IoT Workshop

Trygve Laugstøl <trygvis@trygvis.io>

What is IoT

Preparations

- Install Arduino IDE
- Install the “ESP8266 core” for Arduino, follow the guide on https://github.com/esp8266/Arduino#installing-with-boards-manager.
- Install PubSubClient library with Library manager from within the Arduino IDE.

What is an IoT Device?

- Constrained in (one or more of):
  - Memory
  - CPU
  - Network bandwidth and/or latency
  - Storage
- Has connectivity

IoT Devices - Example chips

Protocol	Chip	Specs
Bluetooth 4/5	nRF52x	32-64 MHz, Cortex-M0/M4F, 24-256k RAM, 192-1024 k Flash, $1.88-$3.85
WiFi	ESP8266/ESP32	80MHz-160MHz, 1-2 cores, ~80k RAM, < $1 - $4.53
LoRa	Semtech	$3.23 - $4.74

ESP8266 Specifications

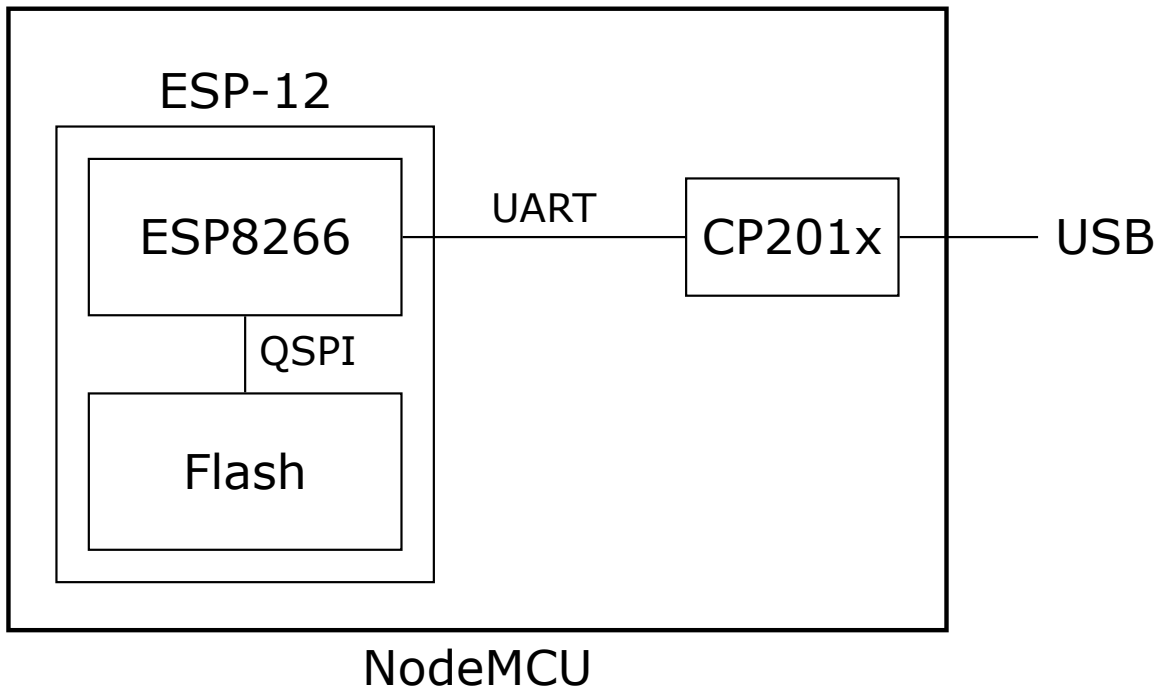
CPU	Tensilica Xtensa L106
Frequency	80MHz (160MHz possible
RAM	32 kB instruction RAM 80 kB user RAM 16 kB system RAM
Flash	None, integrated SPI driver
Peripherals	16 x GPIO I²C, SPI, I²S, UART 10 bit ADC Wi-Fi

ESP8266 Power usage

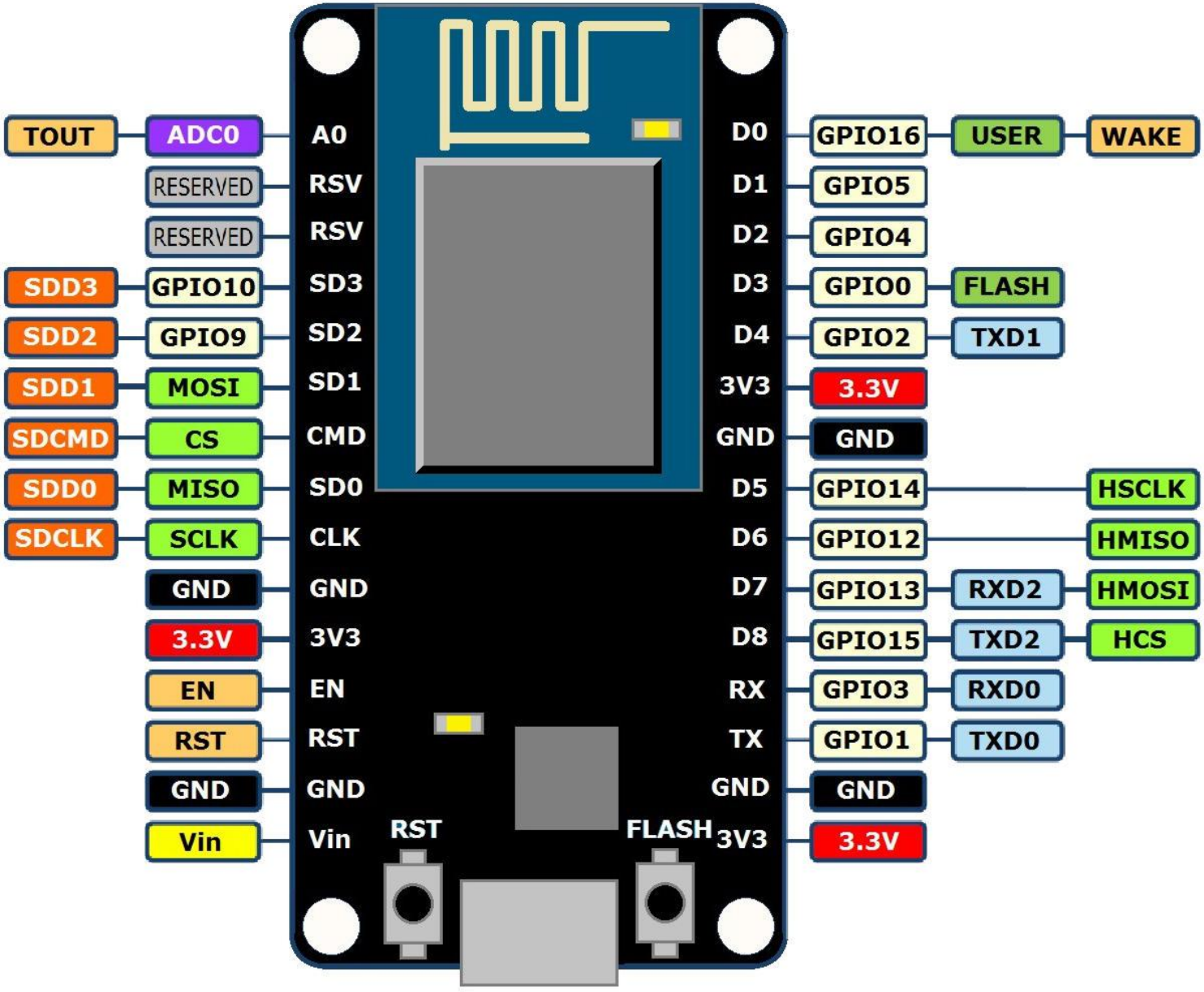
State	Current usage
Off	0.5 µA
Deep sleep with RTC	20 µA
Light sleep (with Wi-Fi)	1 mA
Sleep with peripherials	15 mA
TX	170 mA

NodeMCU

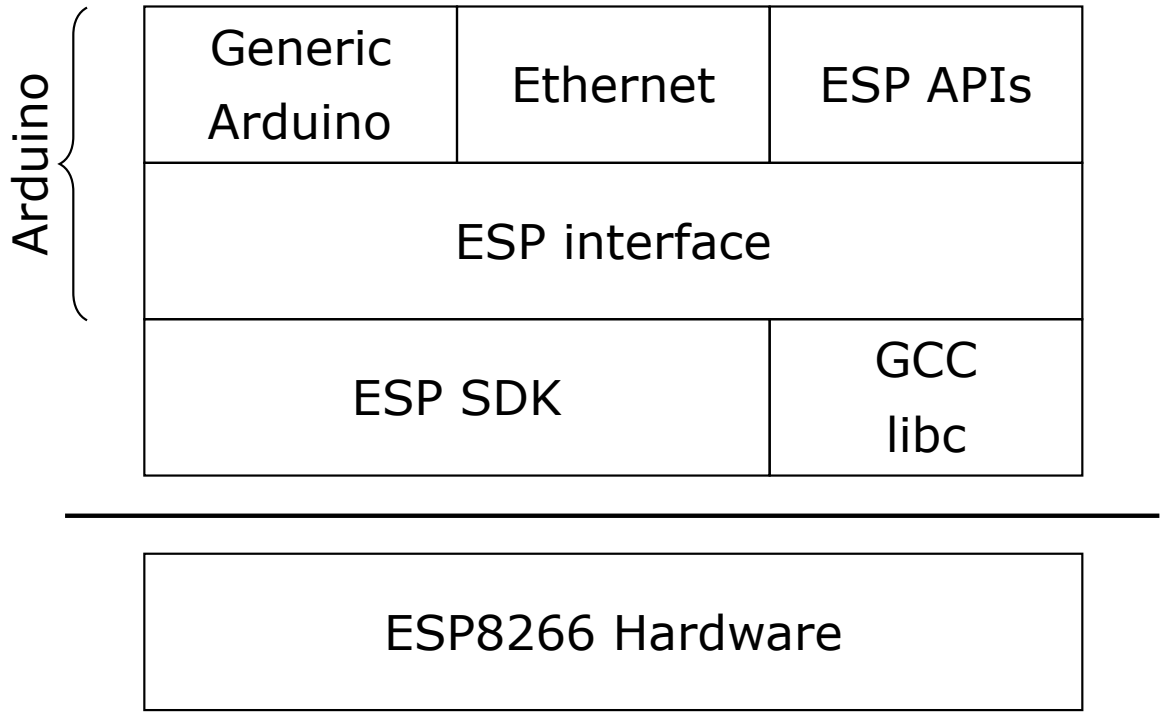
NodeMCU hardware



NodeMCU hardware



ESP8266 software layers



ESP8266 + Arduino

- Standard Arduino IDE
- ESP8266 Arduino core
  - https://github.com/esp8266/Arduino

Arduino IDE


Fil Rediger Skisse Verktøy Hjelp
sketch_apr25a
1 void setup() {
2 // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7 // put your main code here, to run repeatedly:
8
9 }
Module), 80 MHz, 4M (1M SPIFFS), v2 Higher Bandwidth, Disabled, None, Only Sketch, 921600 on /dev/ttyUSB0

Generic Arduino APIs

```
// Pin: D0, D1, etc.
// Mode: OUTPUT, INPUT, INPUT_PULLUP
// State: HIGH, LOW, 1/0

void pinMode(pin, mode);
void digitalWrite(pin, state);
int digitalRead(pin);

unsigned long now = millis();
unsigned long now = micros();
```

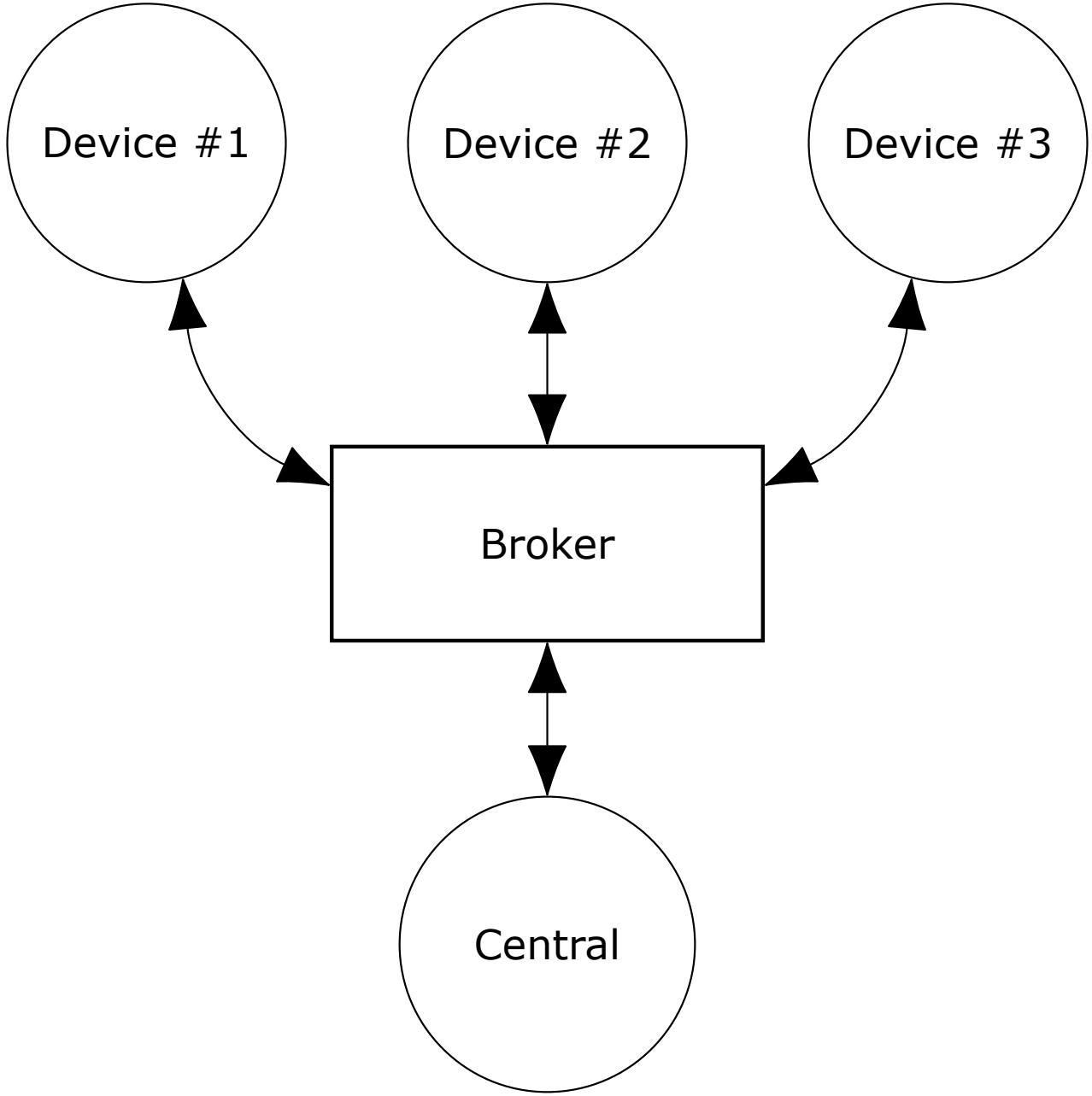
Assignment: `blink-a-led`

Lecture: MQTT

MQTT

- *Message Queuing Telemetry Transport*
- Wikipedia: MQTT

Device and application architecture with MQTT



MQTT Example

The temperature sensor:

- Publishes on:
  - `myapp/$device-id/temperature`
  - `myapp/$device-id/humidity`
  - `myapp/$device-id/altert`
- Subscribes to:
  - `myapp/$device-id/command`

The central application:

- Subscribes to:
  - `myapp/#/temperature`
  - `myapp/#/humidity`
- Publishes on:
  - `myapp/$device-id/command`

MQTT - The protocol

Agents have one of two roles:

- *Client*
  - Publishes *messages*
  - Subscribes / unsubscribes to *topics*
  - Keep alive
- *Broker* (aka Server)
  - Handles network connections
  - Keeps subscriptions
  - Manages client
    - Timeouts and disconnects
    - *last will*
  - Persistence of *retained* messages

MQTT - The protocol - MQTT Topic

- Topic name: `foo/bar/baz`
- Topic filter
  - `foo/bar/?`
  - `foo/#`

ESP Arduino APIs

```
class {
    void restart();
    uint32_t getFreeHeap();
    uint32_t getChipId();

    ...
} ESP;

// Usage
ESP.restart();
```

Connecting to a Wi-Fi

```
#include <ESP8266WiFi.h>

void setup() {
    WiFi.mode(WIFI_STA);
    WiFi.begin("NDC 2018", NULL);

    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }

    Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
}
```

MQTT on Arduino

PubSubClient is our MQTT client implementation.

Preparing to publish messages:

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

WiFiClient wifiClient;
PubSubClient mqtt(wifiClient);

String deviceId = "esp-" + String(ESP.getChipId(), HEX);

void setup() {
    // ...
    mqtt.setServer("broker.hivemq.com", 1883);
}
```

MQTT on Arduino

```
void loop()
{
    if (!mqtt.connected()) {
        reconnect();
    }
    else {
        mqtt.loop();
    }

    // Do work
}
```

MQTT on Arduino

```
void reconnect()
{
    do {
        Serial.println("Connecting to MQTT");
        delay(1000);
    } while (!mqtt.connect(clientId.c_str()));

    Serial.println("Connected to MQTT server");
}
```

MQTT on Arduino

```
void sendMessage()
{
    String topic = "ndc/" + deviceId + "/led";
    mqtt.publish(topic.c_str(), "1");
}
```

MQTT on Arduino

Preparing for subscriptions:

```
void setup() {
    ...
    mqtt.setCallback(callback);
}

void callback(char* topic,
              byte* payload,
              unsigned int length) {
}

void reconnect() {
    ...
    // Subscribe to any topics you need
    mqtt.subscribe(topic_pattern);
}
```

Assignment: `mqtt-with-button`

Content

https://github.com/trygvis/iot-workshop-ndc-2018