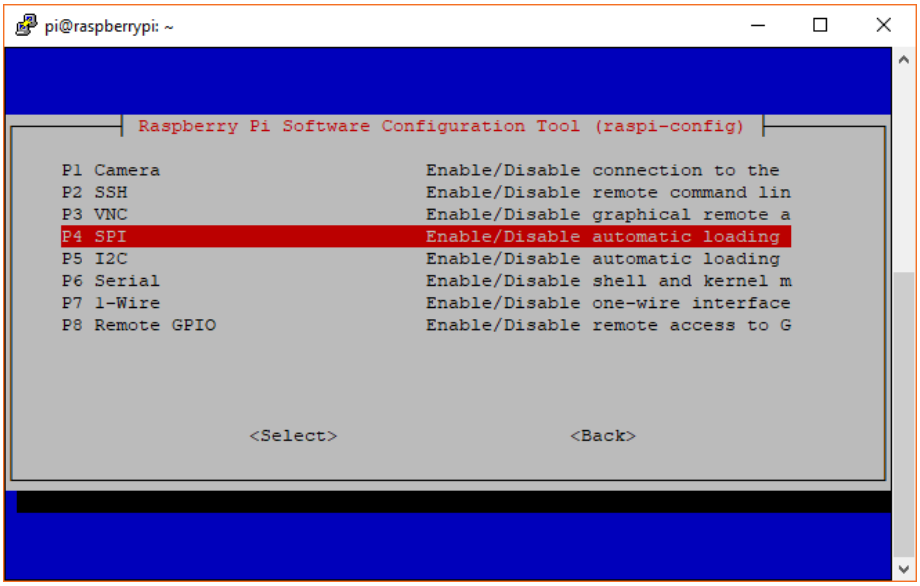
Shutdown

Turn Off, Restart, Log Off

raspi-config Tool

- If you are using a terminal, you will need to:
  1. Run sudo raspi-config.
  2. Use the down arrow to select 5 Interfacing Options
  3. Arrow down to P4 SPI.

4. Select yes when it asks you to enable SPI,
5. Also select yes if it asks about automatically loading the kernel module.
6. Use the right arrow to select the <Finish> button.
7. Select yes when it asks to reboot.



Raspi-config for SPI

- The system will reboot. When it comes back up, log in and enter the following command
  - **>ls /dev/*spi***
- The Pi should respond with
  - **/dev/spidev0.0 /dev/spidev0.1**
- These represent SPI devices on chip enable pins 0 and 1, respectively. These pins are hardwired within the Pi. Ordinarily, this means the interface supports at most two peripherals, but there are cases where multiple devices can be daisy-chained, sharing a single chip enable signal.

Programming Example

- Required Materials
  - The 40-pin Pi Wedge.
  - A Raspberry Pi B+ or Pi 2 Model B single board computer.
  - A Solderless Breadboard.
  - Some jumper wires.

- Headers of your choice.
- A Serial 7-Segment display.

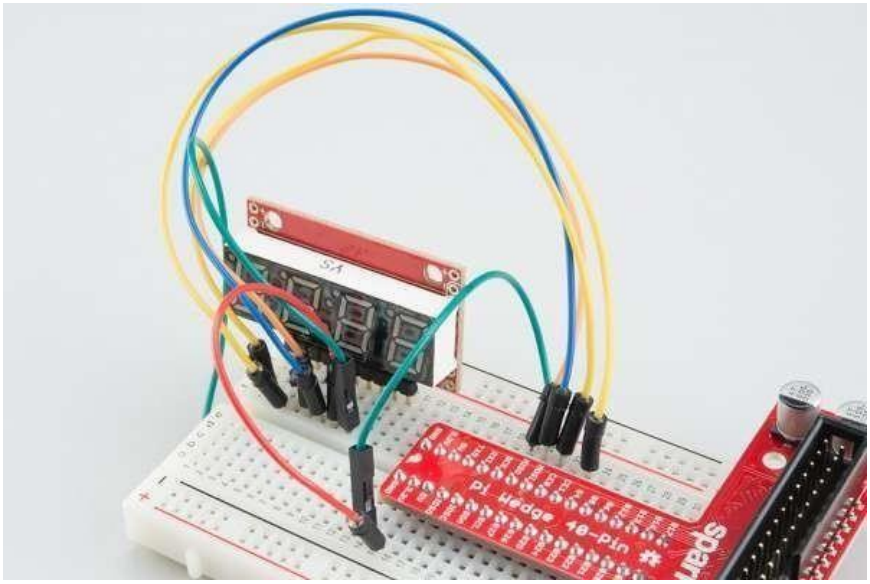
- The Serial 7-Segment display is particularly useful for testing serial interfaces, because it can accept command from a UART, SPI, or I2C. Make sure to solder header pins on the 7-segment display before wiring.

Hookup Table

- The display was connected to the Pi, via the Pi Wedge, as follows.

Raspberry Pi Signal	Serial 7-seg Signal
GND	GND
3.3V	VCC
CE1	SS (Shift Select)
SCK	SCK
MOSI	SDI
MISO	SDO

- The test hardware looked like this.



Serial 7-Segment connections for SPI

Sample Python Program

```
# spitest.py
# A brief demonstration of the Raspberry Pi SPI interface, using
the Sparkfun # Pi Wedge breakout board and a SparkFun Serial 7
Segment display:
#
https://www.sparkfun.com/products/116
29 import time
import spidev
# We only have SPI bus 0 available to us on
the Pi bus = 0
#Device is the chip select pin. Set to 0 or 1, depending on the
connections device = 1
# Enable SPI
spi = spidev.SpiDev()
# Open a connection to a specific bus and device (chip select pin)
spi.open(bus, device)
# Set SPI speed and
mode
spi.max_speed_hz =
500000
spi.mode =
0 # Clear
display msg
= [0x76]
spi.xfer2(ms
g)
time.sleep(5
)
# Turn on one segment of each character to show that
```

```
we can # address all of the segments
i = 1
while i < 0x7f:
   # The decimals, colon and
   apostrophe dots msg = [0x77]
   msg.append(i)
   result =
   spi.xfer2(msg) #
   The first character
   msg = [0x7b]
   msg.append(i)
   result =
   spi.xfer2(msg) #
   The second
   character msg =
   [0x7c]
   msg.append(i)
   result =
   spi.xfer2(msg) #
   The third character
   msg = [0x7d]
   msg.append(i)
   result =
   spi.xfer2(msg) #
   The last character
   msg = [0x7e]
   msg.append(i)
   result = spi.xfer2(msg)
   # Increment to next segment in each
   character i <<= 1
```

 # Pause so we can see

them time.sleep(5)

 # Clear display

again msg =

[0x76]

spi.xfer2(msg)

- Save the program with a name like spitest.py, and run it with:
  - **> python spitest.py**
- This will illuminate each segment in each character for 5 seconds before moving on to the next segment. It should take about 40 seconds for the whole program to run.

I2C on Pi

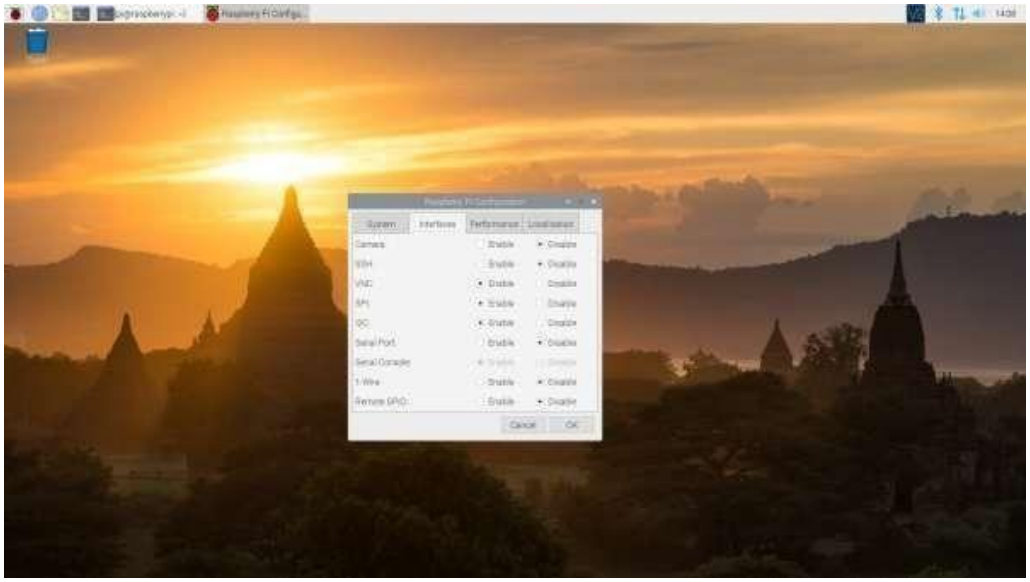
Configurati

on

- The I2C peripheral is not turned on by default. There are two methods to adjust the settings just like the SPI. To enable it, do the following.
- Raspberry Pi Configuration via Desktop GUI
- You can use the Desktop GUI by heading to the Pi Start Menu > Preferences > Raspberry Pi Configuration.

- A window will pop up with different tabs to adjust settings. What we are interested is the Interfaces tab. Click on the tab and select Enable for I2C. At this point, you

can enable additional interfaces depending on your project needs. Click on the OK button to same.



- Click on image for a closer view.
- We recommend restarting your Pi to ensure that the changes to take effect. Click on the Pi Start Menu > Preferences > Shutdown. Since we just need to restart

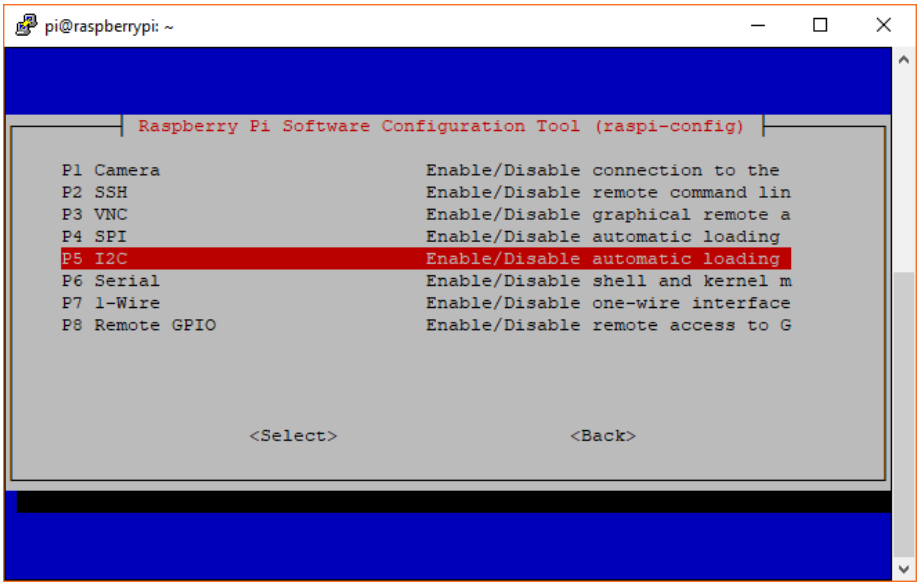
Shutdown **Turn Off, Restart, Log Off**

raspi-config Tool

Like the SPI peripheral, I2C is not turned on by default. Again, we can use raspi-config to enable it.

1. Run sudo raspi-config.
2. Use the down arrow to select 5 Interfacing Options
3. Arrow down to P5 I2C.
4. Select yes when it asks you to enable I2C

5. Also select yes if it asks about automatically loading the kernel module.
6. Use the right arrow to select the <Finish> button.
7. Select yes when it asks to reboot.



Raspi-config for I2C

- The system will reboot. when it comes back up, log in and enter the following command
  - **>ls /dev/*i2c***
- The Pi should respond with
  - **/dev/i2c-1**
- Which represents the user-mode I2C interface.
- Utilities
- There is a set of command-line utility programs that can help get an I2C interface working. You can get them with the apt package manager.
  - **sudo apt-get install -y i2c-tools**
- In particular, the i2cdetect program will probe all the addresses on a bus, and report whether any devices are present.

pi@raspberrypi:~/$ i2cdetect -

y 1 0 1 2 3 4 5 6 7 8 9 a b

c d e f

00: -- -- -- -- -- -- -- -- -- -- -- ----

10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- ---

30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- ---

40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- ---

50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- ---

60: 60 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

70: -- -- -- -- -- -- -- --

- This map indicates that there is a peripheral at address 0x60. We can try to read andwrite its registers using the i2cget, i2cset and i2cdump commands.

Programming Example

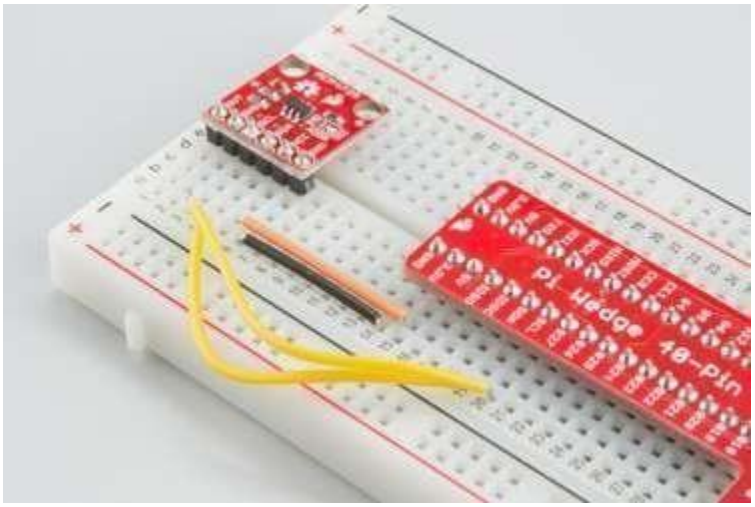
- Required Materials
- The 40-pin Pi Wedge.
- A Raspberry Pi B+ or Pi 2 Model B single board computer.
- A Solderless Breadboard.
- Some jumper wires.
- Header pins of your choice.
- An MCP4725 digital-to-analog converter.

Hookup Table

- The display was connected to the Pi, via the Pi Wedge, as follows.

Raspberry Pi Signal	MCP4725
GND	GND
3.3V	VCC
SCL	SCL
SDA	SDA

- The test hardware looked like this.



DAC on a Breadboard

Sample Python Program

```
# i2ctest.py
# A brief demonstration of the Raspberry Pi I2C interface, using the
Sparkfun # Pi Wedge breakout board and a SparkFun MCP4725
breakout board:
#
https://www.sparkfun.com/products/87
36 import smbus
# I2C channel 1 is connected to the
GPIO pins channel = 1
# MCP4725 defaults to address
0x60 address = 0x60
# Register addresses (with "normal mode" power-
down bits) reg_write_dac = 0x40
# Initialize I2C (SMBus)
bus = smbus.SMBus(channel)
# Create a sawtooth wave 16
times for i in range(0x10000):
  # Create our 12-bit number representing relative
  voltage voltage = i & 0xfff
```

 # Shift everything left by 4 bits and separate bytes

msg = (voltage & 0xff0)

>> 4 msg = [msg, (msg &

0xf) << 4]

 # Write out I2C command: address, reg_write_dac,

msg[0], msg[1] bus.write_i2c_block_data(address,

reg_write_dac, msg)

- Save the program with a name like i2ctest.py, and run it with the command:
  - **python**

i2ctest.py APIS /

PACKAGES

- Wouldn't it be interesting to have an API to control our hardware, like Arduino or Raspberry Pi? We now have the tools to make this possible and easy.
- Having an API for our hardware allows us to remotely control and/or monitor our hardware with code. With this power, we can even build apps on top of our hardware!
- We'll walk through creating an API for the Raspberry Pi Zero W to control four LEDs, each connected to its own GPIO. The same idea works for other hardware too.
- First, we'll need to set up our Raspberry Pi Zero W. Unlike typical Pi setups, we can do this without a monitor. If you need help with this, check out this tutorial.
- To begin, we'll need to make sure our device is updated and the proper libraries are installed.

Update and Install Libraries

- First, let's update our Pi:
  - **sudo apt-get upgrade && sudo apt-get update**
- Next, install the libraries we need:
  - **sudo apt-get install git python-pip python-gpiozero python-pkg-resources**

Clone the Code

- Using Git, clone the code from GitHub:

- **git clone https://github.com/Losant/example-raspberry-pi-api.git**

- Change to the new project directory:
  - **cd example-raspberry-pi-api**
- In the folder, you'll see a file named "index.py". This is where the magic happens. You can see the contents of this file in the Code section.

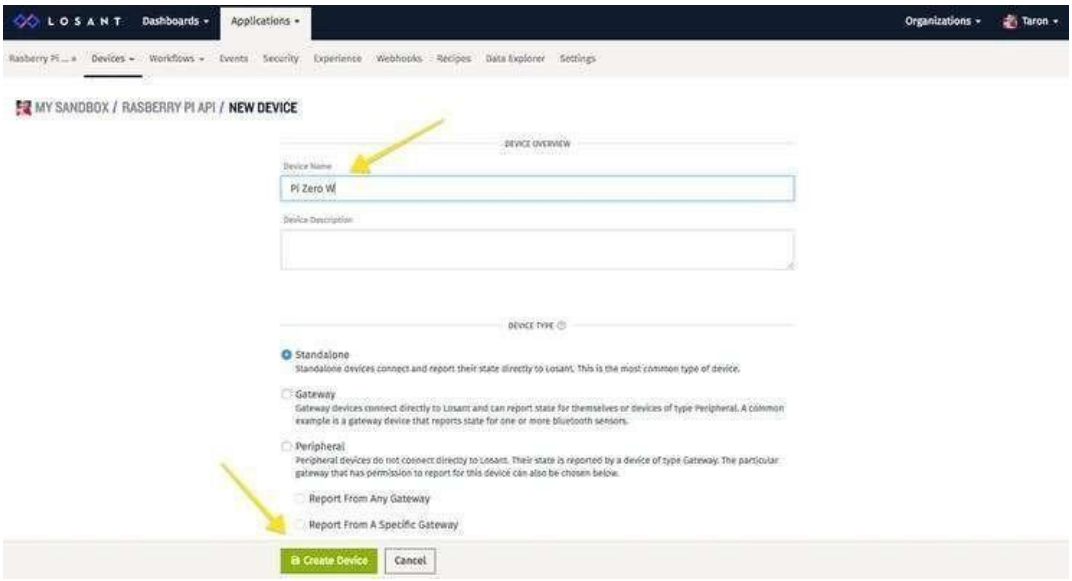
- We still need to configure one line of the code before we can run:
  - **device = Device("my-device-id", "my-app-access-key", "my-app-access-secret")**
- You can obtain the device ID, access key, and access secret from Losant.

Losant

- Losant is an easy-to-use and powerful developer platform designed to help you quickly build connected applications.

1. ***Log in and create an application in Losant.***
2. ***Create a device.***

- In Losant, a device could be Raspberry Pi, Arduino, smart bulb, or any custom hardware. Devices can contain many sensors or attached peripherals.



Create a device in Losant

3. ***Create an Access Key and Secret.***

- To connect your devices to the Losant Platform, you must use a set of security credentials called access keys. Access keys consist of a generated key and secret pair.

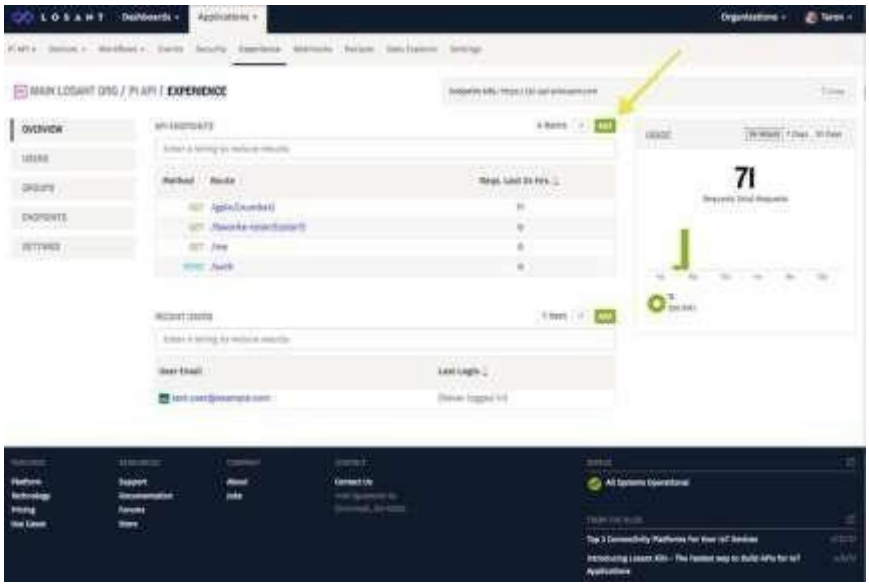
4. Create a Losant Experience.

- Writing an API service, implementing the entire user authentication, and hosting the result somewhere is a lot of work. An Experience in Losant brings all of this functionality directly inside your Losant application.
- Experiences are the key to building fully functional API allowing users to interact with our device.

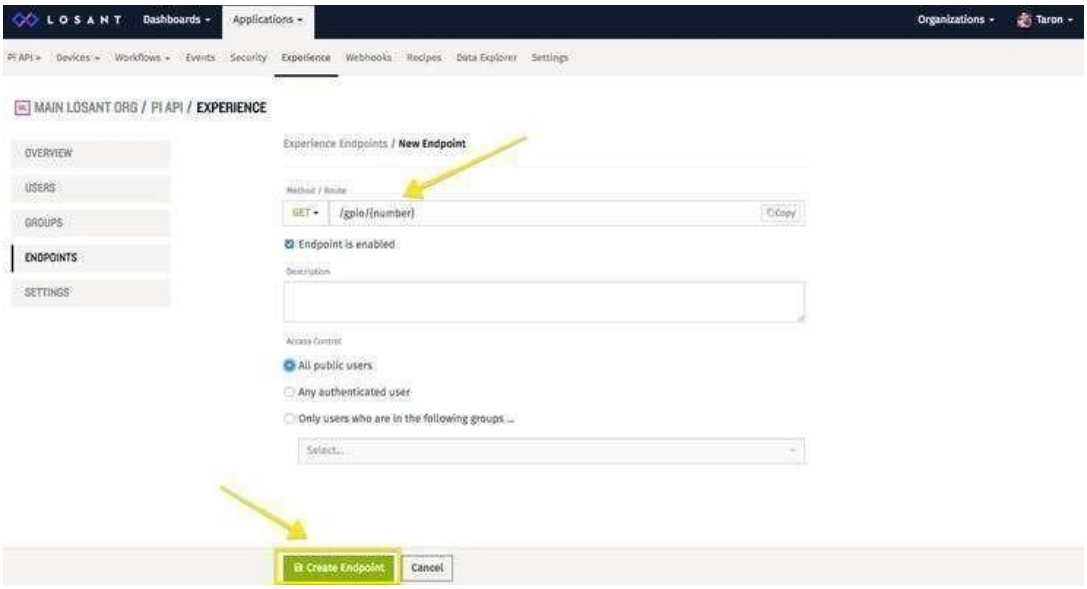
- After creation, a lot happens. We created a Walkthrough Video that walks you through the process and gives some background to what's going on.

5. Create an Endpoint

- An Endpoint is a combination of an HTTP method and a route that, when invoked by an HTTP request, can fire a workflow. That workflow does some work, like control an LED, and responds to the request.



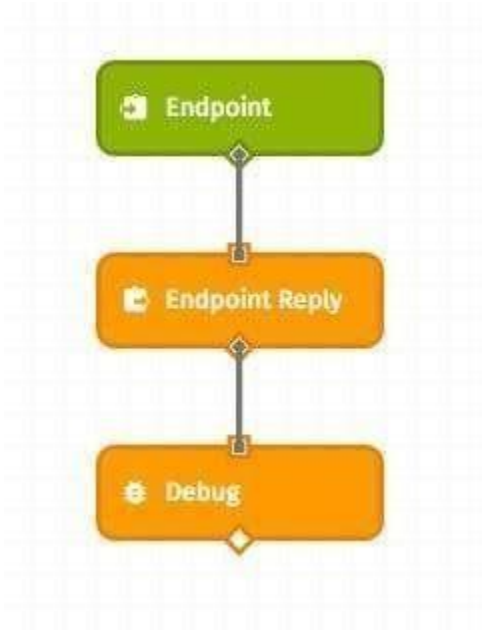
- For this example, we're going to create a GET request to toggle a GPIO pin. If the URL for this route would look something like this:
  - **https://example.onlosant.com/gpio/{number}**
- The method and route, in this case, would be:
  - **GET /gpio/{number}**
- "Number" in the route is a path parameter. We'll replace this with the GPIO we want to toggle.
- You can configure this in Losant like so:



- Note: Every endpoint has access control. To make things easy, our routes will be public. However, you are able to add authentication to endpoints.

6. Create a Workflow for the new endpoint

- Every Endpoint is powered by a Workflow. If you go down to the bottom of the endpoint configuration, you'll see the option to create a workflow for that route.
- Then, you'll have a basic workflow:

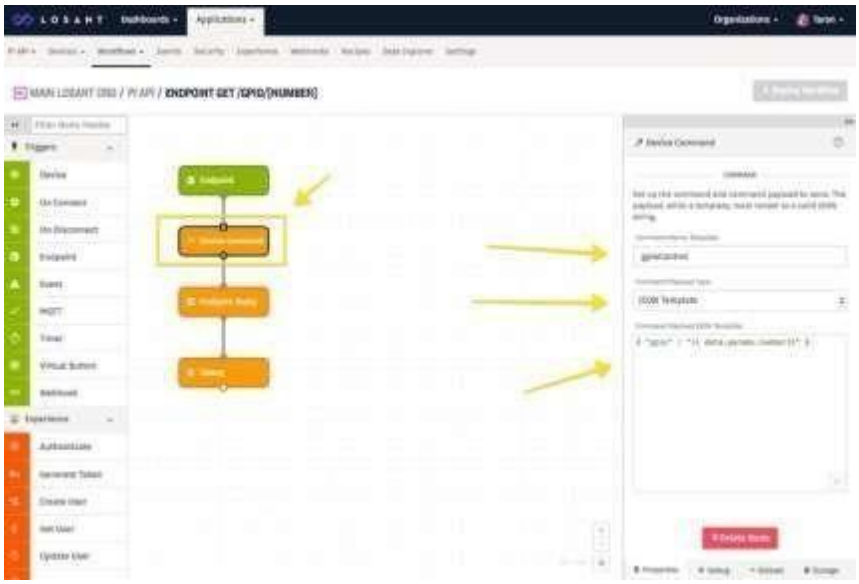


- Every endpoint workflow will start with an Endpoint Trigger and end with an Endpoint Reply. The Endpoint Trigger is fired when you make an HTTP request. The Endpoint

Reply responds to the HTTP request. So, this workflow isn't doing anything but responding immediately.

7. ***Update workflow to send device command.***

- If you take a peek in the Python code, there is an "on_command" function that's attached to an event.
  - **device.add_event_observer("command", on_command)**
- We can trigger this function by sending a Device Command to our Pi. Drag-and-Drop the Device Command node:
- In the code, we are also listening for a command called "gpioControl" and looking for a payload variable called "gpio".
- Select the Device Command node to configure it.
  - Command Name Template: gpioControl
  - Command Payload Type: JSON Template
  - Command Payload JSON Template:
  - **{ "gpio" : "{{ data.params.number}}" }**

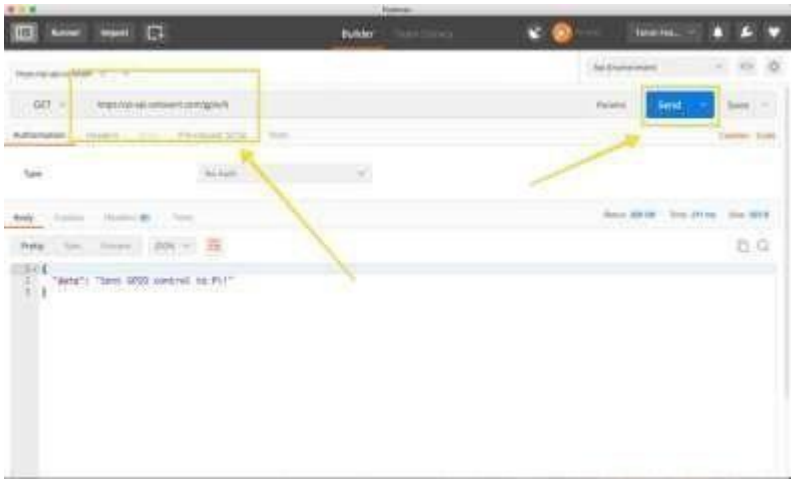


Wiring

- Now, you should wire the Raspberry Pi to the four 4 LEDs. This wiring diagram is supplied in the Wiring section.
- We'll connect the LEDs to 6, 13, 19, and 26.

Testing

- That's it! Now you are ready to test.
- To start your code:
  - **python index.py**
- To trigger an API request, we can use an app called Postman.
- Enter your URL and send the request:



- Once you fire the request, use 6, 13, 19, or 26 as the parameter and watch that LED light up!

WEB SERVICES

Various web servers are available, with different advantages for usage:

1. Apache
2. NGINX

1. *Apache*

- Apache is a popular web server application you can install on the Raspberry Pi to allow it to serve web pages.
- On its own, Apache can serve HTML files over HTTP, and with additional modulescan serve dynamic web pages using scripting languages such as PHP.

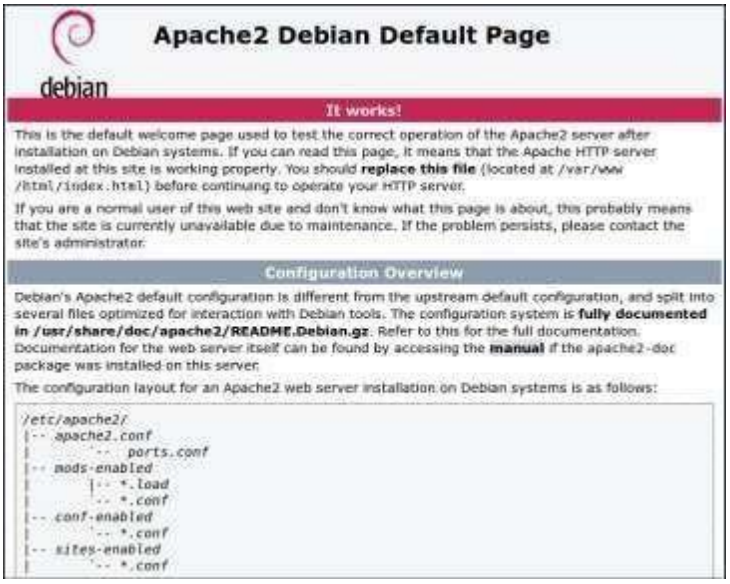
Install Apache

- First, update the available packages by typing the following command into the Terminal:
  - **sudo apt update**

- Then, install the apache2 package with this command:
    - **sudo apt install apache2 -y**

Test the web server

- By default, Apache puts a test HTML file in the web folder. This default web page is served when you browse to http://localhost/ on the Pi itself, or http://192.168.1.10 (whatever the Pi's IP address is) from another computer on the network. To find the Pi's IP address, type hostname -I at the command line (or read more about finding your IP address).
- Browse to the default web page either on the Pi or from another computer on the network and you should see the following:



- This means you have Apache working!

Changing the default web page

- This default web page is just an HTML file on the file system. It is located at
    - **/var/www/html/index.html.**
- Navigate to this directory in a terminal window and have a look at what's inside:
    - **cd /var/www/html**
    - **ls -al**

This will show you:

- **total 12**
- **drwxr-xr-x 2 root root 4096 Jan 8 01:29 .**

- **drwxr-xr-x 12 root root 4096 Jan 8 01:28 ..**
- **-rw-r--r-- 1 root root 177 Jan 8 01:29 index.html**

- This shows that by default there is one file in /var/www/html/ called index.htmland it is owned by the root user (as is the enclosing folder). In order to edit the file, you need to change its ownership to your own username. Change the owner of the file (the default pi user is assumed here) using sudo chown pi: index.html.
- You can now try editing this file and then refreshing the browser to see the web page change.

Your own website

- If you know HTML you can put your own HTML files and other assets in this directory and serve them as a website on your local network.

Additional - install PHP

- To allow your Apache server to process PHP files, you'll need to install the latest version of PHP and the PHP module for Apache. Type the following command to install these:
  - **sudo apt install php libapache2-mod-php -y**
- Now remove the index.html file:
  - **sudo rm index.html**
- and create the file index.php:
  - **sudo nano index.php**
- Put some PHP content in it:
  - **<?php echo "hello world"; ?>**
- Now save and refresh your browser. You should see "hello world". This is not dynamic but still served by PHP. Try something dynamic:
  - **<?php echo date('Y-m-d H:i:s'); ?>**
- or show your PHP info:
  - **<?php phpinfo(); ?>**

2. NGINX

- NGINX (pronounced engine x) is a popular lightweight web server application you can install on the Raspberry Pi to allow it to serve web pages.

- Like Apache, NGINX can serve HTML files over HTTP, and with additional modules can serve dynamic web pages using scripting languages such as PHP.

Refresh database of available packages

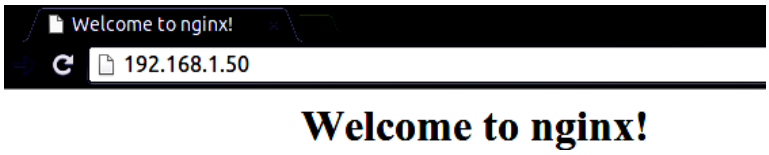
- Ensure that the package manager has up-to-date information about which packages are available:
  - **sudo apt update**
- You only need to do this occasionally, but it's the most likely solution if subsequent steps fail with messages like:
  - **404 Not Found [IP: 93.93.128.193 80]**

Install NGINX

- First install the nginx package by typing the following command in to the Terminal:
  - sudo apt install nginx
- and start the server with:
  - sudo /etc/init.d/nginx start

Test the web server

- By default, NGINX puts a test HTML file in the web folder. This default web page is served when you browse to http://localhost/ on the Pi itself, or http://192.168.1.10 (whatever the Pi's IP address is) from another computer on the network. To find the Pi's IP address, type hostname -I at the command line (or read more about finding your IP address).
- Browse to the default web page either on the Pi or from another computer on the network and you should see the following:



Changing the default web page

- NGINX defaults its web page location to /var/www/html on Raspbian. Navigate to this folder and edit or replace index.nginx-debian.html as you like. You can confirm the default page location at /etc/nginx/sites-available on the line which starts with 'root', should you need to.

Additional - Install PHP

sudo apt install php-fpm

- Enable PHP in NGINX

  cd /etc/nginx

  sudo nano sites-enabled/default

- find the line

  index index.html index.htm;

- roughly around line 25 (Press CTRL + C in nano to see the current linenumber)
- Add index.php after index to look like this:

  index index.php index.html index.htm;

- Scroll down until you find a section with the following content:

  # pass the PHP scripts to FastCGI server listening on

  127.0.0.1:9000 #

  # location ~ \.php$ {

- Edit by removing the # characters on the following lines:

  location ~ \.php$ {

  include snippets/fastcgi-php.conf;

  fastcgi_pass unix:/var/run/php5-

  fpm.sock;

  }

- It should look like this:

  # pass the PHP scripts to FastCGI server listening on

  127.0.0.1:9000 #

  location ~ \.php$ {

  include snippets/fastcgi-php.conf;

  # With php-fpm (or other unix sockets):

  fastcgi_pass unix:/var/run/php/php7.0-

  fpm.sock; # With php-cgi (or other tcp

  sockets):

  # fastcgi_pass 127.0.0.1:9000;

}

- Reload the configuration file

  sudo /etc/init.d/nginx reload

- Test PHP
- Rename index.nginx-debian.html to index.php:

  cd /var/www/html/

  sudo mv index.nginx-debian.html index.php

- Open index.php with a text editor:

  sudo nano index.php

- Add some dynamic PHP content by replacing the current content:

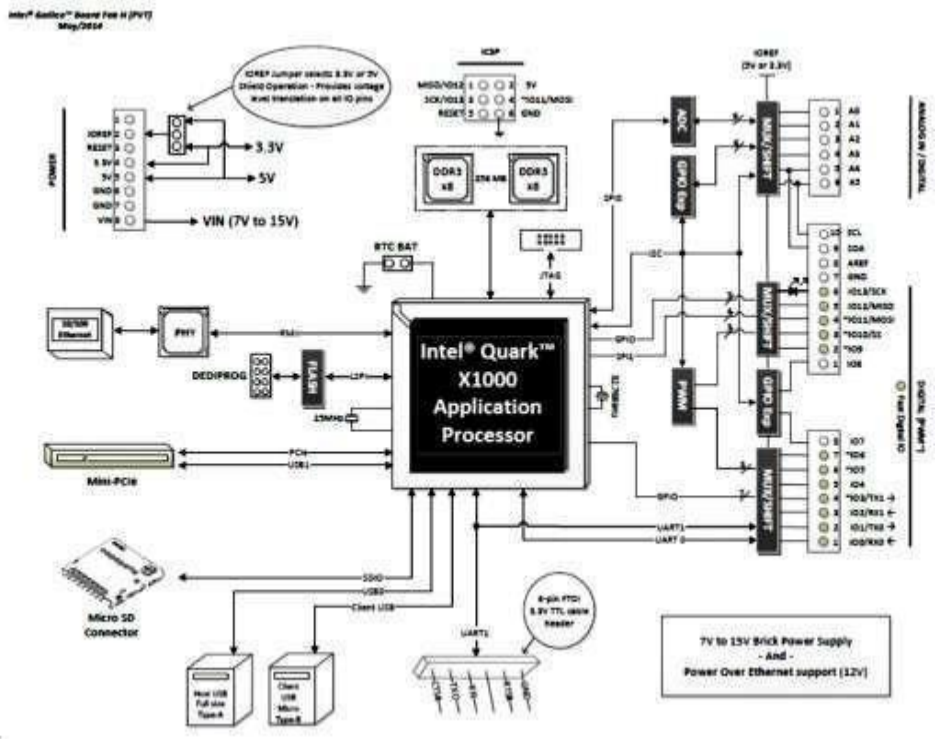
  <?php echo phpinfo(); ?>

- Save and refresh your browser. You should see a page with the PHP version, logo and current configuration settings.

INTEL® GALILEO GEN2 WITH ARDUINO

Overview

- The Intel® Galileo Gen2 is a board based on the Intel® Quark™ SoC X1000, a 32-bit Intel® Pentium® processor-class system on a chip (SoC), operating at speeds up to 400MHz.
- The Quark processor supports the Yocto 1.4 Poky Linux distribution.
- The board has built-in Ethernet with support for Power Over Ethernet(PoE), a USB 2.0 Host Port, micrso-SD slot, PCI Express mini-card slot, 20 digital input/output pins (of which 6 can be used as PWM outputs with 8/12-bits of resolution and 6 as analog inputs with 12 bits of resolution), a micro USB connection, an ICSP header, a JTAG header, and 2 reset buttons.
- There is also an integrated Real Time Clock (RTC), with an optional 3V "coin cell" battery for operation between turn on cycles.
- The Intel® Galileo Gen2 supports shields that operate at either 3.3v or 5v. The board is designed to be hardware and software pin-compatible with Arduino shields designed for the Uno R3. Digital pins 0 to 13 (and the adjacent AREF and GND pins), Analog inputs 0 to 5, the power header, ICSP header, and the UART port pins (0 and 1), are all in the same locations as on the Arduino Uno R3.



Supported Arduino libraries

The software release supports the following Arduino libraries:

- SPI
- EEPROM
- UART
- GPIO
- Wi-Fi
- Servo
- USB Host

Microcontroller	SoC Quark X1000
Operating Voltage	3.3V / 5V

Input Voltage	7-15V
Digital I/O Pins	14 (of which 6 provide 8/12-bit PWM output)
Analog Input Pins	6
Flash Memory	512 kB
RAM	256 MB DDR3
SRAM	512 kB
Flash Storage	8MB
EEPROM	8kB
Clock Speed	400 MHz
PoE compatible	-
Length	124 mm
Width	72 mm

Differences between the Intel® Galileo Gen2 and the Intel® Galileo

- The earlier Intel Galileo did not have an on-board regulator, so the power supply had to be exactly 5V. The Intel Galileo Gen2 has on-board regulator, so it may be powered with any suitable supply providing 7-15 VDC.

Differences between the Arduino Yùn and the Intel® Galileo Gen2

- The Arduino Yùn has two processors: One runs Linux, and an Atmel microcontroller runs the Arduino sketch. The Intel® Galileo Gen2 has one processor. In addition to running Linux, this processor runs the Arduino sketch.

Differences between the Intel® Galileo Gen2 and all Arduinos

- Note that a sketch loaded into the Intel® Galileo Gen2 is lost after a power cycle. It is possible to boot the Intel® Galileo Gen2 from a USD card, and in that case to restore a sketch from the same card. Instructions for doing so are forthcoming.

On-board Linux

- The Yocto 1.4 Poky Linux distribution is installed on your Intel Galileo Gen2. You can access various Linux functions with the system() call.

Power

- The Intel® Galileo Gen2 can be powered only via an external power supply. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. The board can operate on an external supply of 7 to 15 volts. The provided power supply provides 12v.

The power pins are as follows:

- VIN: The input voltage to the Intel® board. You can access the voltage supplied via the power jack through this pin.
- 5V: This pin outputs a regulated 5V from the regulator on the board.
- 3.3V: A 3.3 volt supply generated by the on-board regulator. This regulator also provides the power supply to the Quark microcontroller.
- GND: Ground pins.
- IOREF: This pin on the Arduino board provides the voltage reference with which the microcontroller operates. This can be 3.3V or 5V based on the IOREF jumper position.
- 12V power-over-Ethernet (PoE) capable

Buttons

There are 2 reset buttons with different functions on the board:

- Reboot: resets the Quark X1000 processor
- Reset: resets sketch and any attached shield

Memory

- The Quark X1000 has 512kB of embedded SRAM.
- The board has an additional 256 MB DDR3 Ram amd 8MB of Flash to store firmware and the Arduino sketch. It's possible to update the firmware using the IDE. The on board uSD supports uSD card up 32G, and can be used to provide a complete Yocto 1.4 Poky Linux image.

Input and Output

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data.

- Digital I/O: Digital pins 0 through 13 and Analog pins A0 through A5 can be used as a digital input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They can operate at 3.3 or 5 volts.

Each pin can provide (source) or receive (sink) a current of 16mA @ 5v, or a current of 8mA@3.3V

- PWM: Pins 3,5,6,9,10, and 11
    - Provide 8/12bit PWM output with the analogWrite() function. The resolution of the PWM can be changed with the analogWriteResolution() function.
- SPI: SPI header (ICSP header on other Arduino boards)
    - These pins support SPI communication using the SPI library.
- Analog Inputs: pins A0 through A5
    - Each analog input provide 10/12 bits of resolution. The resolution can be changed with the analogReadResolution() function.
- SDA and SCL: Support TWI communication using the Wire

library. There are 2 reset buttons with different functions on the

board:

- Reboot:
    - resets the Quark X1000 processor
- Reset:
    - resets sketch and attached shield

Communication

- The Intel® Galileo Gen2 has a number of facilities for communicating with a computer, another Galileo, Arduino or other microcontrollers, and different devices like phones, tablets, cameras and so on.
- It provides 2 UART Controllers : UART 0 to Galileo headers 0, 1; UART 1 to 6-pin 3.3V USB TTL FTDI header; optionally directed to Galileo headers 2, 3 .
- The Native USB port can also act as a USB host for connected peripherals such as mice, keyboards, and smart phones. To use these features, see the USB Host reference pages. The onboard microSD card reader is accessible through the SD Library. The communication between Galileo and the SD card is provided by an integrated SD controller and does not require the use of the SPI interface like other Arduinoboards.

- The onboard Ethernet interface is fully supported, use the Ethernet Library. It does not require the use of the SPI interface like existing Arduino shields.
- The Arduino software includes a Wire library to simplify use of the TWI/I2C bus; see the documentation for details. For SPI communication uses the SPI library.
- The board provides also a mini PCI Express (mPCIe) slot. This slot allows full size and half size (with adapter) mPCIe modules to be connected to the board and also provides an additional USB Host port via the slot. Any standard mPCIe module can be connected and used to provide applications such as WiFi, Bluetooth or Cellular connectivity.

Programming

- The Intel® Galileo Gen2 can be programmed with this special version of the Arduino software. It's possible to make requests of the Linux kernel with system() calls. This gives your Arduino sketch access to powerful utilities like Python, Node.js, OpenCV, and all sorts of fun Linux stuff.
- Note that Intel® Galileo Gen2 forgets the sketch after powering down. It is possible to booth the Galileo Gen2 from the uSD card, and in that case, to restore a sketch from the same card. Instructions for that will be forthcoming.

ARDUINO INTERFACES

GPS Module Interfacing With Arduino UNO

- Interfaced GPS receiver module with Arduino UNO and display the Time, Latitude, Longitude, and Altitude info on the Serial window. Arduino read the data serially from GPS receiver using USART communication with 9600 Baud rate.

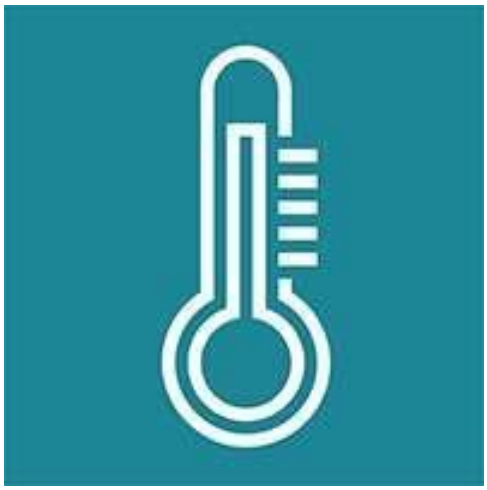
HC-05 Bluetooth Module Interfacing With Arduino UNO

- Bluetooth is a wireless communication protocol used to communicate over short distances. HC-05 Bluetooth module uses serial communication to talk with microcontrollers.

Analog Joystick Interfacing With Arduino UNO

- Analog Joystick is an input device used to control the pointer movement in 2- Dimensional axes. Generally, joystick is used for getting angular movements.

LM35 Interfacing With Arduino UNO

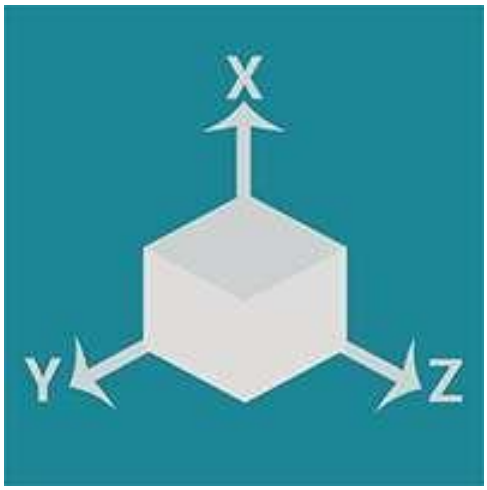


- LM35 is a sensor which is used to measure temperature. It provides an electrical output proportional to the temperature (in Celsius).

4x4 Keypad Interfacing With Arduino UNO

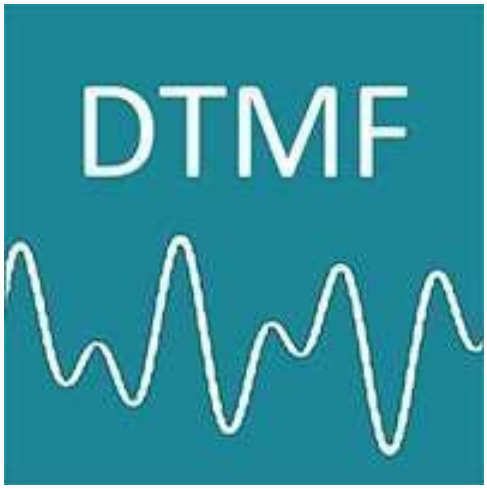
- Keypad is an input device which is generally used in applications such as calculators, ATM machines, computers etc.

ADXL335 Accelerometer Interfacing With Arduino Uno



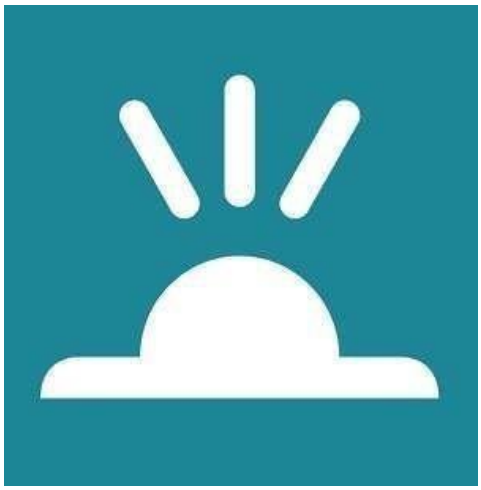
- ADXL335 accelerometer sensor measures acceleration due to gravity. It is used to measure the angle of tilt or inclination in application systems such as in Mobile devices, Gaming applications, Laptops, Digital cameras, Airplanes etc.

MT8870 DTMF Decoder Interfacing With Arduino UNO



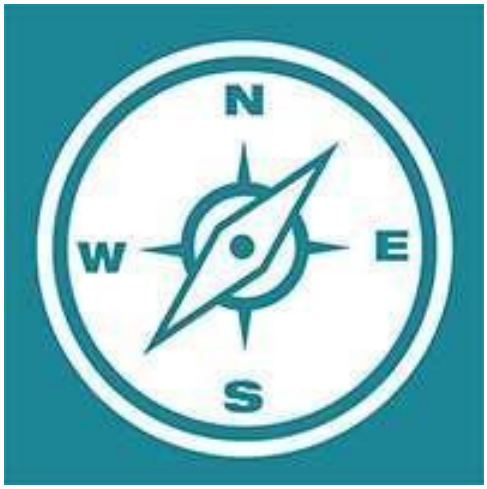
- MT8870 is a DTMF (Dual Tone Multi-Frequency) receiver, which decodes the dial tone generated from a telephone keypad. It used in Interactive Voice Response Systems (IVRS), Remote control, Credit card systems etc.

IR Communication Using Arduino UNO



- IR (infrared) communication is a wireless communication technology, used for short distance data/control transmission. It is commonly used in TV remotes, mobile phones, computers, and PDAs etc.

Magnetometer HMC5883L Interfacing With Arduino UNO

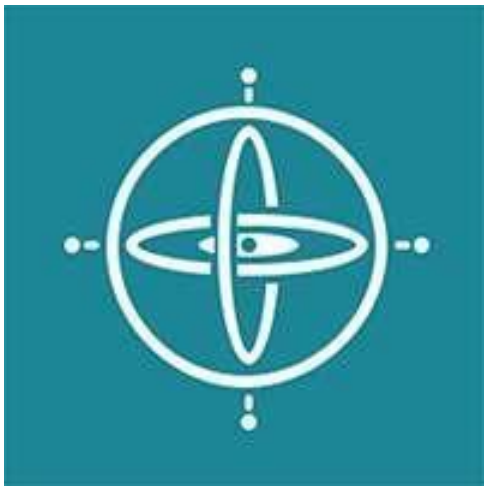


HMC5883L is a triple axis magnetometer developed by Honeywell. It provides the direction of heading. Magnetometer is used as a compass in Mobiles Phones and Navigation systems in vehicles to indicate directions.

PIR Sensor Interfacing With Arduino UNO

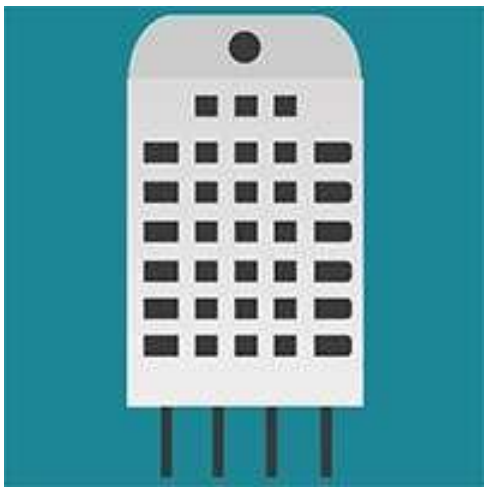
- PIR motion sensor senses Infrared signals. It is generally used to detect the motion of humans or animals.

MPU6050 Interfacing with Arduino UNO



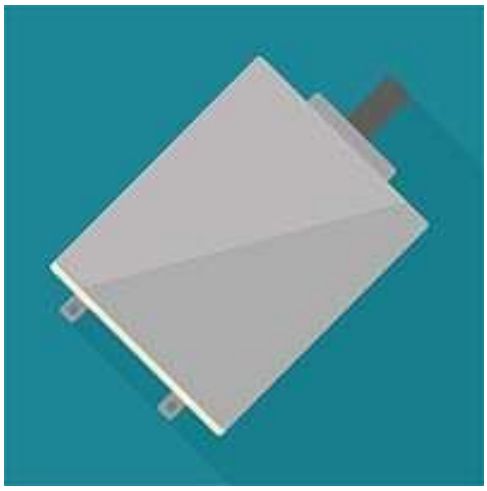
- MPU6050 sensor module is a combination of 3-axis Gyroscope, 3-axis Accelerometer and Temperature sensor with on-board Digital Motion Processor (DMP). It is used in mobile devices, motion enabled games, 3D mice, gesture (motion command) control technology etc.

DHT11 Sensor Interfacing With Arduino UNO



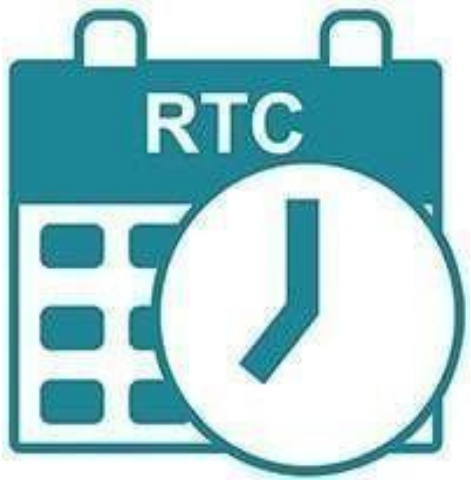
- Interface single wire DHT11 sensor with Arduino Uno to read the values of Temperature and Humidity. Display these Temperature and Humidity values on a serial window.

DC Motor Interfacing With Arduino UNO



- DC Motor is a device which converts electrical energy into mechanical energy. It is used in robotics field, toys, quad copters etc.

DS1307 RTC Module Interfacing With Arduino UNO

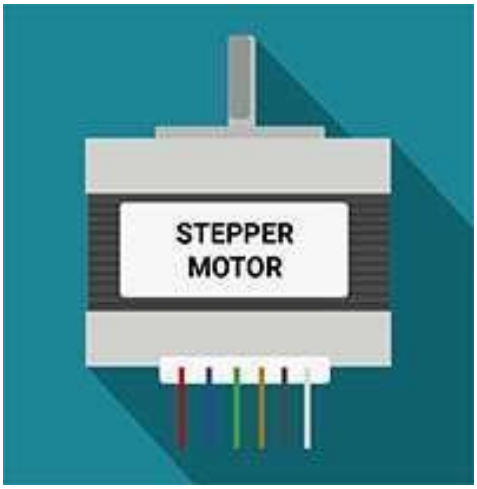


- DS1307 is a two wire (I2C) serial interface RTC (Real Time Clock) with 56 byte of nonvolatile RAM. This provides clock and calendar with second, minute, hour, day, date, month and year.

LCD 16x2 Interfacing With Arduino Uno

- LCD16x2 has two lines with 16 characters in each line. LCD16x2 is generally used for printing values and strings in embedded application

Stepper Motor Interfacing With Arduino UNO



- Rotate the Stepper Motor Clockwise and anti-clockwise by interfacing it with Arduino. Here, we are using a ULN2003 driver which is used to drive a stepper motor.

7-Segment Display Interfacing With Arduino UNO

- 7-segment LED displays are used for displaying numerical values from 0 to 9 and few characters like A, b, C, d, e, F, H, L, O, P, U etc.7-segment displays are widely used in digital clock

Servo Motor Interfacing With Arduino Uno

- Servo motor is an electromechanical device which consists of motor, gear assembly and feedback circuitry. It is used in Robotics applications, airplanes, rudders, quad copters, etc.

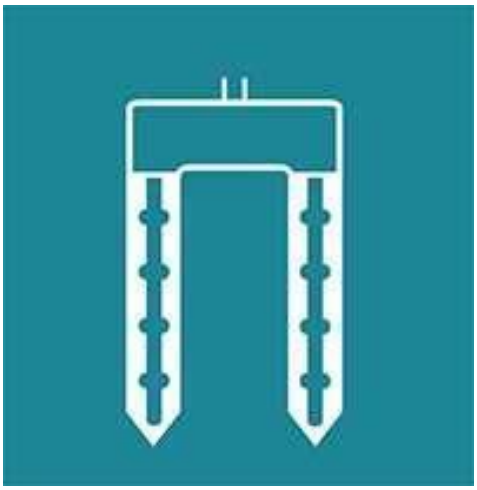
TCP Client Using SIM900A GPRS and Arduino UNO

- Transmission Control Protocol (TCP) is standard transport layer internet protocol in between server and client. Using SIM900A GPRS module, we can implement TCP server/client over GPRS for IoT applications

HTTP Client Using Sim900A GPRS And Arduino UNO

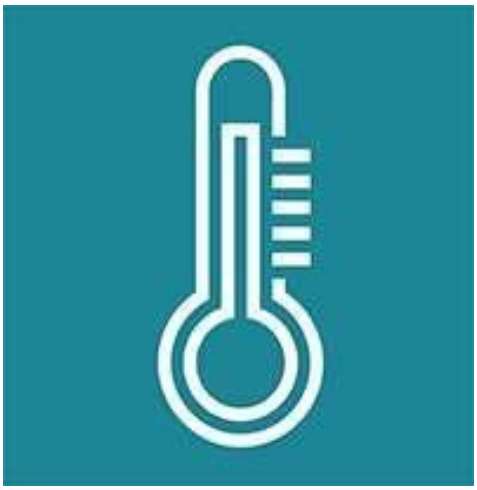
- Hypertext Transfer Protocol (HTTP) is standard application layer protocol in between server and client. Using SIM900A GPRS module, we can implement HTTP server/client over GPRS for IoT applications

Soil Moisture Sensor Interfacing With Arduino UNO



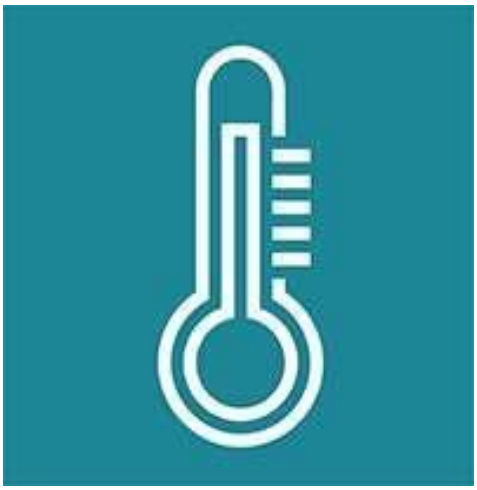
- Soil moisture sensor is used to measure the water content (moisture) in the soil. It is used in agriculture applications, irrigation and gardening systems, etc.

Thermistor Interfacing With Arduino UNO



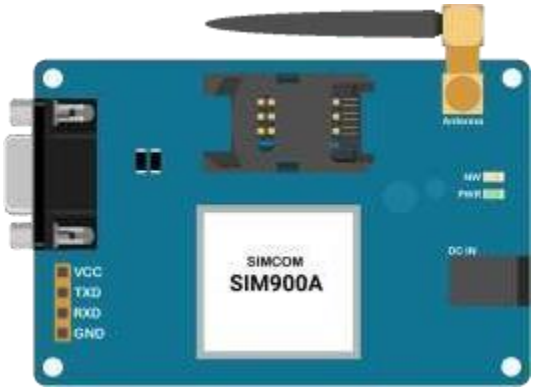
- Thermistor is a type of resistor whose resistance changes in accordance with change in temperature. It is used to measure the temperature over small range typically -100 °C to 300 °C

Thermocouple Interfacing With Arduino UNO



- Thermocouple is a device consisting of two dissimilar metals connected together creating a junction which is used for measuring the temperature

Sim900A GSM Module Interfacing With Arduino UNO

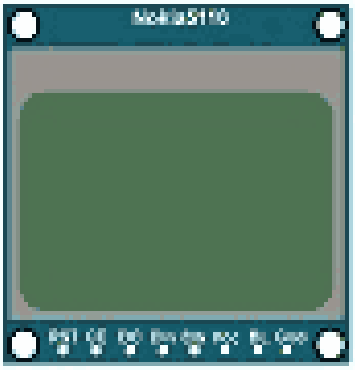


- SIM900A GSM module is a communication device which is used to make or receive calls, send or receive SMS, connect to the internet over GPRS

ESP8266 WiFi Module Interfacing With Arduino UNO

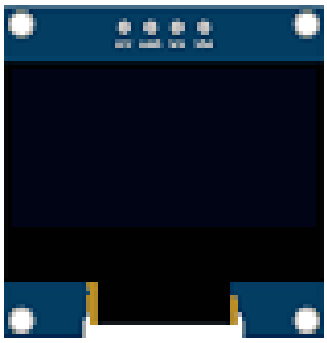
- ESP8266 is Wi-Fi enabled system on chip (SoC) module developed by Espressif system. It is mostly used for development of IoT (Internet of Things) embedded applications.

Nokia5110 Graphical Display Interfacing With Arduino UNO



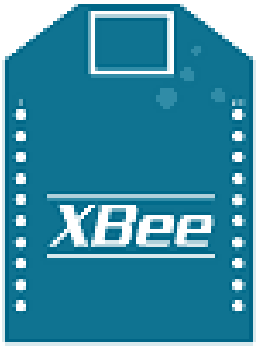
- Nokia5110 is 48x84 dot LCD display with Serial Peripheral Interface (SPI) Connectivity. It was designed for cell phones, also used in embedded applications

OLED Graphic Display Interfacing With Arduino UNO



- OLED graphic display modules are compact and have high contrast pixels which make these displays easily readable. They do not require backlight since the display creates its own light. Hence they consume less power. Both I2C and SPI based OLED modules are available in market

XBee S2 (ZigBee) Interfacing With Arduino UNO



- XBee is a radio module developed by Digi International. It is a popular wireless transceiver used to send or receive data

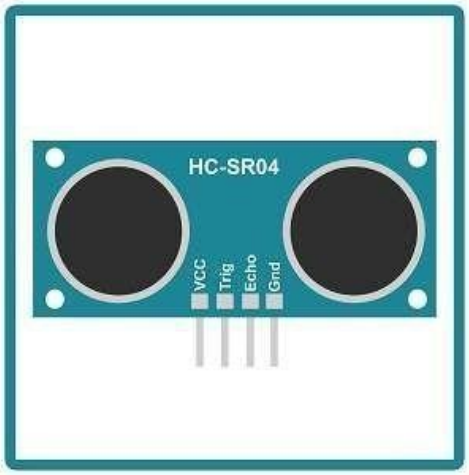
NRF24L01 Interfacing With Arduino UNO

- nRF24L01 is ultra-low power RF transreceiver radio module for 204GHz ISM band which is developed by Nordic Semiconductor.

MicroSD Card Interfacing With Arduino

- MicroSD (Secure Digital) card is an electronic digital data storage device, which provides storage in a gigabyte (GB) at low cost and small size. It is used in various applications like Mobile Phones, Laptops, Digital Camera, etc.

Ultrasonic Sensor HC-SR04 Interfacing With Arduino Uno



- Ultrasonic Sensor HC-SR04 interfaced with Arduino for finding a distance to an object. It can operate in the range of 2cm-400cm.

ARDUINO IDE

- The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

1. Writing Sketches

- Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right-hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.
- NB: Versions of the Arduino Software (IDE) prior to 1.0 saved sketches with the extension .pde. It is possible to open these files with version 1.0, you will be

prompted to save the sketch with the .ino extension on save.

- ***Verify:*** Checks your code for errors compiling it.
- ***Upload:*** Compiles your code and uploads it to the configured board. See uploading below for details.
- ***Note:*** If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"
- ***New:*** Creates a new sketch.
- ***Open:*** Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.
- ***Note:*** due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketchbook menu instead.
- ***Save:*** Saves your sketch.
- ***Serial:*** Monitor Opens the serial monitor.

- Additional commands are found within the five menus: File, Edit, Sketch, Tools, and Help. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

2. File

- ***New:*** Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.
- ***Open:*** Allows loading a sketch file browsing through the computer drives and folders.
- ***Open Recent:*** Provides a short list of the most recent sketches, ready to be opened.
- ***Sketchbook:*** Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.
- ***Examples:*** Any example provided by the Arduino Software (IDE) or library shows up in this menu item. All the examples are structured in a tree that allows easy access by topic or library.
- ***Close:*** Closes the instance of the Arduino Software from which it is clicked.
- ***Save:*** Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as.." window.
- ***Save as:*** Allows saving the current sketch with a different name.

- ***Page Setup:*** It shows the Page Setup window for printing.
- ***Print:*** Sends the current sketch to the printer according to the settings defined in Page Setup.
- ***Preferences:*** Opens the Preferences window where some settings of the IDE maybe customized, as the language of the IDE interface.
- ***Quit:*** Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.

3. **Edit**

- ***Undo/Redo:*** Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.
- ***Cut:*** Removes the selected text from the editor and places it into theclipboard.
- ***Copy:*** Duplicates the selected text in the editor and places it into theclipboard.
- ***Copy for Forum:*** Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax coloring.
- ***Copy as HTML:*** Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages.
- ***Paste:*** Puts the contents of the clipboard at the cursor position, in the editor.
- ***Select All:*** Selects and highlights the whole content of the editor.
- ***Comment/Uncomment:*** Puts or removes the // comment marker at the beginning of each selected line.
- ***Increase/Decrease Indent:*** Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.
- ***Find:*** Opens the Find and Replace window where you can specify text to searchinside the current sketch according to several options.
- ***Find Next:*** Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position.
- ***Find Previous:*** Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.

4. Sketch

- ***Verify/Compile:*** Checks your sketch for errors compiling it; it will report memory usage for code and variables in the console area.

- ***Upload:*** Compiles and loads the binary file onto the configured board through the configured Port.
- ***Upload Using Programmer:*** This will overwrite the boot loader on the board; you will need to use Tools > Burn Boot loader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so Tools -> Burn Boot loader command must be executed.
- ***Export Compiled Binary:*** Saves a .hex file that may be kept as archive or sent to the board using other tools.
- ***Show Sketch Folder:*** Opens the current sketch folder.
- ***Include Library:*** Adds a library to your sketch by inserting #include statements at the start of your code. For more details, see libraries below. Additionally, from this menu item you can access the Library Manager and import new libraries from .zip files.
- ***Add File:*** Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu accessible clicking on the small triangle icon below the serial monitor one on the right side o the toolbar.

5. Tools

- ***Auto Format:*** This formats your code nicely: i.e. indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more.
- ***Archive Sketch:*** Archives a copy of the current sketch in .zip format. The archive is placed in the same directory as the sketch.
- ***Fix Encoding & Reload:*** Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.
- ***Serial Monitor:*** Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over serial port opening.
- ***Board:*** Select the board that you're using. See below for descriptions of the various boards.
- ***Port:*** This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.
- ***Programmer:*** For selecting a hardware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't

need this, but if you're burning a boot loader to a new microcontroller, you will use this.

- ***Burn Boot loader:*** The items in this menu allow you to burn a boot loader onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino or Genuino board but is useful if you purchase a new ATmega microcontroller (which normally come without a boot loader). Ensure that you've selected the correct board from the Boards menu before burning the boot loader on the target board. This command also set the right fuses.

6. Help

- Here you find easy access to a number of documents that come with the Arduino Software (IDE). You have access to Getting Started, Reference, this guide to the IDE and other documents locally, without an internet connection. The documents are a local copy of the online ones and may link back to our online website.
- ***Find in Reference:*** This is the only interactive function of the Help menu: it directly selects the relevant page in the local copy of the Reference for the function or command under the cursor.

Sketchbook

- The Arduino Software (IDE) uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in your sketchbook can be opened from the File > Sketchbook menu or from the Open button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from with the Preferences dialog.
- Beginning with version 1.0, files are saved with an .ino file extension. Previous versions use the .pde extension. You may still open .pde named files in version 1.0 and later, the software will automatically rename the extension to .ino.

Tabs, Multiple Files, and Compilation

- Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

Uploading

- Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Port menus. The boards are described below. On the Mac, the serial port is probably something like /dev/tty.usbmodem241 (for a Uno or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or /dev/tty.USA19QW1b1P1.1 (for a serial board

connected with a Keyspan USB-to- Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the sports section of the Windows Device Manager. On Linux, it should be /dev/ttyACMx , /dev/ttyUSBx or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the Sketch menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

- When you upload a sketch, you're using the Arduino boot loader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The boot loader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The boot loader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

Libraries

- Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the Sketch > Import Library menu. This will insert one or more #include statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its #include statements from the top of your code.
- There is a list of libraries in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch.

Third-Party Hardware

- Support for third-party hardware can be added to the hardware directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, boot loaders, and programmer definitions. To install, create the hardware directory, then unzip the third-party platform into its own sub-directory. (Don't use "arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

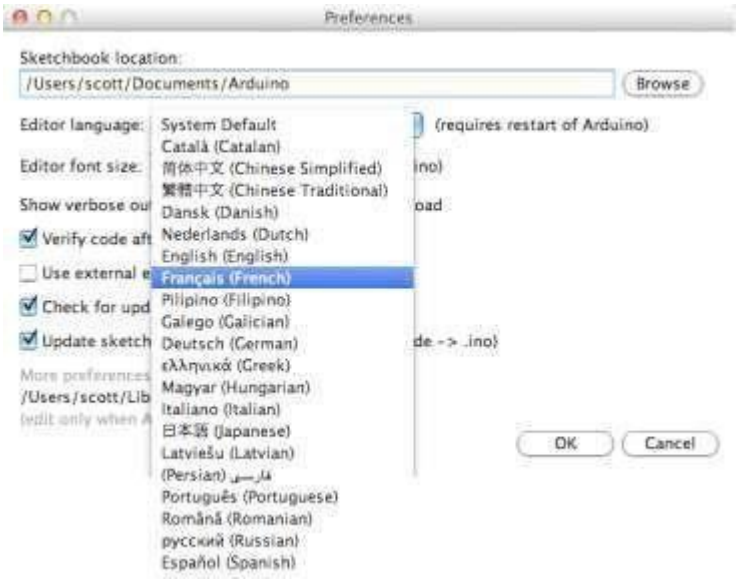
Serial Monitor

- This displays serial sent from the Arduino or Genuino board over USB or serial connector. To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down menu that matches the rate passed to Serial.begin in your sketch. Note that on Windows, Mac or Linux the board will reset (it will rerun your sketch) when you connect with the serial monitor. Please note that the Serial Monitor does not process control characters; if your sketch needs a complete management of the serial communication with control characters, you can use an external terminal program and connect it to the COM port assigned to your Arduino board.

Preferences

- Some preferences can be set in the preferences dialog (found under the Arduino menu on the Mac, or File on Windows and Linux). The rest can be found in the preferences file, whose location is shown in the preference dialog.

Language Support



- Since version 1.0.1 , the Arduino Software (IDE) has been translated into 30+ different languages. By default, the IDE loads in the language selected by your operating system. (Note: on Windows and possibly Linux, this is determined by the locale setting which controls currency and date formats, not by the language the operating system is displayed in.)
- If you would like to change the language manually, start the Arduino Software (IDE) and open the Preferences window. Next to the Editor Language there is a dropdown menu of currently supported languages. Select your preferred language from the menu, and restart the software to use the selected language.

If your operating system language is not supported, the Arduino Software (IDE) will default to English.

- You can return the software to its default setting of selecting its language based on your operating system by selecting System Default from the Editor Language drop-down. This setting will take effect when you restart the Arduino Software (IDE). Similarly, after changing your operating system's settings, you must restart the Arduino Software (IDE) to update it to the new default language.

Boards

- The board selection has two effects: it sets the parameters (e.g. CPU speed and baud rate) used when compiling and uploading sketches; and sets and the file and fuse settings used by the burn boot loader command. Some of the board definitions differ only in the latter, so even if you've been uploading successfully with a particular selection you'll want to check it before burning the boot loader. You can find a comparison table between the various boards here.

- Arduino Software (IDE) includes the built in support for the boards in the following list, all based on the AVR Core. The Boards Manager included in the standard installation allows adding support for the growing number of new boards based on different cores like Arduino Due, Arduino Zero, Edison, and Galileo and so on.
- ***Arduino Yùn:*** An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.
- ***Arduino/Genuino Uno:*** An ATmega328P running at 16 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
- ***Arduino Diecimila or Duemilanove w/ ATmega168:*** An ATmega168 running at 16 MHz with auto-reset.
- ***Arduino Nano w/ ATmega328P:*** An ATmega328P running at 16 MHz with auto-reset. Has eight analog inputs.
- ***Arduino/Genuino Mega 2560:*** An ATmega2560 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
- ***Arduino Mega:*** An ATmega1280 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
- ***Arduino Mega ADK:*** An ATmega2560 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
- ***Arduino Leonardo:*** An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.

- ***Arduino/Genuino Micro:*** An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.
- ***Arduino Esplora:*** An ATmega32u4 running at 16 MHz with auto-reset.
- Arduino Mini w/ ATmega328P: An ATmega328P running at 16 MHz with auto-reset, 8 Analog In, 14 Digital I/O and 6 PWM.
- ***Arduino Ethernet:*** Equivalent to Arduino UNO with an Ethernet shield: An ATmega328P running at 16 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
- ***Arduino Fio:*** An ATmega328P running at 8 MHz with auto-reset. Equivalent to Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328P, 6 Analog In, 14 Digital I/O and 6 PWM.
- ***Arduino BT w/ ATmega328P:*** ATmega328P running at 16 MHz. The boot loader burned (4 KB) includes codes to initialize the on-board Bluetooth module, 6 Analog In, 14 Digital I/O and 6 PWM..
- ***LilyPad Arduino USB:*** An ATmega32u4 running at 8 MHz with auto-reset, 4 Analog In, 9 Digital I/O and 4 PWM.
- ***LilyPad Arduino:*** An ATmega168 or ATmega132 running at 8 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
- ***Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328P:*** An ATmega328P running at 16 MHz with auto-reset. Equivalent to Arduino Duemilanove or Nano w/ ATmega328P; 6 Analog In, 14 Digital I/O and 6 PWM.
- ***Arduino NG or older w/ ATmega168:*** An ATmega168 running at 16 MHz without auto- reset. Compilation and upload is equivalent to Arduino Diecimila or Duemilanove w/ ATmega168, but the boot loader burned has a slower timeout (and blinks the pin 13 LED three times on reset); 6 Analog In, 14 Digital I/O and 6 PWM.
- ***Arduino Robot Control:*** An ATmega328P running at 16 MHz with auto-reset.
- ***Arduino Robot Motor:*** An ATmega328P running at 16 MHz with auto-reset.
- ***Arduino Gemma:*** An ATtiny85 running at 8 MHz with auto-reset, 1 Analog In, 3 Digital I/O and 2 PWM.

ARDUINO PROGRAMING

- Arduino programs are written in the Arduino Integrated Development Environment (IDE). Arduino IDE is special software running on your system that allows you to write sketches (synonym for program in Arduino language) for

different Arduino boards. The Arduino programming language is based on a very simple hardware programming language called processing, which is similar to the C language. After the sketch is written in the Arduino IDE, it should be uploaded on the Arduino board for execution.

- The first step in programming the Arduino board is downloading and installing the Arduino IDE. The open source Arduino IDE runs on Windows, Mac OS X, and Linux. Download the Arduino software (depending on your OS) from the official website and follow the instructions to install.

The basics of Arduino programming.

- The structure of Arduino program is pretty simple. Arduino programs have a minimum of 2 blocks,
- Preparation & Execution
- Each block has a set of statements enclosed in curly braces:

```
void setup( )
{
statements-1;
.
.
.
statement-n;
}
void loop ( )
{
statement-1;
.
.
.
statement-n;
}
```

- Here, setup ( ) is the preparation block and loop ( ) is an execution block.
- The setup function is the first to execute when the program is executed, and this function is called only once. The setup function is used to initialize the pin modes

and start serial communication. This function has to be included even if there are no statements to execute.

```
void setup ( )
{
pinMode (pin-number, OUTPUT); // set the 'pin-number' as output
pinMode (pin-number, INPUT); // set the 'pin-number' as output
}
```

- After the setup ( ) function is executed, the execution block runs next. The execution block hosts statements like reading inputs, triggering outputs, checking conditionsetc..

- In the above example loop ( ) function is a part of execution block. As the name suggests, the loop( ) function executes the set of statements (enclosed in curly braces) repeatedly.

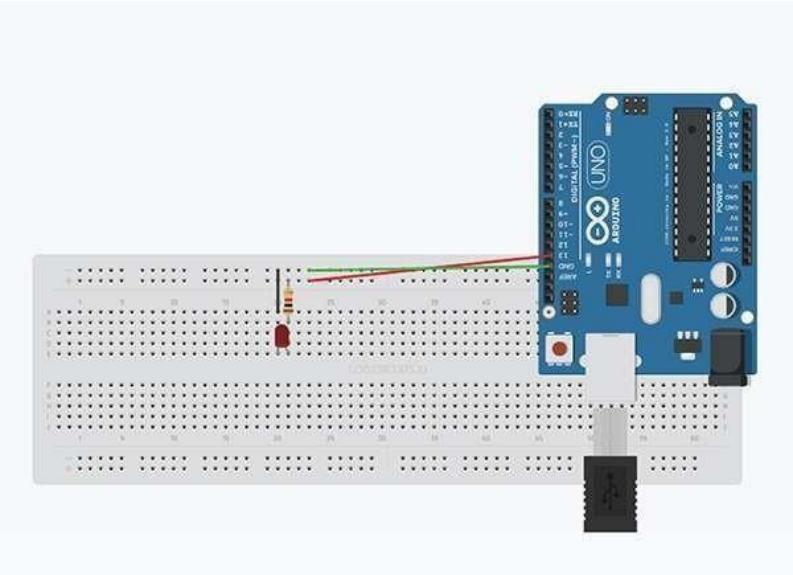
```
Void loop ( )
{
digitalWrite (pin-number,HIGH); // turns ON the component connected to 'pin-number'
delay (1000); // wait for 1 sec
digitalWrite (pin-number, LOW); // turns OFF the component connected to 'pin-number'
delay (1000); //wait for 1sec
}
```

- Note: Arduino always measures the time duration in millisecond. Therefore, whenever you mention the delay, keep it in milli seconds.
- Now, let's take a giant leap and do some experiments with Arduino
  1. Blinking the LED
  2. Fade-in and fade-out the LED
- In the process of experimenting with Arduino, writing the Arduino program is not the only important thing, building the breadboard circuit is equally important.
- Let's take a look at how the breadboard circuit has to be built for both theexperiments.
- Components required:

- Arduino UNO R3 -1
- Breadboard -1
- Breadboard connectors -3
- LED -1
- 1K resistor -1

1. Blinking LED

- Steps in building a breadboard connection:
- ***Step-1:*** Connect the Arduino to the Windows / Mac / Linux system via a USB cable
- ***Step-2:*** Connect the 13th digital pin of Arduino to the positive power rail of the breadboard and GND to the negative
- ***Step-3:*** Connect the positive power rail to the terminal strip via a 1K ohm resistor
- ***Step-4:*** Fix the LED to the ports below the resistor connection in the terminal strip
- ***Step-5:*** Close the circuit by connecting the cathode (the short chord) of the LED to the negative power strip of the breadboard



Arduino program for LED blink (Version-1)

```
int LED =13; // The digital pin to which the LED is
 connected void setup ( )
{
 pinMode (LED, OUTPUT); //Declaring pin 13 as output pin
```

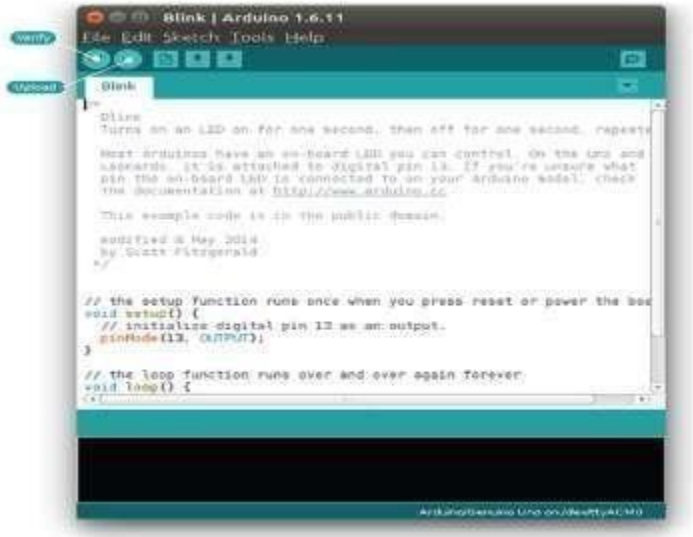
```
}
void loop( ) // The loop function runs again and again
{
digitalWrite (LED, HIGH); //Turn ON
the LED delay(1000); //Wait for 1sec
digitalRead (LED, LOW); // Turn off
the LED delay(1000); // Wait for 1sec
}
```

Arduino program for LED blink (Version-2)

```
void setup ( )
{
pinMode (13, OUTPUT); //pin 13 is set as output pin
}
void loop( ) // The loop function runs again and again
{
digitalWrite (13,HIGH); // Turn ON the LED on
pin 13 delay (1000); //Wait for 1sec
digitalWrite (13, LOW); //Turn OFF the LED on pin 13
}
```

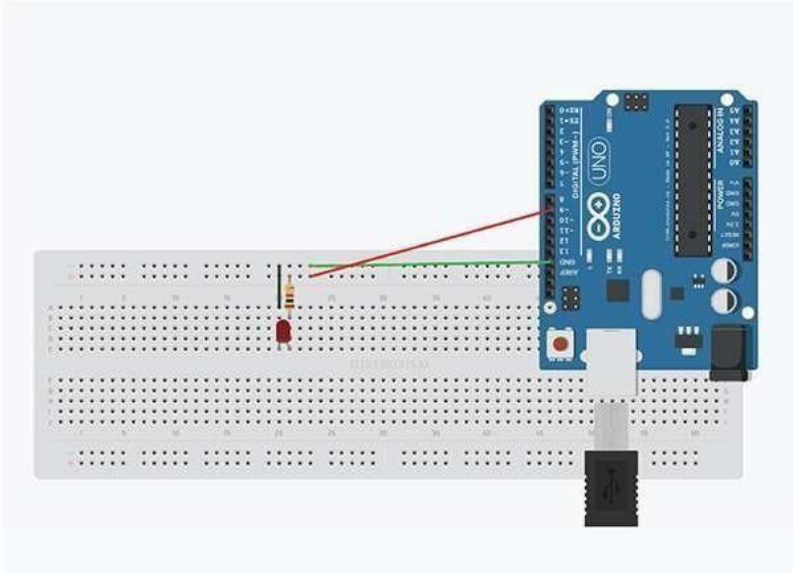
- In version-1 of the LED blink program LED is declared globally and is set to pin number
  13. This will reduce the number of iterations required to update the pin number in the program when you connect the LED to the other digital pin. Whereas, the pin number has to be changed in 3 different statements in version-2.

- The magic happens when you click the upload icon in the Arduino IDE. The program will be uploaded into the microcontroller of Arduino board and LED in the circuit starts blinking.



2. Fade-in and fade-out the LED

- The step for the LED bling experiment are the same as those followed for building the breadboard circuit except that in step-2, you connect the 9th digital pin to the positive power rail of the breadboard.



<u>Arduino program for LED fade-in and fade-out</u>

<u>(Version-1)</u> **int led = 9; // The digital pin to which the**

LED is connected int brightness = 0; // Brightness

```
of LED is initially set to 0 int fade = 5; // By how
many points the LED should fade void setup()
{
pinMode(led, OUTPUT); //pin 10 is set as output pin
}
void loop() // The loop function runs again and again
{
analogWrite(led, brightness); // set the brightness of LED
brightness = brightness + fade; //Increase the brightness of LED
by 5 points if (brightness <= 0 || brightness >= 255) // check the
level of brightness
 {
 fade = -fade;
 }
delay(30); // Wait for 30 milliseconds
}
```

Arduino program for LED fade-in and fade-out (Version-2)

```
int led=19; // The digital pin to which
the LED is connected void setup()
{
pinMode(led, OUTPUT); //pin 10 is set as output pin
}
 void loop() // The loop function runs again and again
{
 for (int fade=0; fade<=255; fade=fade+5)
{
 analogWrite (led, fade); // Change the brightness of LED by
5 points delay (30);
 }
}
```

- In both the versions of the LED fade-in and fade-out programs, the analog Write statement is used for a led connected to a digital pin. The reason for this is that, digital pin 10 is a PWM pin. As discussed in the previous article, A tour of the Arduino UNO board, a PWM pin is capable of generating Analog output. In both the programs pin 10 is used as analog output pin.

VARIOUS REAL TIME APPLICATIONS OF IOT

Overview

- Understand the varied applications of the Internet of Things (IoT)
- Includes detailed study about smart homes, connected devices & many other work areas

Introduction

- Do you know what separates humans from other living beings?
  - Curiosity. Humans are curious. We question a lot. We are the ones who challenge the status quo of existing rules and strive to build/produce something better. Such curiosity & efforts have promised us a life where electronic devices & machines will probably become our best friend.
- Yes, you read it correctly the vision to make machines smart enough to reduce human labor to almost nil. The idea of inter-connected devices where the devices are smart enough to share information with us, to cloud based applications and to each other (device to device).
- Smart devices or "Connected devices" as they are commonly called, are designed in such a way that they capture and utilize every bit of data which you share or use in everyday life. And these devices will use this data to interact with you on daily basis and complete tasks.

1. Smart Home

- With IoT creating the buzz, 'Smart Home' is the most searched IoT associated feature on Google. But, what is a Smart Home?
- Wouldn't you love if you could switch on air conditioning before reaching home or switch off lights even after you have left home? Or unlock the doors to friends for temporary access even when you are not at home. Don't be surprised with IoT taking shape companies are building products to make your life simpler and convenient.
- Smart Home has become the revolutionary ladder of success in the residential spaces and it is predicted Smart homes will become as common as smart phones.

- The cost of owning a house is the biggest expense in a homeowner's life. Smart Home products are promised to save time, energy and money. With Smart home companies like Nest, Ecobee, Ring and August, to name a few, will become household brands and are planning to deliver a never seen before experience.

2. Wearable's

- Wearables have experienced an explosive demand in markets all over the world. Companies like Google, Samsung have invested heavily in building such devices. But, how do they work?
- Wearable devices are installed with sensors and software's which collect data and information about the users. This data is later pre-processed to extract essential insights about user.
- These devices broadly cover fitness, health and entertainment requirements. The pre- requisite from internet of things technology for wearable applications is to be highly energy efficient or ultra-low power and small sized.

3. Connected Cars

- The automotive digital technology has focused on optimizing vehicles internal functions. But now, this attention is growing towards enhancing the in-car experience.
- A connected car is a vehicle which is able to optimize its own operation, maintenance as well as comfort of passengers using onboard sensors and internet connectivity.
- Most large auto makers as well as some brave startups are working on connected car solutions. Major brands like Tesla, BMW, Apple, and Google are working on bringing the next revolution in automobiles.

4. Industrial Internet

- Industrial Internet is the new buzz in the industrial sector, also termed as Industrial Internet of Things (IIoT). It is empowering industrial engineering with sensors, software and big data analytics to create brilliant machines.
- According to Jeff Immelt, CEO, GE Electric, IIoT is a "beautiful, desirable and investable" asset. The driving philosophy behind IIoT is that, smart machines are more accurate and consistent than humans in communicating through data. And, this data can help companies pick inefficiencies and problems sooner.
- IIoT holds great potential for quality control and sustainability. Applications for tracking goods, real time information exchange about inventory among suppliers and retailers and automated delivery will increase the supply chain efficiency.

According to GE the improvement industry productivity will generate $10 trillion to $15 trillion in GDP worldwide over next 15 years.

5. Smart Cities

- Smart city is another powerful application of IoT generating curiosity among world's population. Smart surveillance, automated transportation, smarter energy management systems, water distribution, urban security and environmental monitoring all are examples of internet of things applications for smart cities.
- IoT will solve major problems faced by the people living in cities like pollution, traffic congestion and shortage of energy supplies etc. Products like cellular communication enabled Smart Belly trash will send alerts to municipal services when a bin needs to be emptied.
- By installing sensors and using web applications, citizens can find free available parking slots across the city. Also, the sensors can detect meter tampering issues, general malfunctions and any installation issues in the electricity system.

6. IoT in agriculture

- With the continuous increase in world's population, demand for food supply is extremely raised. Governments are helping farmers to use advanced techniques and research to increase food production. Smart farming is one of the fastest growing fields in IoT.
- Farmers are using meaningful insights from the data to yield better return on investment. Sensing for soil moisture and nutrients, controlling water usage for plant growth and determining custom fertilizer are some simple uses of IoT.

7. Smart Retail

- The potential of IoT in the retail sector is enormous. IoT provides an opportunity to retailers to connect with the customers to enhance the in-store experience.
- Smartphone's will be the way for retailers to remain connected with their consumers even out of store. Interacting through Smartphone's and using Beacon technology can help retailers serve their consumers better. They can also track consumer's path through a store and improve store layout and place premium products in high traffic areas.

8. Energy Engagement

- Power grids of the future will not only be smart enough but also highly reliable. Smart grid concept is becoming very popular all over world.
- The basic idea behind the smart grids is to collect data in an automated fashion and analyze the behavior or electricity consumers and suppliers for improving

efficiency as well as economics of electricity use.

- Smart Grids will also be able to detect sources of power outages more quickly and at individual household levels like nearby solar panel, making possible distributed energy system.

9. IOT in Healthcare

- Connected healthcare yet remains the sleeping giant of the Internet of Things applications. The concept of connected healthcare system and smart medical devices bears enormous potential not just for companies, but also for the well-being of people in general.
- Research shows IoT in healthcare will be massive in coming years. IoT in healthcare is aimed at empowering people to live healthier life by wearing connected devices.
- The collected data will help in personalized analysis of an individual's health and provide tailor made strategies to combat illness. The video below explains how IoT can revolutionize treatment and medical help.

10. IoT in Poultry and Farming

- Livestock monitoring is about animal husbandry and cost saving. Using IoT applications to gather data about the health and well being of the cattle, ranchers knowing early about the sick animal can pull out and help prevent large number of sick cattle.
- With the help of the collected data and ranchers can increase the poultry production. Watch this interesting video.

CONNECTING IOT TO CLOUD

- The Internet of Things (IoT) will continue to transform the business landscape as well as the way we live. Cloud computing is the backbone of this transformation. Increased cloud adoption has acted as a springboard for many IoT applications and business models, offering the ability for companies to reduce time-to-market and total cost of ownership.

Cloud and Connectivity Synergy for Enterprise IoT

- Cloud computing allows companies to store and manage data over cloud platforms, providing scalability in the delivery of applications and Software as a Service (SaaS). IoT devices can generate a significant amount of data per second, with Cisco estimating that IoT will generate 847 zettabytes per year by 2021.
- IoT devices are often sensors that collect data and send it to be processed. In

the domain of IoT, physical sensors are virtualized before the data is uploaded to the cloud. While IoT devices can generate lots of data, cloud computing paves the way for this data to travel. This data enables better workflow for developers, who can store and access data remotely, which allows them to implement projects without delay.

- Cloud computing enables the storage and analysis of data to be done quickly and in real- time, allowing enterprises to get the maximum benefit. This is supported by an industry survey from InformationWeek where 65% of respondents said "the ability to quickly meet business demands" was one of the most important reasons a business should move to a cloud environment.
- With cloud computing, organizations do not have to deploy extensive hardware or configure and manage networks and infrastructure. Cloud computing also enables enterprises to scale up the infrastructure, depending on their needs, without setting up additional hardware and infrastructure. This not only helps speed up the development

  process but can also cut down on development costs. Half of all CIOs and IT leaders surveyed by the cloud-security company Bitglass, reported cost savings from using cloud-based applications.
- Devices are connected to the cloud through many different methods depending on the device connectivity capabilities. These methods include cellular, satellite, Wi-Fi, Low Power Wide Area Networks (LPWAN) (e.g. NB-IoT), and direct connection to the Internet via Ethernet. Cellular connectivity offers an excellent choice for uninterrupted data transfer between devices, applications, and the cloud.

Cellular Connectivity: The Reliable and Secure Choice for IoT Projects

- Cellular technologies have been designed for reliability, security, and scalability. Cellular Internet of Things (CIoT), based on 3GPP standards, utilizes existing infrastructure to provide excellent IoT coverage and a fast time-to-market.
- Cellular IoT projects will include everything from devices, to a dedicated SIM card and management platform. CIoT hardware is always equipped with SIM cards and can connect to networks via 2G, 3G, or LTE connectivity, depending on which cell towers are available in different regions. For example, 2G switch-off has already begun in many parts of the world, as detailed on our previous blog post.

Cellular benefits include:

- ***Coverage:*** Cellular data coverage today is extensive and growing and has the added benefit of reaching underground spaces, buildings, and rural

environments.

- ***Security:*** End-to-end security is supported and aided by the SIM credentials. This keeps data secure from the device to the cloud.
- ***Bandwidth:*** Cellular technologies based on 4G LTE are now as fast as 1 Gbps, with 5G expected to have speeds of up to 10 Gbps.
- For example, fleet tracking relies on secure coverage across boundaries with no risk of losing connection. To ensure this doesn't happen, cellular connectivity is better equipped than alternative connectivity methods, e.g. NB-IoT is designed for low throughput devices where a delay in communications will not hinder the service. In fleet management, with vehicles driving across borders, it's essential to have a robust and constant connection so that devices can be tracked in real time.

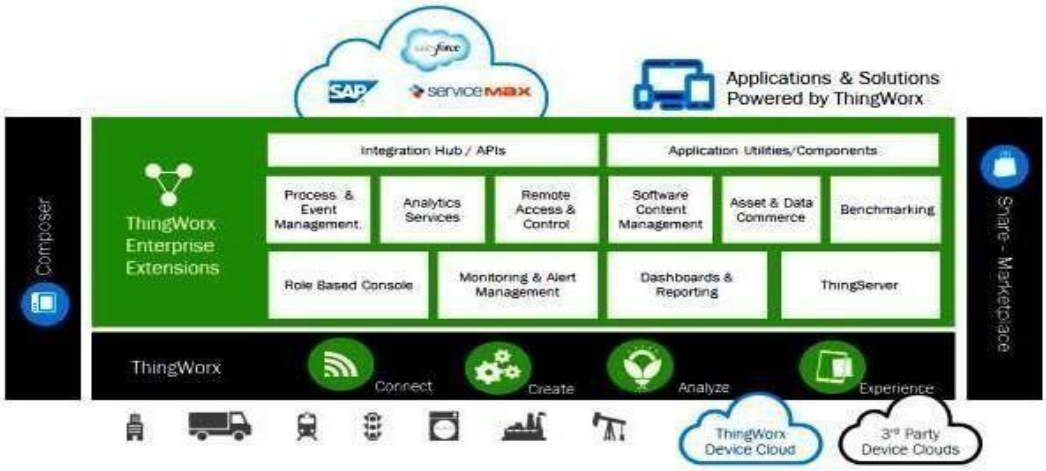
EMnify – Bringing the Cloud and Cellular Connectivity Together

- EMnify is an IoT connectivity provider that enables cellular IoT products and services worldwide. The company serves the growing needs of developers and enterprises that require secure, global connectivity for IoT/M2M applications. Its multi-network EMnify SIM can operate instantly in over 180 countries across 540 mobile networks.
- EMnify connected devices benefit from a purpose-built and fully virtualized mobile core network that enables features like virtual private device cloud and dynamic regional internet breakouts. The EMnify management platform gives an oversight of IoT device connectivity as well as access to analytics with an easy to use and intuitive UI. The EMnify SIM is multi-carrier meaning it will automatically connect to the best cellular network and ensure the connection remains stable and continuously connected all around the world.
- In August 2018, EMnify achieved the AWS IoT Competency which differentiates EMnify as an AWS Partner Network (APN) member that has demonstrated relevant technical proficiency and proven customer success delivering solutions seamlessly in the AWS Cloud environment. Learn more about the successful transition to a Cloud Mobile Network and its benefits in the EMnify case study prepared for Telecom Liechtenstein.

CLOUD STORAGE FOR IOT

1. Thingworx 8 IoT Platform

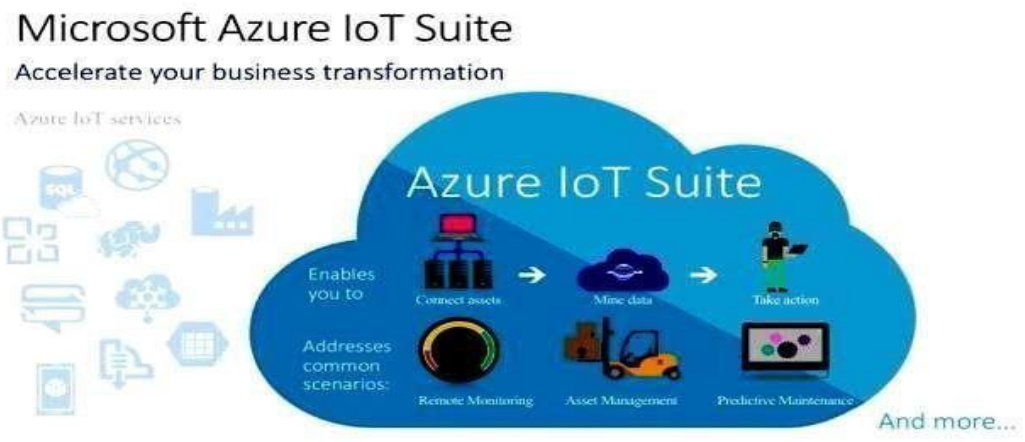
- Thingworx is one of the leading IoT platforms for industrial companies, which provides easy connectivity for devices. It enables the experience from today's connected world. Thingworx 8 is a better, faster, easier platform, providing the functionality to build, deploy, and extend industrial projects and apps.



- Thingworx is an IoT platform designed by PTC for development of enterprise app development. It offers basic features, such as:
  - Easy connectivity with electronic devices, like sensors and RFIDs
  - You can work remotely once you are done with the setup
  - Pre-built widgets for the dashboard
  - Remove Complexity of the project
  - Integrated machine learning
- **Pros**
  - Easy web page designs for customers
  - Easy to manage devices
  - Simple connectivity solutions
- **Cons**
  - Difficult to use with custom programs in C#
  - Hard to manage complex systems.
  - The limitation to install edge program on a custom platform.

2. *Microsoft Azure IoT Suite*

- Microsoft Azure provides multiple services to create IoT solutions. It enhances your profitability and productivity with pre-built connected solutions. It analyzes untapped data to transform business. This provides the solutions for a small PoC to Rolling out your ideas. Azure Suite can easily analyze and act on new data.



- Azure IoT Suite provides features like:
  - Easy Device Registry.
  - Rich Integration with SAP, Salesforce, Oracle, WebSphere, etc
  - Dashboards and visualization
  - Real-time streaming

- **Pros**
  - Offers third-party services
  - Secure and scalable
  - High availability
- **Cons**
  - Requires management
  - Expensive
  - No support for bugs

3. *Google Cloud's IoT Platform*

- Google's platform is among the best platforms we currently have. Google has an end-to- end platform for Internet-of-Things solutions. It allows you to easily connect, store, and manage IoT data. This platform helps you to scale your business.

- Their main focus is on making things easy and fast. Pricing on Google Cloud is done on a per-minute basis, which is cheaper than other platforms.
- Google Cloud's IoT platform provides features, including:
  - Provides huge storage
  - Cuts cost for server maintenance
  - Business through a fully protected, intelligent, and responsive IoT data
  - Efficient and scalable
  - Analyze big data
- **Pros**
  - Fastest input/output
  - Lesser access time
  - Provides integration with other Google services
- **Cons**
  - Most of the components are Google technologies
  - Limited programming language choices

4. IBM Watson IoT Platform

- IBM Watson is a powerful platform backed by IBM 's the Bluemix and hybrid cloud PaaS (platform as a service) development platform. By providing easy sample apps and interfaces for IoT services, they make it accessible to beginners. You can easily try out their sample to see how it works, which makes

it stand out from otherplatforms.

- Users can get the following features:
  - Real-time data exchange
  - Secure Communication
  - Cognitive systems
  - Recently added data sensor and weather data service
- **Pros**
  - Process untapped data
  - Handle huge quantities of data
  - Improve customer services
- **Cons**
  - Need a lot of maintenance.
  - Take time for Watson integration
  - High switching cost.

5. AWS IoT Platform

- Amazon made it much easier for developers to collect data from sensors and Internet- connected devices. They help you collect and send data to the cloud and analyze that information to provide the ability to manage devices.
- You can easily interact with your application with the devices even they are offline.

- Main features of the AWS IoT platform are:
  - Device management
  - Secure gateway for devices
  - Authentication and encryption

- Device shadow

- **Pros**
  - Good integration with laas offering.
  - Price dropped over the last six years
  - Open and flexible
- **Cons**
  - A big learning curve for AWS
  - Three outages in the last 2 years
  - Not secure for hosting critical enterprise applications

6. Cisco IoT Cloud Connect

- Cisco Internet of Things accelerates digital transformation and actions from your data. Cisco IoT Cloud Connect is a mobile, cloud-

based suite. It offers solutions for mobile operators to provide phenomenal IoT experience. It provides flexible deployment options for your devices.

- The main feature of the Cisco Cloud Connect:
  - Data and voice connectivity
  - Device and IP session report
  - Billing is customizable
  - Flexible deployment options

7. Salesforce IoT Cloud

- Salesforce IoT Cloud is powered by Salesforce Thunder. It gathers data from devices, websites, applications, and partners to trigger actions for real-time responses. Salesforce combined with IoT delivers improved customer service.

- Key features of Salesforce IoT Cloud:
  - Enhanced data collection
  - Improved customer engagement
  - Real-time event processing
  - Technology optimization
- **Pros**
  - Scale to billions of devices and messages.
  - Easy UI designs to connect with customers.
- **Cons**
  - Security liability
  - Flexibility limitations

8. Kaa IoT Platform

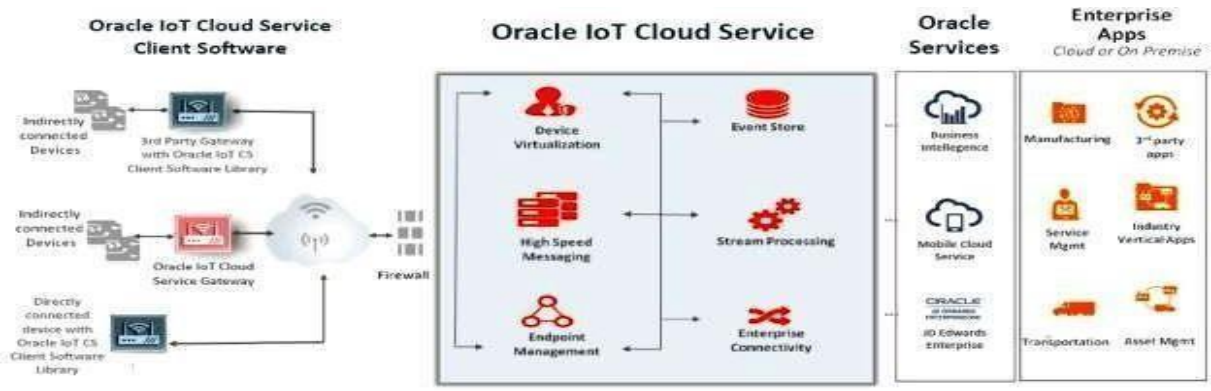
- Kaa is an open-source, multipurpose, middleware platform for complete end-to-end IoT development and smart devices. It reduces cost, risk, and market time. Also, Kaa offers a range of IoT tools that can be easily plugged in and

implemented in IoT use cases.

- It provides many features that make it unique, such as:
  - Reduce development time
  - Open source and free
  - Easy and direct device implementation
  - Reduce marketing time
  - Handle millions of devices
- **Pros**
  - Ease of use
  - Third-party integration
  - Data security
- **Cons**
  - Not able to deploy applications based on the PaaS model

9. Oracle IoT Platform

- Oracle offers real-time Internet of Things data analysis, endpoint management, and high speed messaging where the user can get real-time notification directly on their devices. Oracle IoT cloud service is a Platform as a Service (PaaS), cloud-based offering that helps you to make critical business decisions.

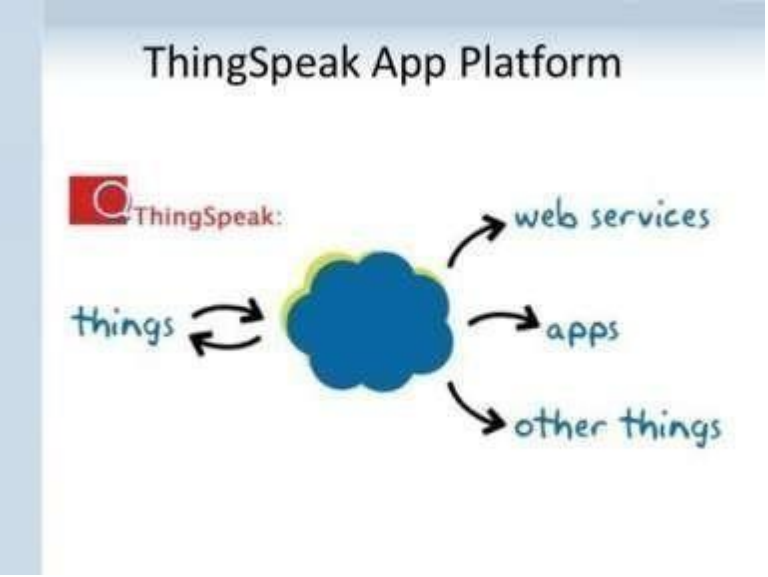


- Features offering to users
  - Secure and scalable
  - Real-time insight
  - Integrated
  - Faster to market
- **Pros**

- Device visualization
- High speed messaging
- Event store

10. Thingspeak IoT Platform

- Thingspeak is an open-source platform that allows you to collect and store sensordata to the cloud. It provides you the app to analyze and visualize your data in Matlab. You can use Arduino, Raspberry Pi, and Beaglebone to send sensor data. You can create a separate channel to store data.



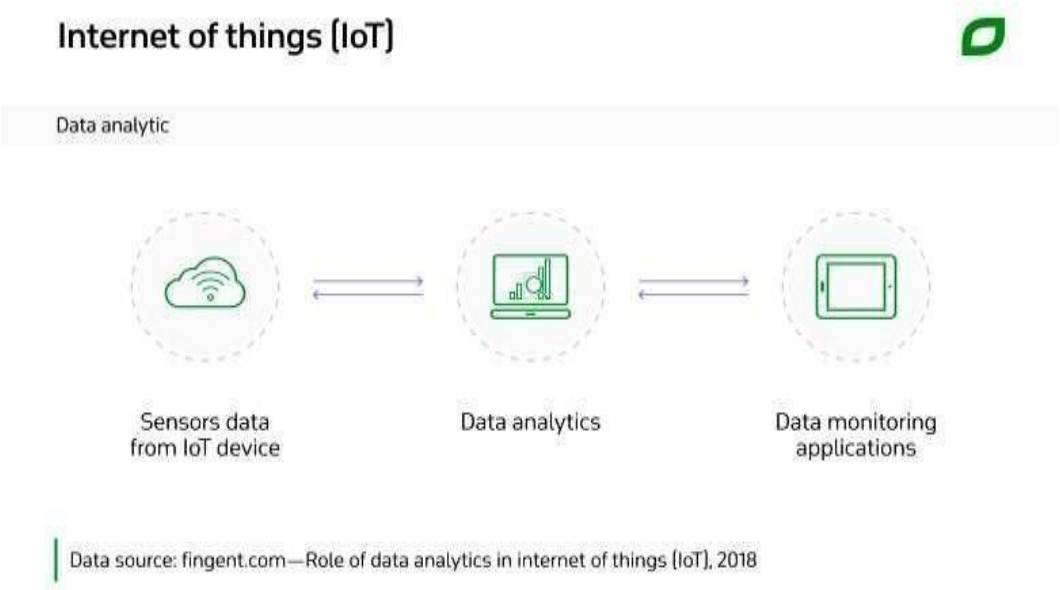
- Features of Thingspeak:
  - Collect data in private channels
  - App integration
  - Event scheduling
  - MATLAB analytics and visualization
- **Pros**
  - Free hosting for channels
  - Easy visualization
  - Provides additional features for Ruby, Node.js, and Python
- **Cons**
  - Limited data uploading for API
  - ThingSpeak API can be a hurdle for beginners

11. GE Predix IoT Platform

- Predix is the world's first industrial platform. Predix was designed to target factories and provides simple ecosystem. It can directly analyze data from the machine and store. GE wants to provide the growing industrial Internet of Things for its cloud platform. This platform is secure and scalable.

- According to the customers their IoT platform can:
  - Optimize assets and operations
  - Provides key performance data
  - Reduces unplanned downtime
  - Real-time operational data

DATA ANALYTICS FOR IOT



- Simply put, IoT data analytics is the analysis of huge data volumes generated by connected devices. Organizations can derive a number of benefits from it: optimize operations, control processes automatically, engage more customers, and empower employees. The combination of IoT and data analytics has already proven to be beneficial in retail, healthcare, telematics, manufacturing, and smart cities. However, its true value for organizations has yet to be fully realized.
- At least a decade ago, it was exceedingly difficult and expensive to analyze massive volumes of information provided by a variety of connected devices. As time goes on, the cost to store data is going down, and analytics capabilities are making huge leaps forward. This creates favorable conditions for organizations to start investing in and implementing IoT data analytics.
- As the accessibility of IoT data analytics grows, more and more organizations are seeing the benefits of having it. Widely-known companies such as Microsoft, GE, Amazon, SAP, and Salesforce have already started implementing IoT data analytics into their day-to- day processes.

The value of IoT data for organizations

- A huge variety of devices connect to the internet and share data through sensors every day. This data is worthless without analysis. However, with an IoT analytics solution put in place, the data that organizations produce is effectively collected, analyzed, and stored. As a result, it allows organizations to optimize their operations at all levels, improve decision making, and achieve a number of benefits.

Better human productivity

- Some organizations install smart sensors throughout their facilities to collect data on employee engagement, performance ratings, and other work-related activities. This data is later used to improve day-to-day business operations and help utilize employee time and energy more efficiently.
- As an example, Humanyze has developed a system that uses smart employee badges with sensors that capture more than 100 data points to measure the productivity of workers. It analyzes how people interact, how much they gesture or listen, and what their tone of voice is. At one point, this data helped discover that bringing call-center workers together for lunch at the same time instead of staggering them so that somebody is always available to answer the phone, significantly improved productivity due to:
    - Job stress level decreased by 19%
    - Productivity increased by 23%

- Retention rate increased by 28%

- If IoT data analytics is evaluated and implemented properly, it can have a positive impact on employees' productivity and overall business success. However, there is a privacy concern: Accenture reports that only 32% of employees consent to their company using their workplace data, and 55% of businesses don't ask for consent.

Improved Equipment Maintenance

- The combination of IoT sensors and data analytics may help companies, especially in the manufacturing industry, to determine when equipment requires maintenance by measuring vibration, heat, and other important figures. Smart equipment can also send messages to operators about potential breakdowns, wear, and delivery schedules. This not only facilitates regular equipment maintenance but also contributes to predictive maintenance.

- Sensor data is used to predict when assets need to be serviced, which allows maintenance to be scheduled at the optimal time, thus reducing breakdowns and saving maintenance costs.

- IoT allows workers to see exactly how their machines are performing in real-time, and alerts them to any issues that might be arising. Being able to prevent unscheduled downtime by using predictive maintenance can provide significant benefits. According to PwC's Predictive Maintenance 4.0 study, utilizing predictive maintenance can reduce costs by 12%, improve uptime by 9%, and reduce safety, health, environment, and quality risks by 14% on average. Additionally, it can extend the lifetime of an aging asset by 20%.

Operations optimization and automation

- With IoT and analytics working in tandem, organizations can automatically control processes that previously could only be tracked manually. For example, it gives manufacturers a comprehensive view of what's going on at every point in production. This allows them to maintain a continuous flow of final products, identify bottlenecks in real-time, and avoid defects. It also reduces the risk of human error.

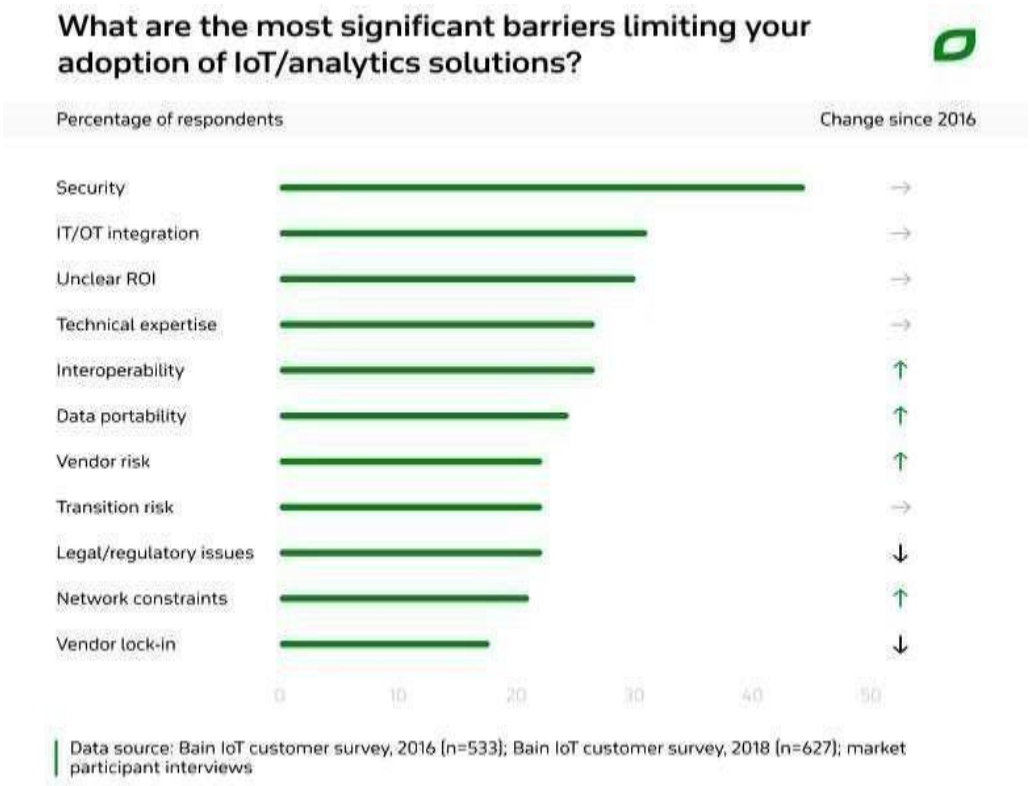
- General Motors, for instance, monitors humidity with IoT sensors to optimize painting. When conditions for painting are unfavorable, they send the detail to another place with more appropriate humidity levels, thereby reducing downtime and the need for repainting.

- When another major manufacturing company, Bosch Rexroth, started utilizing IoT, they saw a significant increase in productivity of both workers and equipment.

Enhanced customer experience

- Be it a retail shop or a healthcare center, each organization strives to create a better and more personalized customer experience. The implementation of IoT data analytics can help with this onerous task. IoT data reveals a wealth of customer behaviors and preferences that can be analyzed and used to predict customer needs.
- For example, when a customer enters a store, the IoT data analytics system can guide the shopper to the jeans they have been looking at online. It can also send them a personalized coupon to make the purchase in-store that day. In addition, retailers, restaurant chains, and makers of consumer goods can use IoT data to do targeted marketing and promotions.
- In healthcare, hospitals can use real-time IoT data analytics to manage increased patient traffic and improve general operational efficiency.
- This is beneficial to both parties; consumers gain more value through convenience and time saving, while organizations increase their revenue and stay more attractive to customers.

Challenges and barriers



- IoT data analytics can certainly provide benefits in many business areas, like reducing maintenance costs, avoiding equipment failures, and improving customer experiences and human productivity. But despite the proven benefits of applying IoT data, many organizations don't derive value from their data assets simply because they don't know what to do with the technology.
- Also, sometimes employees get overwhelmed with the amount of data coming in, preventing them from sifting through it all and finding efficient ways of utilizing it. With multiple sensors providing information as frequently as every 30 seconds, it could result in data overload for the workers or the computers, if the right ones aren't beingused.
- Other possible barriers to IoT data analytics adoption include security issues and high costs. Since it requires multiple machines and devices working together and exchanging information, if one system is subjected to a security breach, it could spread throughout all systems.
- Additionally, because IoT data analytics is still fairly new, the implementation costs can be high. This can deter business owners from adopting it, especially if it is difficult to see the long-term benefits of making that investment.

IoT data analytics on the rise, still

- At a time when organizations are looking to gain a competitive edge, many are starting to look toward the internet of things. It offers an opportunity to gather even more information than before, which can improve processes, drive innovation, and enhance customer experience. While it hasn't reached every corner of business and operations yet, it is already being used to make workplaces safer and more efficient.
- Having the ability to receive diagnostic and predictive data in real-time can be a game- changer, especially for companies who rely heavily on their equipment working properly and at full capacity. IoT data analytics allows workers to schedule downtimes that are convenient or know exactly what is wrong without having to bring in a technician to examine the machine. That all adds up to a less expensive and more efficientoperation.
- IoT data analytics can also be used to make customers happier and enhance their shopping experiences. By gathering data on the preferences of their audiences, companies can give customers what they are looking for and even know how much they are willing to pay for it. It can also gather useful data from product reviews, giving companies a better idea of what their customers are looking for or would like to see done differently.
- Even though there are still significant challenges to overcome, IoT data analytics continues to grow and gain popularity. It has the ability to offer

unprecedented insights that have never before been so readily available. It is reasonable to think that we will see a time in the near future where the general consensus is that its pros outweigh its cons.

SOFTWARE & MANAGEMENT TOOLS FOR IOT

- IoT platforms and tools are considered as the most significant component of the IoT ecosystem. Any IoT device permits to connect to other IoT devices and applications to pass on information using standard Internet protocols. IoT platforms fill the gap between the device sensors and data networks. IoT platforms connect the data to the sensor system and give insights using back-end applications to create a sense of the plenty of data developed by the many sensors.
- The Internet of Things (IoT) is the future of technology that helps the Artificial intelligence (AI) to regulate and understand the things in a considerably stronger way.
- We have picked up a mix of best known IoT platforms and tools that help you todevelop the IoT projects in an organized way.

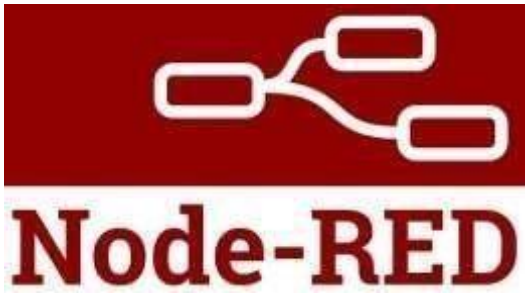
1. Zetta

- 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, Web Sockets 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.

2. *Arduino*

- 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).

3. *Node-RED*



- 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

- Flutter is a programmable processor core for electronics projects, designed for students, and engineers. Flutter's take to glory is its 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.

4. 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.

5. ThingsBoard

- 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.

6. Kinoma

- Kinoma, a Marvell Semiconductor hardware prototyping platform, involves three different open source projects. Kimona Create is a DIY construction kit for prototyping electronic devices. Kimona Studio is the development environment that functions with Set up and the Kinoma Platform Runtime. Kimona Connect is a free iOS and Android app that links smart phones and stands with IoT devices.

7. Kaa IoT Platform

- 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 micro services 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

8. *Site Where*

- Site Where 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 Site Where 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 Site Where data

9. *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.

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 hacker's boards such as Raspberry Pi, Intel Edison, and ESP8266.
- Thinger can be integrated with IFTT, and it provides real-time data on a beautiful dashboard.