ESP-MESH

Painlessmesh

Table of contents

01 Pengenalan

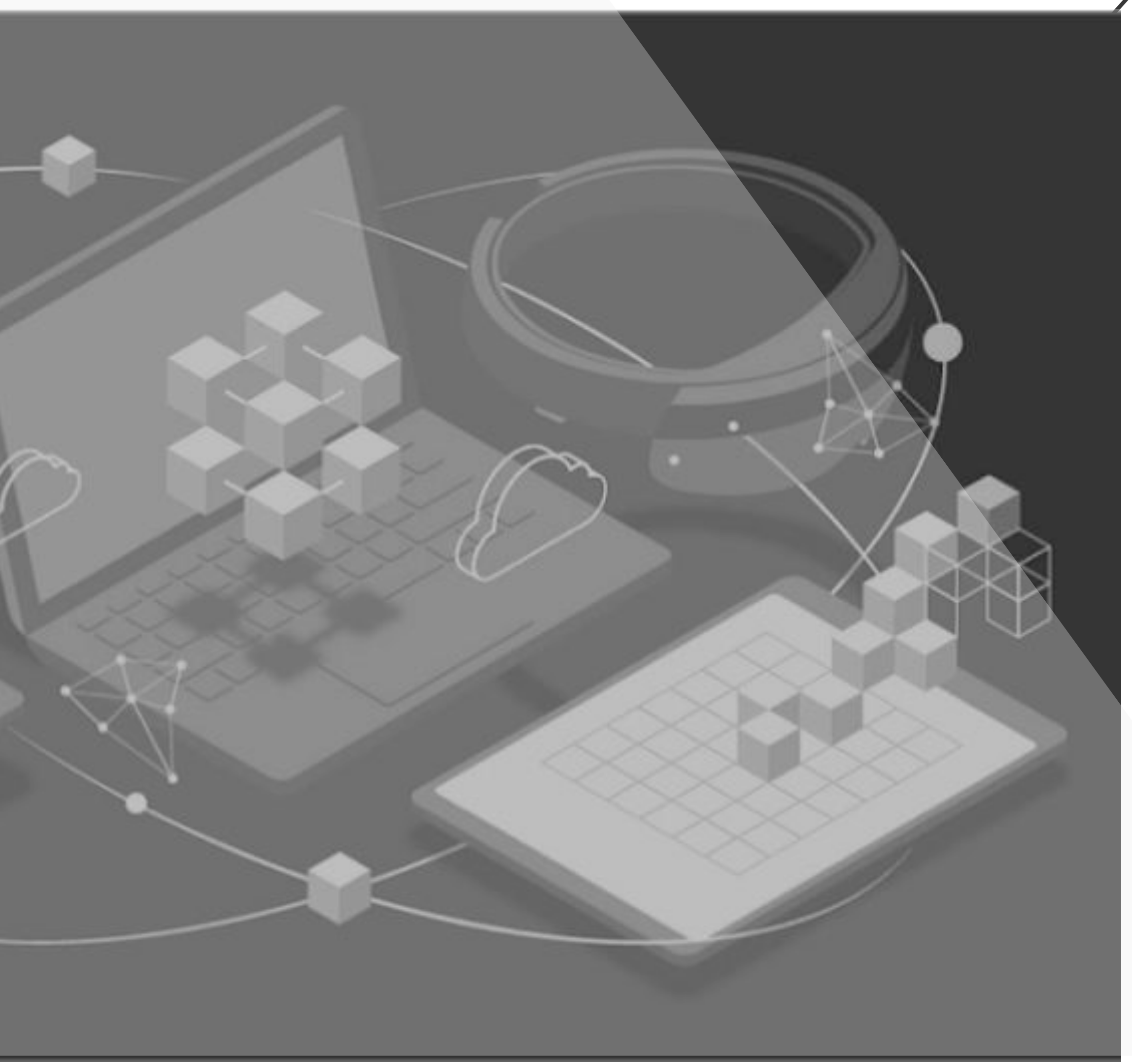
Pengenalan

02 PAINLESSMESH CONNECTION

Memulai dengan ESPNOW ESP32

03 PAINLESSMESH

Koneksi Esp32 ke dengan mesh



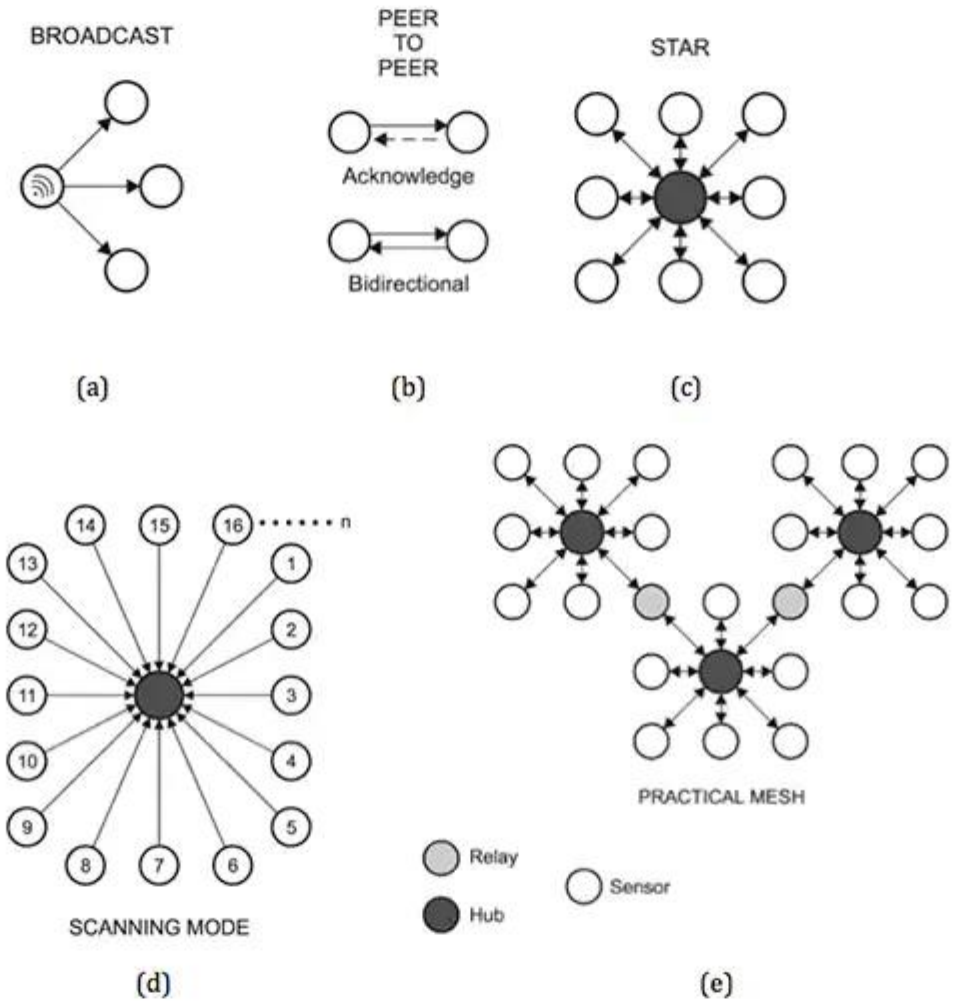
01

Pengenalan

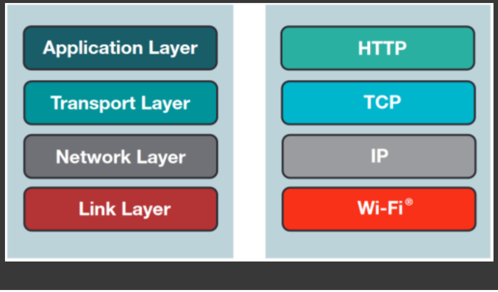
GOAL

- Memahami sistem Mesh dan metode koneksinya
- Membuat program sensor dengan jaringan Mesh
- Melakukan field test

Network topology



COMMUNICATION PROTOCOL



WiFi™

Bluetooth®

ZigBee®

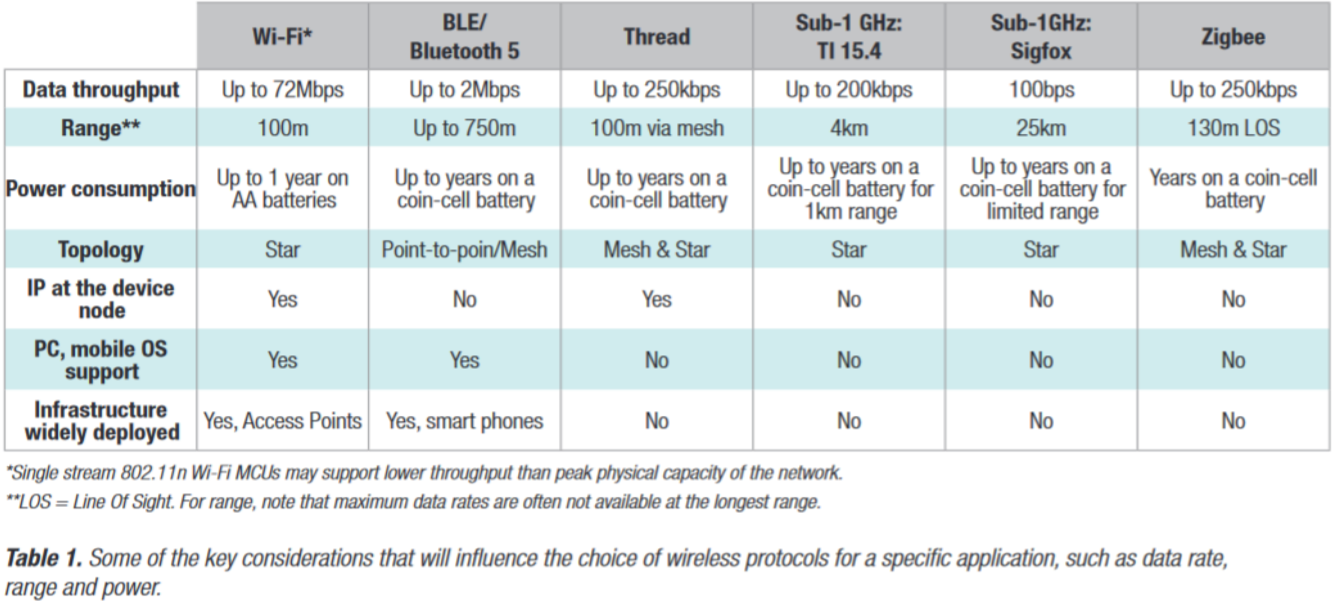
Sub-1GHz

LoWPAN

sigfox

THREAD

WirelessHART®



	Wi-Fi*	BLE/ Bluetooth 5	Thread	Sub-1 GHz: TI 15.4	Sub-1GHz: Sigfox	Zigbee
Data throughput	Up to 72Mbps	Up to 2Mbps	Up to 250kbps	Up to 200kbps	100bps	Up to 250kbps
Range**	100m	Up to 750m	100m via mesh	4km	25km	130m LOS
Power consumption	Up to 1 year on AA batteries	Up to years on a coin-cell battery	Up to years on a coin-cell battery	Up to years on a coin-cell battery for 1km range	Up to years on a coin-cell battery for limited range	Years on a coin-cell battery
Topology	Star	Point-to-poin/Mesh	Mesh & Star	Star	Star	Mesh & Star
IP at the device node	Yes	No	Yes	No	No	No
PC, mobile OS support	Yes	Yes	No	No	No	No
Infrastructure widely deployed	Yes, Access Points	Yes, smart phones	No	No	No	No

*Single stream 802.11n Wi-Fi MCUs may support lower throughput than peak physical capacity of the network.

**LOS = Line Of Sight. For range, note that maximum data rates are often not available at the longest range.

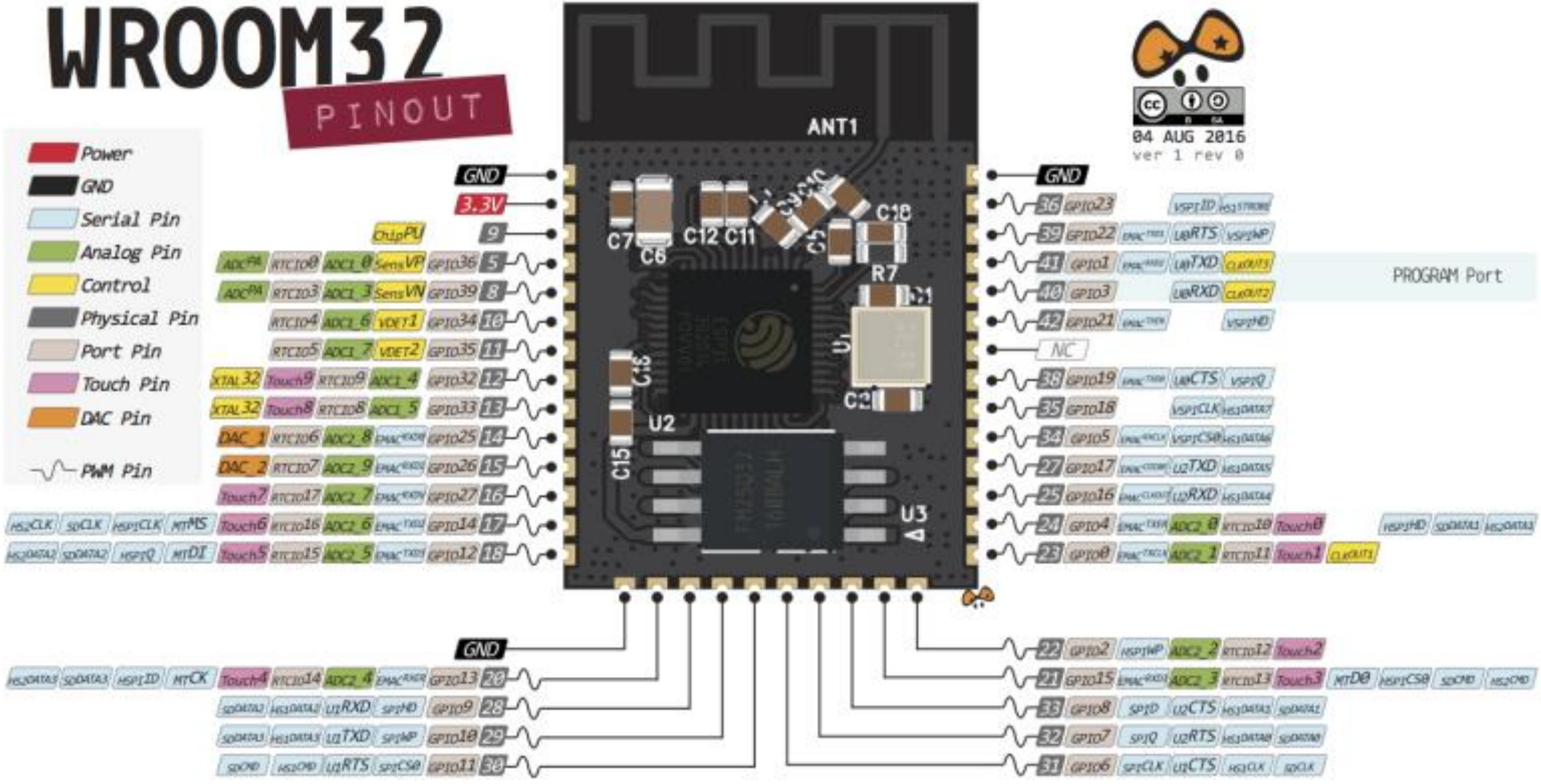
Table 1. *Some of the key considerations that will influence the choice of wireless protocols for a specific application, such as data rate, range and power.*

Wireless connectivity for the Internet of Things: One size does not fit all", Nick Lethaby, Texas instruments, October 2017

Mesh connection

Tentang PainlessMesh

WROOM32 PINOUT

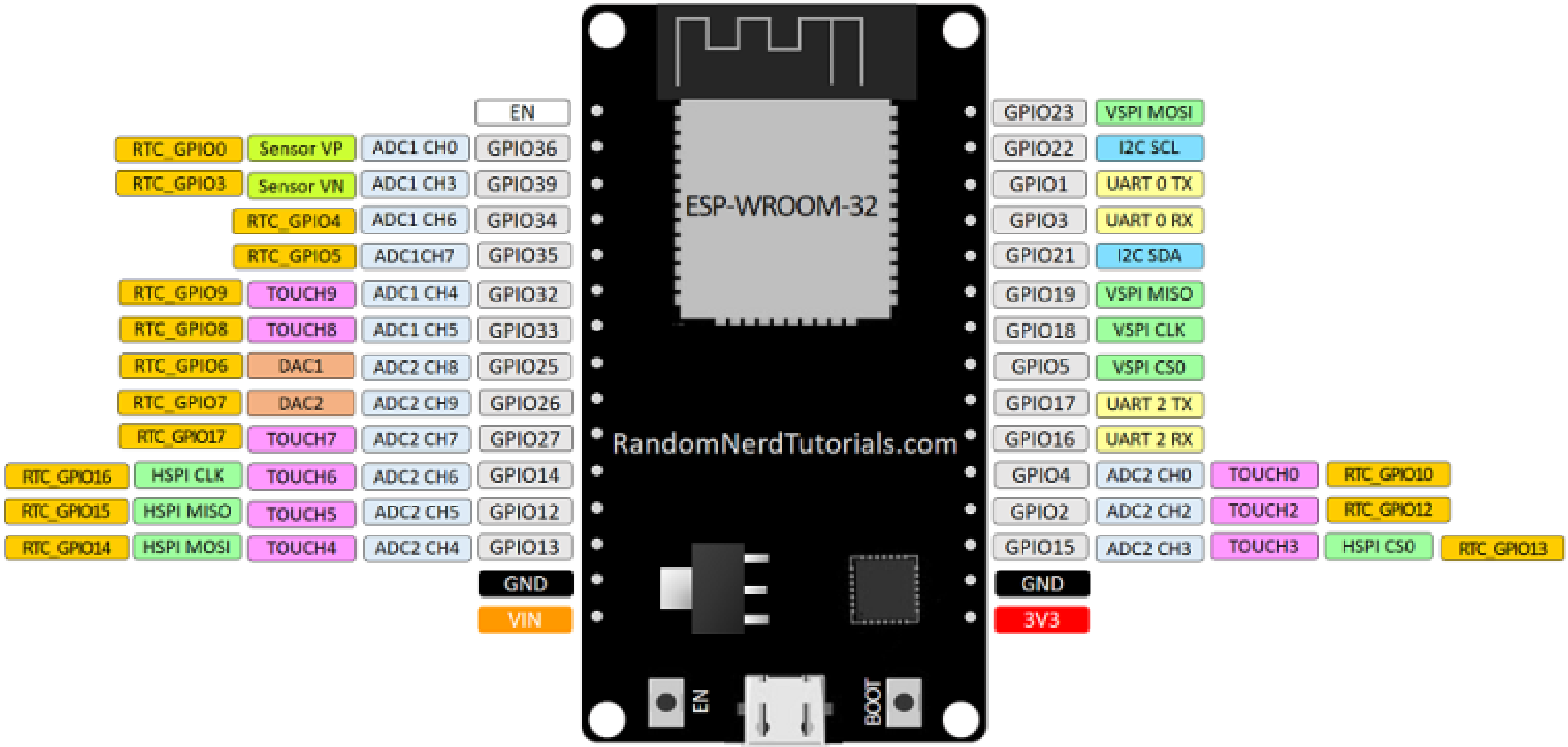


Power
GND
Serial Pin
Analog Pin
Control
Physical Pin
Port Pin
Touch Pin
DAC Pin
PWM Pin
ANT1
C7
C6
C12
C11
C5
C18
R7
C1
U1
C2
C16
C15
U2
U3
GND
3.3V
ChipPU 9
ADC PA RTCIO0 ADC1_0 SensVP GPIO36 5
ADC PA RTCIO3 ADC1_3 SensVN GPIO39 8
RTCIO4 ADC1_6 VDET1 GPIO34 10
RTCIO5 ADC1_7 VDET2 GPIO35 11
XTAL32 Touch9 RTCIO9 ADC1_4 GPIO32 12
XTAL32 Touch8 RTCIO8 ADC1_5 GPIO33 13
DAC_1 RTCIO6 ADC2_8 EMAC_RXD0 GPIO25 14
DAC_2 RTCIO7 ADC2_9 EMAC_RXD1 GPIO26 15
Touch7 RTCIO17 ADC2_7 EMAC_RX_DV GPIO27 16
HS2CLK SDCLK HSPICLK MTMS Touch6 RTCIO16 ADC2_6 EMAC_TXD2 GPIO14 17
HS2DATA2 SDDATA2 HSPIQ MTDI Touch5 RTCIO15 ADC2_5 EMAC_TXD3 GPIO12 18
GND
HS2DATA3 SDDATA3 HSPIID MTCK Touch4 RTCIO14 ADC2_4 EMAC_RX_ER GPIO13 20
SDDATA2 HS1DATA2 U1RXD SPIHD GPIO9 28
SDDATA3 HS1DATA3 U1TXD SPIWP GPIO10 29
SDCMD HS1CMD U1RTS SPICS0 GPIO11 30
GND
36 GPIO23 VSPIID HS1STROBE
39 GPIO22 EMAC_TXD1 U0RTS VSPIWP
41 GPIO1 EMAC_RXD2 U0TXD CLKOUT3
40 GPIO3 U0RXD CLKOUT2
PROGRAM Port
42 GPIO21 EMAC_TX_EN VSPIHD
NC
38 GPIO19 EMAC_TXD0 U0CTS VSPIQ
35 GPIO18 VSPICLK HS1DATA7
34 GPIO5 EMAC_RX_CLK VSPICS0 HS1DATA6
27 GPIO17 EMAC_CLK_180 U2TXD HS1DATA5
25 GPIO16 EMAC_CLK_OUT U2RXD HS1DATA4
24 GPIO4 EMAC_TX_ER ADC2_0 RTCIO10 Touch0 HSPIHD SDDATA1 HS2DATA1
23 GPIO0 EMAC_TX_CLK ADC2_1 RTCIO11 Touch1 CLKOUT1
22 GPIO2 HSPIWP ADC2_2 RTCIO12 Touch2
21 GPIO15 EMAC_RXD3 ADC2_3 RTCIO13 Touch3 MTDO HSPICS0 SDCMD HS2CMD
33 GPIO8 SPID U2CTS HS1DATA1 SDDATA1
32 GPIO7 SPIQ U2RTS HS1DATA0 SDDATA0
31 GPIO6 SPICLK U1CTS HS1CLK SDCLK

04 AUG 2016
ver 1 rev 0

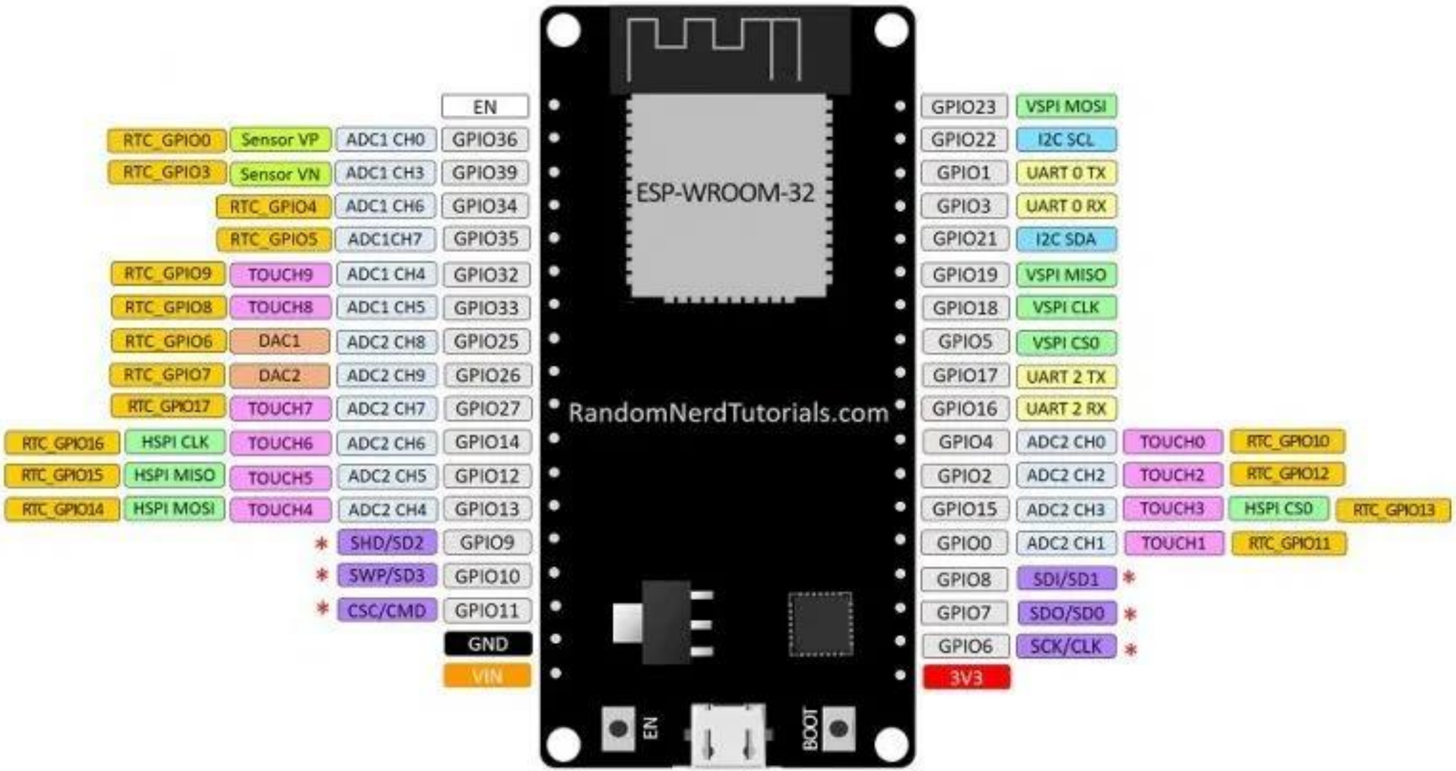
ESP32 DEVKIT V1 – DOIT

version with 30 GPIOs



ESP32 DEVKIT V1 – DOIT

version with 36 GPIOs


EN
RTC_GPIO0 Sensor VP ADC1 CH0 GPIO36
RTC_GPIO3 Sensor VN ADC1 CH3 GPIO39
RTC_GPIO4 ADC1 CH6 GPIO34
RTC_GPIO5 ADC1CH7 GPIO35
RTC_GPIO9 TOUCH9 ADC1 CH4 GPIO32
RTC_GPIO8 TOUCH8 ADC1 CH5 GPIO33
RTC_GPIO6 DAC1 ADC2 CH8 GPIO25
RTC_GPIO7 DAC2 ADC2 CH9 GPIO26
RTC_GPIO17 TOUCH7 ADC2 CH7 GPIO27
RTC_GPIO16 HSPI CLK TOUCH6 ADC2 CH6 GPIO14
RTC_GPIO15 HSPI MISO TOUCH5 ADC2 CH5 GPIO12
RTC_GPIO14 HSPI MOSI TOUCH4 ADC2 CH4 GPIO13
* SHD/SD2 GPIO9
* SWP/SD3 GPIO10
* CSC/CMD GPIO11
GND
VIN
GPIO23 VSPI MOSI
GPIO22 I2C SCL
GPIO1 UART 0 TX
GPIO3 UART 0 RX
GPIO21 I2C SDA
GPIO19 VSPI MISO
GPIO18 VSPI CLK
GPIO5 VSPI CS0
GPIO17 UART 2 TX
GPIO16 UART 2 RX
GPIO4 ADC2 CH0 TOUCH0 RTC_GPIO10
GPIO2 ADC2 CH2 TOUCH2 RTC_GPIO12
GPIO15 ADC2 CH3 TOUCH3 HSPI CS0 RTC_GPIO13
GPIO0 ADC2 CH1 TOUCH1 RTC_GPIO11
GPIO8 SDI/SD1 *
GPIO7 SDO/SD0 *
GPIO6 SCK/CLK *
3V3
ESP-WROOM-32
RandomNerdTutorials.com
EN
BOOT

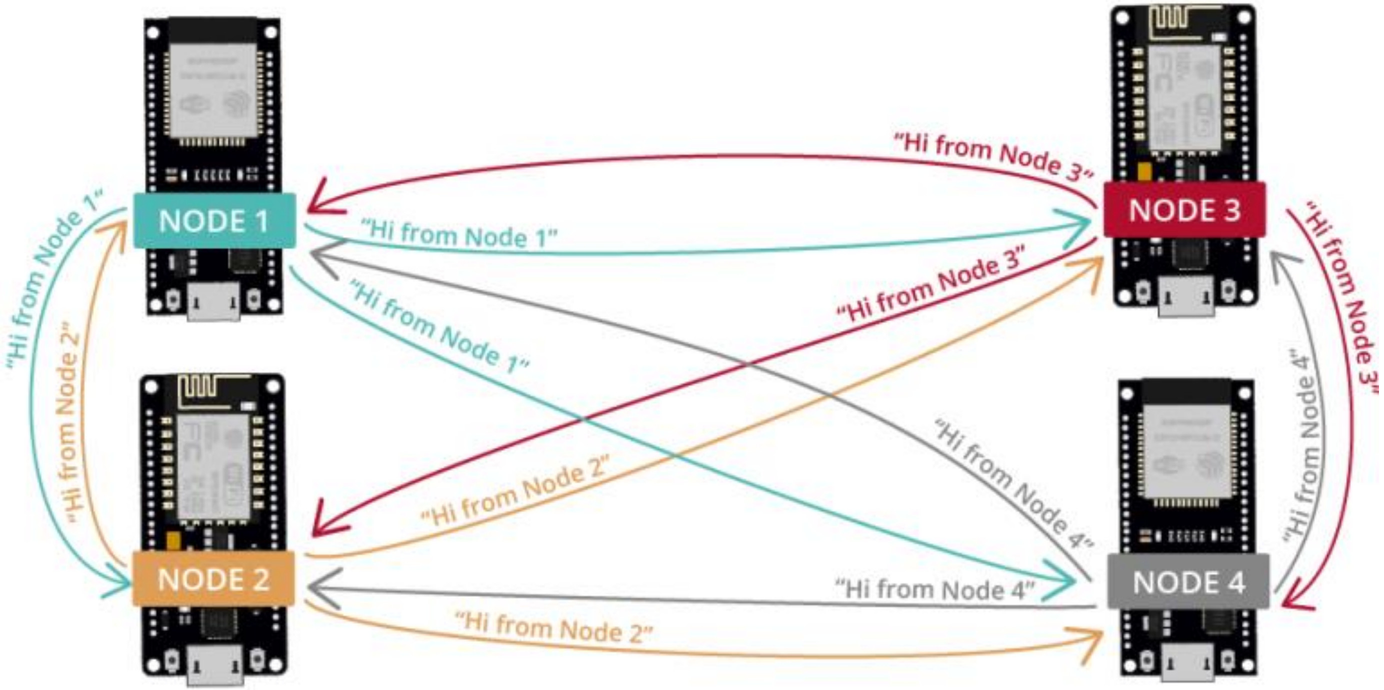
* Pins SCK/CLK, SDO/SD0, SDI/SD1, SHD/SD2, SWP/SD3 and SCS/CMD, namely, GPIO6 to GPIO11 are connected to the integrated SPI flash integrated on ESP-WROOM-32 and are not recommended for other uses.

Json Data point

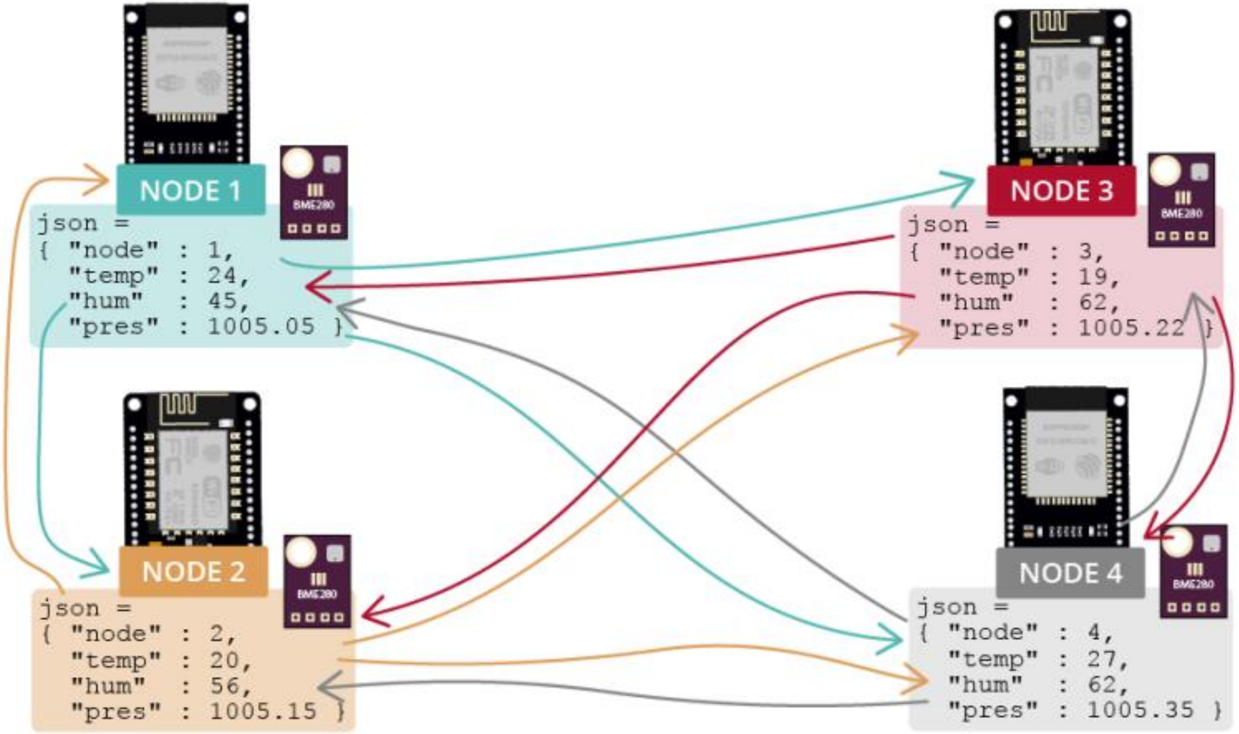
```
{
"temperature": 42.2,
"humidity": 70,
"hvacEnabled": true,
"hvacState": "IDLE",
"configuration": {
                "someNumber": 42,
                "someArray": [1,2,3],
                "someNestedObject": {"key": "value"}
                }
}
```

PAINLESSMESH

Topology mesh ESP32



Topology mesh ESP8266

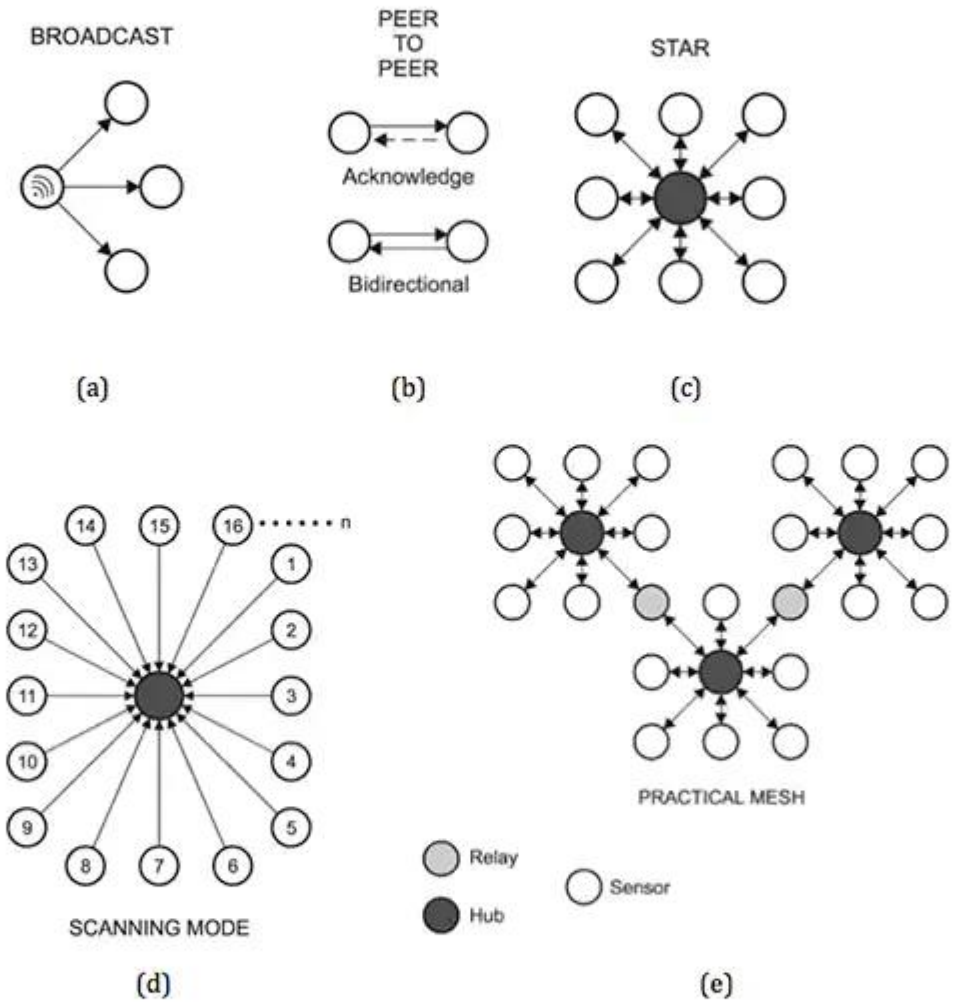


```
json =
{ "node" : 1,
  "temp" : 24,
  "hum"  : 45,
  "pres" : 1005.05 }
```

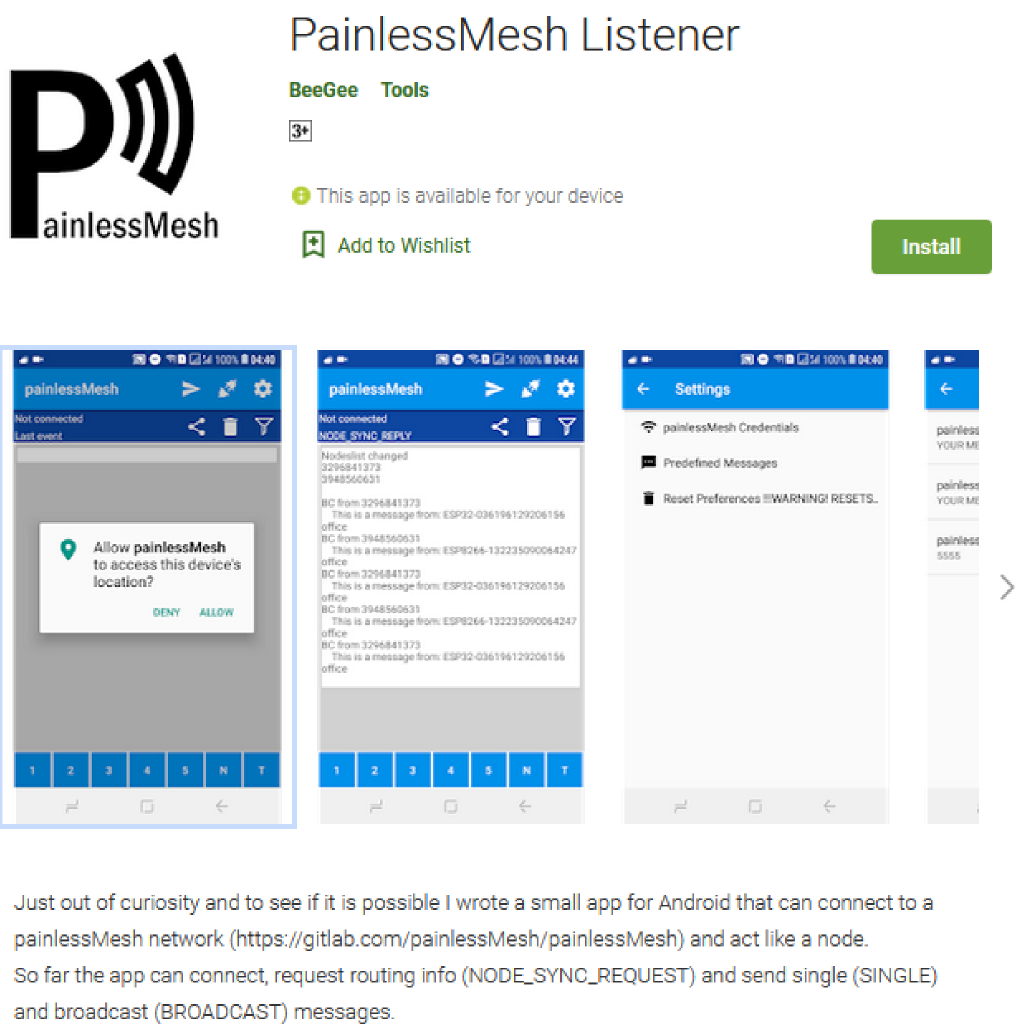
```
json =
{ "node" : 3,
  "temp" : 19,
  "hum"  : 62,
  "pres" : 1005.22 }
```

```
json =
{ "node" : 2,
  "temp" : 20,
  "hum"  : 56,
  "pres" : 1005.15 }
```

```
json =
{ "node" : 4,
  "temp" : 27,
  "hum"  : 62,
  "pres" : 1005.35 }
```



Network topology



PainlessMesh Listener

BeeGee Tools

3+

```
Not connected
NODE_SYNC_REPLY
Nodeslist changed
3296841373
3948560631

BC from 3296841373
   This is a message from: ESP32-036196129206156
office
BC from 3948560631
   This is a message from: ESP8266-132235090064247
office
BC from 3296841373
   This is a message from: ESP32-036196129206156
office
BC from 3948560631
   This is a message from: ESP8266-132235090064247
office
BC from 3296841373
   This is a message from: ESP32-036196129206156
office
```

Just out of curiosity and to see if it is possible I wrote a small app for Android that can connect to a painlessMesh network (https://gitlab.com/painlessMesh/painlessMesh) and act like a node.
So far the app can connect, request routing info (NODE_SYNC_REQUEST) and send single (SINGLE) and broadcast (BROADCAST) messages.

PainlessMesh

PainlessMesh is a true ad-hoc network, meaning that no-planning, central controller, or router is required.
Any system of 1 or more nodes will self-organize into fully functional mesh.
The maximum size of the mesh is limited (we think) by the amount of memory in the heap that can be allocated to the sub-connections buffer and so should be really quite high.

https://gitlab.com/painlessMesh/painlessMesh

API

```
#include <painlessMesh.h>
painlessMesh  mesh;


void painlessMesh::init(String ssid, String password, uint16_t port = 5555,
WiFiMode_t connectMode = WIFI_AP_STA, _auth_mode authmode =
AUTH_WPA2_PSK, uint8_t channel = 1, phy_mode_t phymode =
PHY_MODE_11G, uint8_t maxtpw = 82, uint8_t hidden = 0, uint8_t
maxconn = 4)
```

API

```
void painlessMesh::stop()
void painlessMesh::update( void )

void painlessMesh::onReceive( &receivedCallback )
            void receivedCallback( uint32_t from, String &msg )
void painlessMesh::onNewConnection( &newConnectionCallback )
            void newConnectionCallback( uint32_t nodeId )
void painlessMesh::onChangedConnections( &changedConnectionsCallback )
            void onChangedConnections()

bool painlessMesh::isConnected( nodeId )

void painlessMesh::onNodeTimeAdjusted( &nodeTimeAdjustedCallback )
            void onNodeTimeAdjusted(int32_t offset)
void onNodeDelayReceived(nodeDelayCallback_t onDelayReceived)
            void onNodeDelayReceived(uint32_t nodeId, int32_t delay)
```

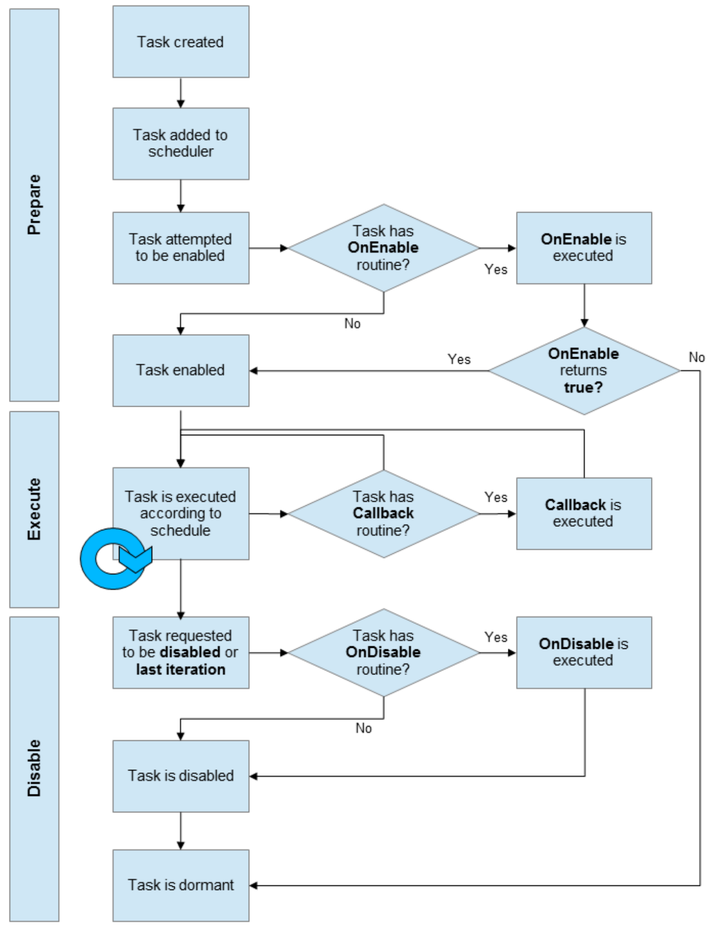
API

```
bool painlessMesh::sendBroadcast( String &msg, bool includeSelf = false)
bool painlessMesh::sendSingle(uint32_t dest, String &msg)

String painlessMesh::subConnectionJson()
uint32_t painlessMesh::getNodeId( void )
void painlessMesh::stationManual( String ssid, String password, uint16_t port, uint8_t
*remote_ip )
```

NO DELAY

But TaskScheduler

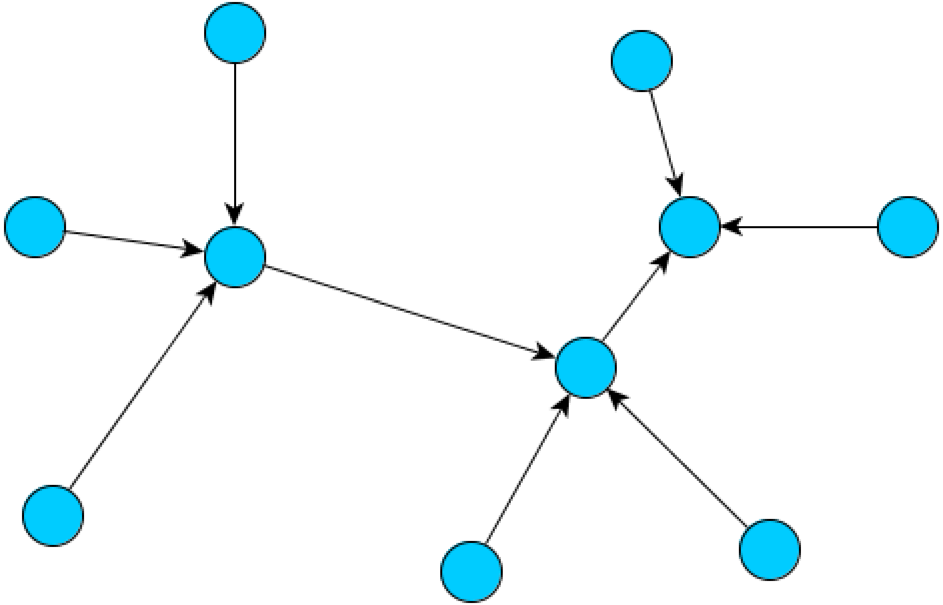


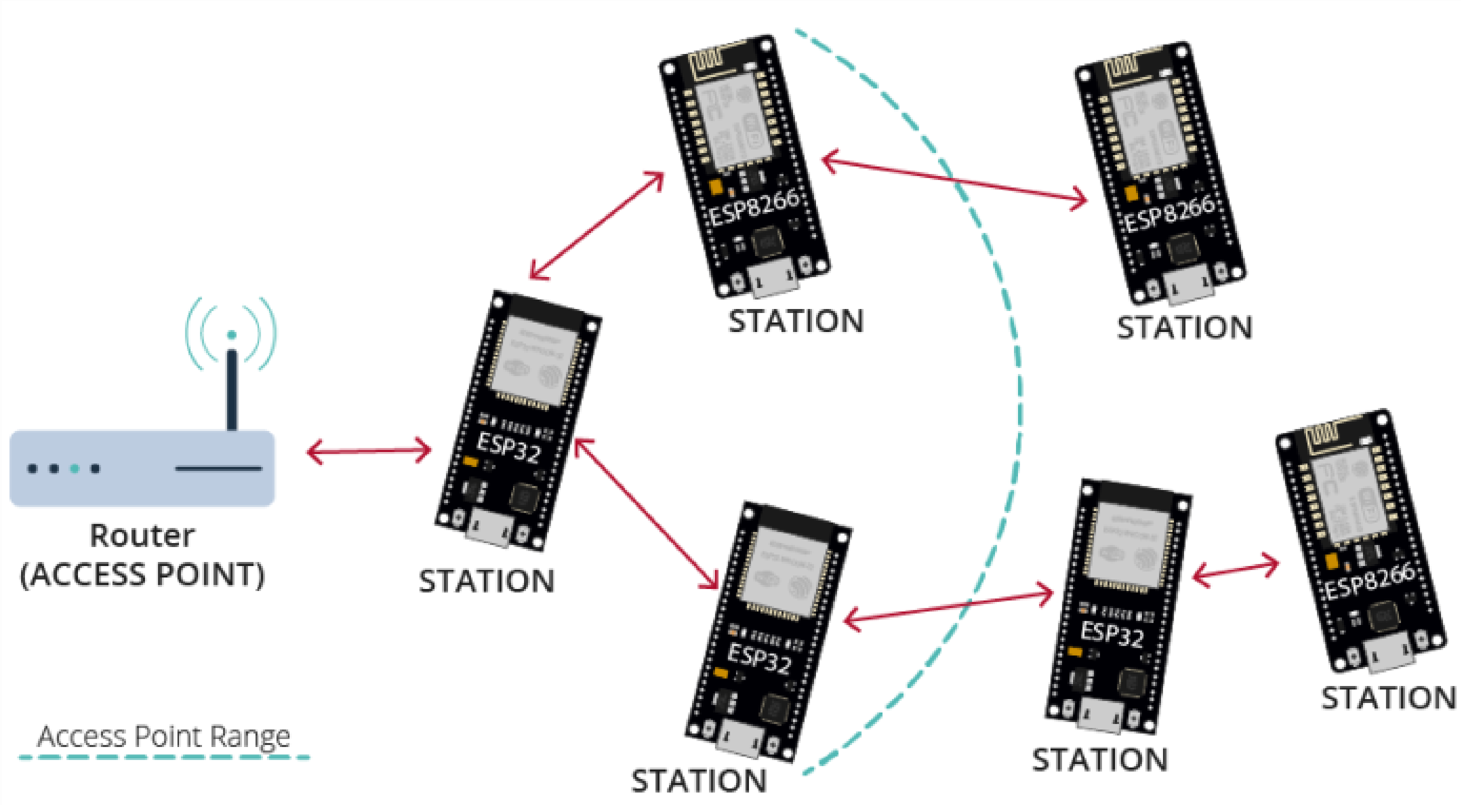
Prepare
Execute
Disable
Task created
Task added to scheduler
Task attempted to be enabled
Task has **OnEnable** routine?
Yes
No
OnEnable is executed
OnEnable returns **true?**
Yes
No
Task enabled
Task is executed according to schedule
Task has **Callback** routine?
Yes
Callback is executed
Task requested to be **disabled** or **last iteration**
Task has **OnDisable** routine?
Yes
No
OnDisable is executed
Task is disabled
Task is dormant

MeshNet

```
#define   MESH_PREFIX       "meshnet"
#define   MESH_PASSWORD     "meshnet123"
#define   MESH_PORT         5555
```

Network Layout



ESP8266
STATION
ESP8266
STATION
ESP32
Router
(ACCESS POINT)
STATION
ESP8266
ESP32
ESP32
STATION
Access Point Range
STATION
STATION

THANKS

Do you have any question?

hasbiida@gmail.com

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, infographics & images by **Freepik**