Computer Networks

Project 2

**MQTT**

Mqtt is mosquito telemetry transport system which is a lightweight open messaging protocol that provides resource-constrained network clients with a simple way to distribute telemetry information in low-bandwidth environments.

MQTT is an OASIS standard messaging protocol for the Internet of Things (IoT). It is designed as an extremely lightweight publish/subscribe messaging transport that is ideal for connecting remote devices with a small code footprint and minimal network bandwidth. MQTT today is used in a wide variety of industries, such as automotive, manufacturing, telecommunications, oil and gas, etc.

What where and why I used mqtt?

This system was conceived to demonstrate application of networking centred around Internet of Things. MQTT messages are usually transported over TCP/IP protocol. In order to demonstrate the functionality of MQTT in IoT applications, I chose this protocol. This protocol is used in Raspberry Pi to create a broker-client architecture.

**UART interface**

UART is a hardware communication protocol that uses asynchronous serial communication with configurable speed. There is no clock signal to synchronize the output bits from the transmitting device going to the receiving end.

There are two wires TX and RX. TX takes care of the transmitting end and RX takes care of the receiving end.

What where and why did I used uart?

As stated above, MQTT operates over TCP/IP protocol and since the Pi Pico doesn’t have an inherent capability to communicate over this protocol without the help of additional shield or component, I chose to mimic sending the data over to Raspberry Pi through UART. This also facilitated the demonstration of a peer-to-peer networking along with server-client communication topology as in the case of MQTT.

IDE used: Thorny for coding into raspberry pi

Hardware used: raspberry pi, raspberry pi pico

**Use Case:**

This project showcases how a typical IoT system may operate, where nodes are used to collect sensor data and publish or broadcast over a channel where every other device relying on this data can subscribe to. This helps I reducing the overhead of frequent communication to the server to ask for the most recent value of any sensor data. Like IoT technology, this system is pervasive and is widely used in a range of applications from home automation systems to industrial automation to environmental monitoring.

**Picture / Diagram / Logical Diagram**

Diagram

Description automatically generated

Diagram, schematic

Description automatically generated

A picture containing text, electronics, circuit

Description automatically generated

**High Level Messaging / Functionality / Functional description / Purpose**

The project’s vision was to use two raspberry pi Pico. First, as a publisher that pushes randomly generated data (mimicking as if being received via a sensor) over to MQTT broker. The second was envisioned to be a subscriber which listens to the broadcasted data by the first pico. However, since the Pico’s do not have inherent capability of TCP/IP communication, I decided to choose UART to transmit the data over to Raspberry Pi. The Raspberry pi then, uses an additional script to publish this data received over UART to the MQTT topic called “readingserver” on localhost at port 1883.

Since, Raspberry Pi has only one UART channel, I decided to simulate the subscribing process of MQTT on Raspberry Pi itself.

Raspberry Pi also acts as an MQTT broker that facilitates message exchanges between publishers and subscribers.

**Algorithm / Pseudo Code / Structure / Logical thinking**

First, we start MQTT server by,

mosquito\_sub -h localhost -t “readingserver”

Following code runs in RPi Pico:

Import libraries

Initialise UART at specific baud rate

Loop:

Generate random number

Send over UART

Following code for UART message processing runs in RPI:

Import libraries

Initialise Parameters for MQTT connection

Initialise UART

While (True)

Read data from UART

Decode to utf-8

Publish to MQTT topic

Following code is used to subscribe and receive data on RPi (originally expected to run on another Pico)

Import libraries

Initialise MQTT parameters

Establish connection with MQTT broker

Subscribe to the topic

While (True):

If Message received:

Print message

**Networking / Comms model: e.g. Client Server or Peer-to-Peer - > Any lower layers (Application and/or Network)**

Here, we demonstrate both client-server and peer-to-peer type of networking

MQTT is a client server protocol that runs on RPi with RPi being the server (broker) and Pico being the client (in this case, publisher). In lieu of second Pico as Subscriber, we use RPi itself as a Subscriber too.

Peer to peer networking is demonstrated through UART. Here we send randomly generated data from Pico to RPi.

**Implementation details**

All the codes were written using Python3 script. We use mosquito library to execute MQTT related commands.

**Highlights/Unique Selling Points  of your system/implementation.**

It is not platform dependent. Since any hardware platform can connect over to the localhost on port 1883 through specified topic.

**Changes, advances and/or improvements from Project 1, and why**

Did not submit project one due to delay in research but thought about building a master server through which I could interface between two rpi pico’s, one being the sender and other being the receiver.

***References:***

1. (<https://medium.com/geekculture/serial-connection-between-raspberry-pi-and-raspberry-pico-d6c0ba97c7dc>)

1. (<https://www.emqx.com/en/blog/use-mqtt-with-raspberry-pi>)
2. (<https://github.com/raspberrypi/pico-micropython-examples>)

Name: Pranav Manish Vora