PSI3542 – 2023
SISTEMAS EMBARCADOS DISTRIBUIDOS

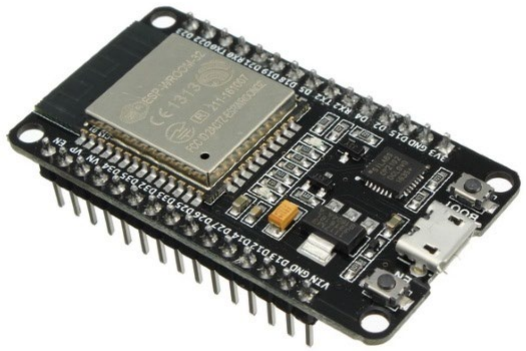
AULA 09 – ATIVIDADE 9.2 DISPOSITIVO MQTT PUBLISH-SUBSCRIBER

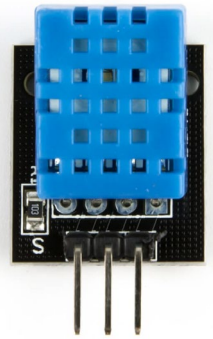
SERGIO TAKEO KOFUJI

KOFUJI@USP.BR

ATIVIDADE 9.2

DISPOSITIVO IOT SENSOR-ATUADOR publisher-subscriber mqtt





MQQT-SUB
led

MQQT-PUB
led

MQQT-PUB
Temp/hum

MQQT-PUB
led

ThingSpeak™

MQTT X

Canal ThingSpeak

Descrição

- Field1 – temperatura (0-100)
- Field2 – humidade (0-100)
- Field3 – led (on-off)
- Dispositivo MQTT01
  - dispositivo ESP32/ESP8266
- Dispositivo MQTT02
  - cliente MQTTX, para controle de liga/desliga do led da placa ESP32/ESP8266

Dispositivo MQTT subscriber

Programa main.py

Importar Módulos

```
import network
import time
import sys
from machine import Pin
#utilizaremos a versao “robusta” do cliente umqtt em vez da “simple”
from umqtt.robust import MQTTClient
```

Ajustar os Parâmetros de conexão

# MQTT Server Parameters
MQTT_CLIENT_ID = “xxxxxx”
MQTT_BROKER = "mqtt3.thingspeak.com”
MQTT_USER = “xxxxxx”
MQTT_PASSWORD = “yyyyyy”
#https://www.mathworks.com/help/thingspeak/subscribetoachannelfieldfeed.html
#substituir zzzzzz pelo channel ID ThingSpeed
MQTT_PUB_TOPIC = "channels/zzzzzz/publish”
MQTT_SUB_TOPIC = "channels/zzzzzz/subscribe/fields/field3”

#WIFI parameters
WIFI_SSID = “qqqqqq”
WIFI_PASSWD = “rrrrrr”

Callback function

```
#função callback
def callback(topic, msg):
    print((topic, msg))
    if msg == b'0':
        led1.on()
    else:
        led1.off(
            )
```

Configurar os pinos de LED e sensor

```
#GPIO16
#sensor = dht.DHT11(Pin(16))
sensor = dht.DHT22(Pin(16))

#vamos utilizar o led da placa ESP32/8266 (GPIO02)
led1=Pin(2, Pin.Out)
led1.on()
led1.off()
```

Conexão WIFI

```
#Create a WLAN network interface object. STA_IF = station aka client, connects to upstream WiFi access points
print("Connecting to WiFi", end="")
nic = network.WLAN(network.STA_IF)
nic.active(True)
nic.connect(WIFI_SSID, WIFI_PASSWD)
while not nic.isconnected():
    print(".", end="")
    time.sleep(0.1)
print(" Connected!")
```

Conexão com o broker mqtt thingspeak

```
#conexão com broker Thingspeak
print("Connecting to MQTT server... ", end="")
client = MQTTClient (MQTT_CLIENT_ID, MQTT_BROKER, user = MQTT_USER, password = MQTT_PASSWORD, ssl=False)

# callback to handle data when MQTT channel updates
client.set_callback(callback)

try:
   client.connect()
except Exception as e:
   print('could not connect to MQTT server {}{}'.format(type(e).__name__, e))
   sys.exit()
print("Connected!")

client.subscribe(MQTT_SUB_TOPIC)
```

Loop de espera de mensagem mqtt sub

```
#loop publish and subscribe
prev_weather = ""
while True:
    client.check_msg()
    print("Measuring weather conditions... ", end="")
    sensor.measure()
    message="field1={temp}&field2={hum}&status=MQTTPUBLISH".format(temp=sensor.temperature(),hum=sensor.humidity())
    print(message)
    if message != prev_weather:
        print("Updated!")
        print("Reporting to MQTT topic {}: {}".format(MQTT_TOPIC, message))
        client.publish(MQTT_TOPIC, message)
        prev_weather = message
    else:
        print("No change")
    time.sleep(1)
```

MQTTX

MQTTX

- O Cliente MQTTX será utilizado para testar o dispositivo ESP32/8266
- Configure o MQTTX – conecte ao dispositivo mqtt02 do ThingSpeed
  - Dispositivo **publisher** do **field3**
    - Tópico channels\channelid\publisher\Fields\field3
- Verifique o envio de dados ao ThingSpeed
- Verifique o envio de dados ao dispositivo ESP32/8266

Bom Trabalho

kofuji@usp.br