MANF 486: Mechatronic Systems Laboratory

Lab Project - MQTT Protocol

School of Engineering

Introduction

In this report, we will be exploring the MQTT protocol. MQTT is a lightweight, publish-subscribe, machine to machine network protocol for message queue/message queuing service. MQTT requires the use of a broker to facilitate the communication between multiple clients that can all share and receive information. The MQTT protocol utilizes a publish-subscribe system, meaning clients can publish data to a given topic which can then be accessed by a different client if they are subscribed to said topic. This results in the only data being shared to a client is that of what they require and requires lower bandwidth from the network being used by the protocol.

The communication between machines is a crucial component of any manufacturing facility, allowing for data to be shared and decisions to be made from said data. MQTT offers a lightweight solution to share data across a manufacturing facility. It requires an extremely low bandwidth which in turn results in very low power consumption compared to other communication protocol alternatives. This is advantageous as it saves network bandwidth for other systems and reduces power consumption which is especially useful on battery powered devices.

The MQTT protocol can be used on multiple levels of communication in a manufacturing facility, all the way from smart sensors to a web system such as AWS (Amazon Web Service). An example of one of these levels of communication is a scenario that we have studied in other courses (MANF 455 and MANF 465); the communication between the MES system and the SCADA system. A very informative graphic of these levels of communication can be seen below:

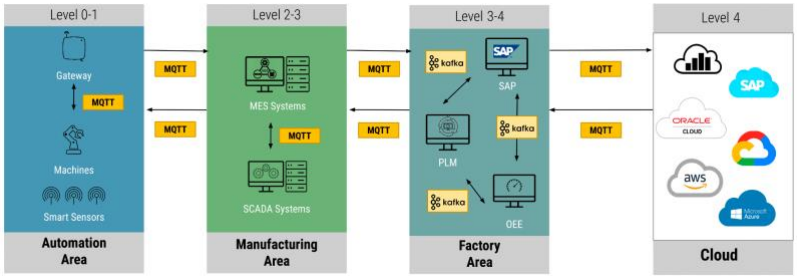

Figure 0: MQTT communication graphic

The MQTT protocol is gaining popularity in the manufacturing industry and with the surge of IOT devices/networks. Having an understanding of this technology allows us to be ready for future opportunities in either school or even our careers where the MQTT protocol could be implemented. This report describes the steps that we took to implement this protocol on a small scale which could easily be expanded if desired.

Implementation

To demonstrate the use of MQTT protocol using a PLC, we have selected a use case in which an MPS Separating Station counts up the number of workpieces that have been processed and reports to a remote client whenever a batch has been successfully processed. A batch in this case refers to 3 workpieces. Our design also allows for the remote client to halt the processing of workpieces at any time by sending the number “1” to the PLC client or a “0” to reactivate the MPS station. However, this second application will not be discussed in this report.

To implement the MQTT protocol in our design, we required 4 things, namely:

1. The LMQTT library to be properly integrated into the Siemens project
2. Download and install an MQTT broker software onto a laptop
3. Download and install MQTT client software onto the same laptop
4. Write a custom program to allow communication between the PLC and the remote client

Step 1: Integrating the LMQTT library into Siemens portal

1.1 Install LMQTT FB and DB

The LMQTT library can be found under Global libraries. This library is imported into the MainOB1 where it will create a corresponding Function Block(FB4) and DataBlock(InstLMQTT_Client/DB1). FB4 has code which implements the MQTT protocol whereas DB1 contains the input/output parameters of the LMQTT block. We then create another DataBlock(MqttDb/DB2) which has the same parameters as DB1 and link its tags to the LMQTT block. The inputs and outputs of the LMQTT block as it communicates with various clients can be changed and viewed from this datablock(see Figure 1)

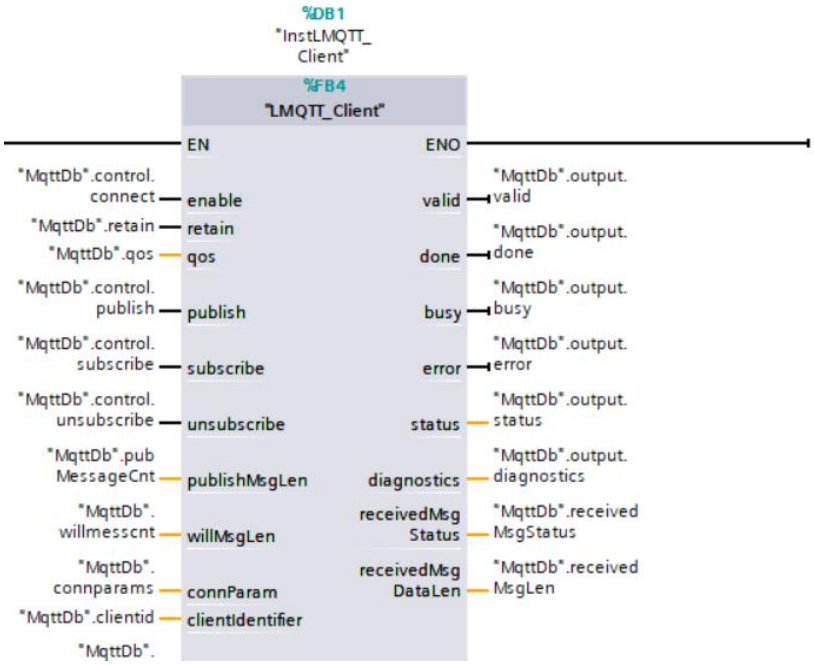

Figure 1. LMQTT block with inputs and outputs linked to DB2

1.2 Set Key Input parameters

Some values of the LMQTT block must be configured in DB2 in order to set up connection with the broker. Below are some of the necessary parameters that must be set up:

I. Connection Parameters Used: IP address = 10.0.0.6, Port = 1883 and Hardware ID(hwID)=64
II. Publish Topic Used: “BATCH_COUNT”
III. Subscribe Topic Used: “PAUSE”

Step 2: Download and install MQTT broker software

The broker software used in this project is Mosquitto broker. It is run using Command Prompt interface set to administrator privileges. On the command window, we are able to activate and deactivate the broker, pin the IP addresses of other clients to diagnose connection issues and open the 1883 port required for MQTT communication(see Figure 2)

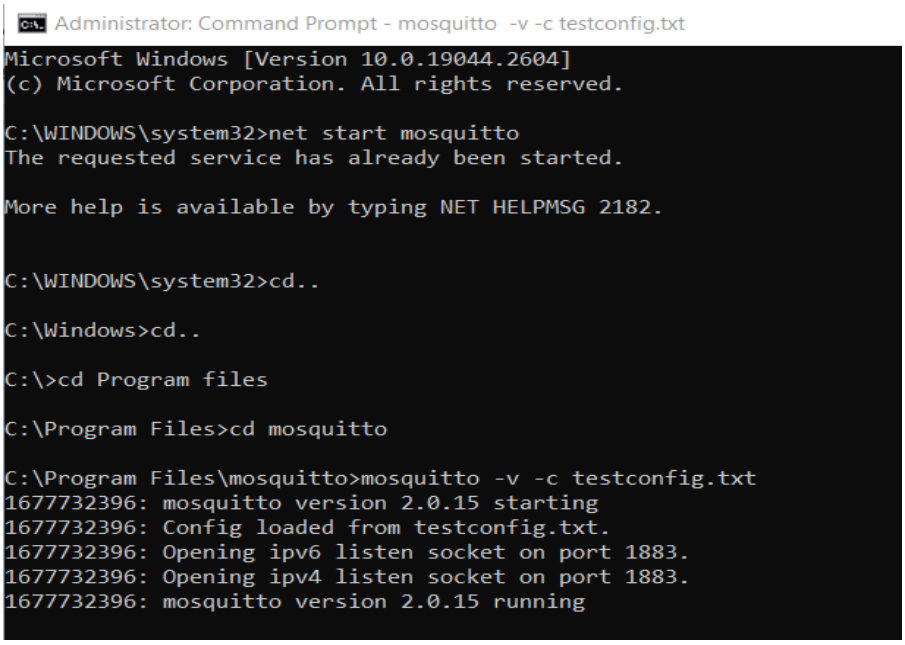

Figure 2. Code to open 1883 port

Step 3 : Download and install MQTT client software onto the same laptop

The MQTT client used in this project is MQTT explorer. It turns the computer its installed on into a client and provides an interface where the user can subscribe and publish to any desired topic. The IP address of the broker to be connected to can be specified on the login page.

Step 4: Write a custom program to allow communication between the PLC and the remote client

4.1 Data Collection Feature

We programmed the MPS Station to trigger the output tag Q1_LED whenever a workpiece is being processed. This tag is used to trigger a counter block which counts up to 3. After which, batch number is incremented in the next counter and the value is published to the specified publish topic “BATCH_COUT”(see Figure 3).

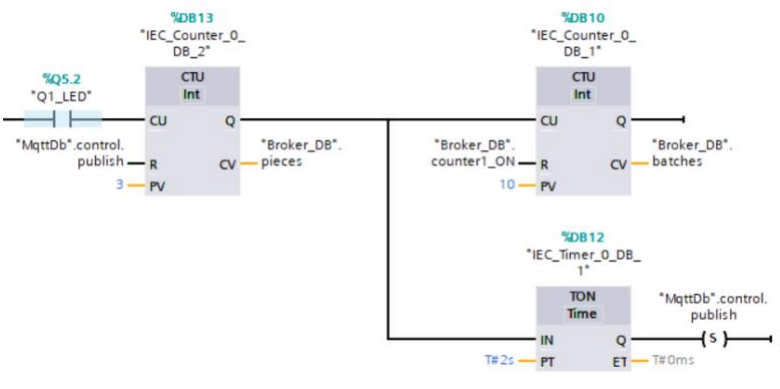

Figure 3. Publish data rung

The publish tag works on a rising edge hence we must reset the bit back to 0 in order to trigger it for the next batch. The rung in Figure 4 serves this purpose.

Figure 4. Publish Reset rung

The batch count is now published to the MQTT Explorer (see Figure 4).

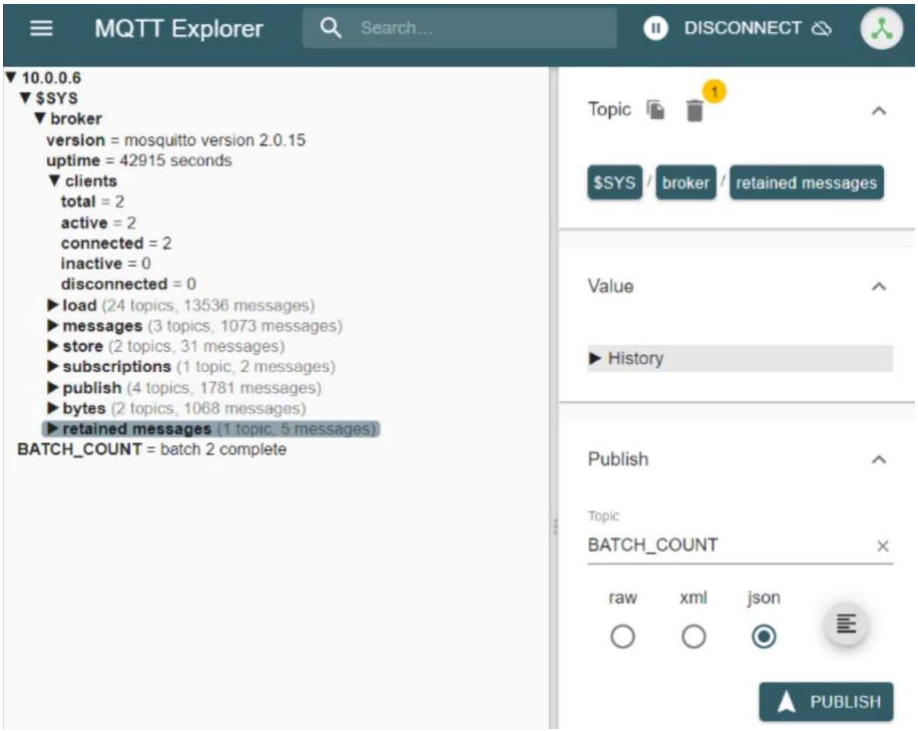

Figure 5. Batch Count incremented in MQTT Explorer Window

4.2 MQTT Explorer Control feature

As mentioned, we also allowed the PLC to be controlled using input from the remote client. The MQTT Explorer client can publish to the topic “PAUSE” which the PLC is subscribed to. Looking at the SFC in Figure 6 we see that sending a 1 to the PLC disrupts the transition condition, halting the station at step 7.

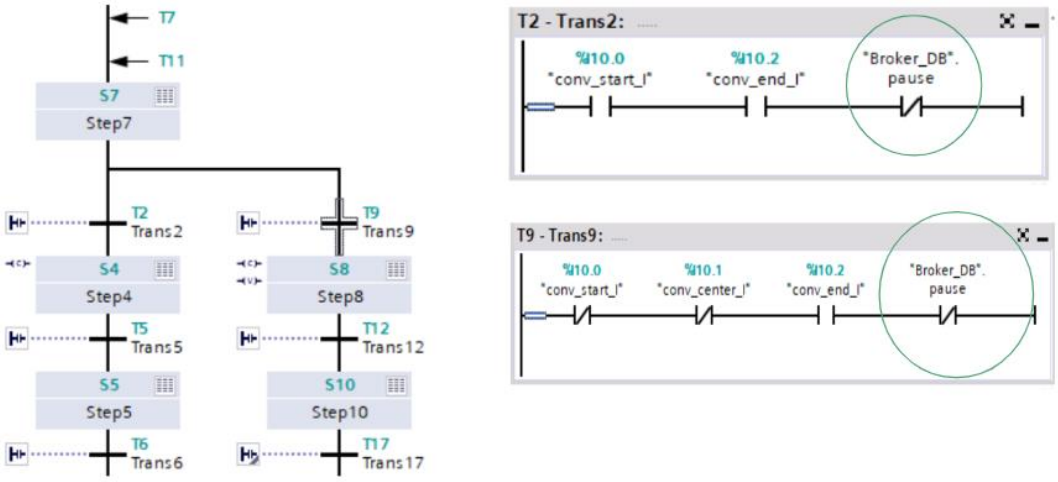

Figure 6. Using Pause tag to control SFC flow

However, the rung contacts can only be controlled by booleans values and not byte values sent over the network. We use the network in Figure 7 to change the bytes to the appropriate type.

Network 5:

Comment

```
1  "Broker_DB".pause := BYTE_TO_BOOL("MqttDb".receivedmessage[0]);
2  "MqttDb".message[6] := INT_TO_BYTE( "Broker_DB".batches);
3
```

Figure 7. Type casting network

Conclusion

The use of MQTT protocol has revolutionized the way in which data can be transmitted in IoT applications. This report has demonstrated how the MQTT protocol can be implemented in a PLC to count the number of workpieces processed and report to a remote client when a batch has been successfully processed. Moreover, the report has highlighted how the remote client can halt the processing of workpieces by sending the appropriate message to the PLC. It is clear that the MQTT protocol has numerous applications across various domains, ranging from sensor level to machine level, and even to cloud server level. This technology is highly efficient and uses very low bandwidth, making it ideal for IoT applications.

One important takeaway from this report is the need to be aware of the data types being sent and received through MQTT protocol. While this technology is highly efficient, it is important to ensure that data is being transmitted in the appropriate format to prevent errors in data processing. This is especially critical when sending commands to the PLC for controlling the

processing of workpieces. It is essential to ensure that the appropriate data type is being sent in order to avoid any errors or malfunctions.

The future applications of MQTT protocol are endless, as this technology continues to be adopted in various industries. As IoT applications continue to grow, MQTT protocol is expected to play a significant role in the development of smart cities and smart homes. With the increasing need for efficient and low-cost communication in IoT applications, MQTT protocol is well-positioned to meet the demands of the market. As such, it is clear that the MQTT protocol will continue to be an important technology in the IoT space and will play a critical role in shaping the future of our world.