

# 01OVEOQ-Innovation wireless platforms for the internet of things

**Exercise:** Exam Type 3 – Exercise 2

**Candidate:** Yenchia Yu (s287513)

**Device:** Pycom FiPy , Pycom PySense 2.0 X

## 1. Pycom chat via MQTT

In order to let FiPy device can obtain temperature information and chat with other devices using MQTT protocol, library including “mqtt.py”, “SI7006A20.py” and “pycoproc\_2.py” are used. The public MQTT broker used in this system is “test.mosquitto.org” with port 1883. As you can see from code “EX2-1 MQTT”, the code is divided into two parts: the FiPy chat system and test-shell to test the working of the system.

In the design of FiPy chat system, it will subscribe to topic “/user/broadcast/rx” and topic “/user/rexYu\_Fipy/rx”. In order to receive messages from the subscribed topics, method “check\_msg()” of the mqtt client is needed to be called periodically. To avoid the conflict with user input, a micro-thread is created to run the “check\_msg” method every second. As Fig.1 (left) shows, when a FiPy received a message from the topics it subscribes, it will print the sender and the message on Pymark terminal and publishes its reply to a specific topic according to the received message.

Meanwhile, FiPy can also be the one to start the conversation. As Fig.2 (left) shows, in Pymark terminal, we can use “sendto all <message>” command to send a broadcast message. And if we want to send a message to a specific user, we can use “sendto <user\_name> <message>” command. In the FiPy chat system, commands will be convert in to the format as {from: <string>, msg: <string>} and insert to the payload of MQTT message.

The test-shell is designed to test the function FiPy chat system. AS Fig.1 (right) shows, we can use the same command as above to send message in a specific topic. And similar to FiPy chat system, the test-shell subscribes to topic “/user/broadcast/rx” and topic “/user/rexYu\_desktop/rx”. Since it is a test tool, when it receives a message, it only prints it on the terminal but not return other information (as Fig.2 right shows).

<pre>&gt; &gt; &gt; user/broadcast/rx From rexYu_desktop: hello man! user/rexYu_Fipy/rx From rexYu_desktop: diagnostic &gt;</pre>	<b>FiPy</b>	<pre>&gt; sendto all hello man! &gt; From rexYu_Fipy: Hi I am here!  &gt; sendto rexYu_Fipy diagnostic &gt; From rexYu_Fipy: It's hot here! The temperature is 49.34 degree Celsius. &gt;</pre>	<b>Test-shell</b>
---	-------------	---	-------------------

**Fig.1 Test-shell(right) send messages and get response and FiPy(left) prints the received message**

<pre>&gt; sendto all hello &gt; sendto rexYu_desktop diagnostic &gt;</pre>	<b>FiPy</b>	<pre>From rexYu_Fipy: hello From rexYu_Fipy: diagnostic</pre>	<b>Test-shell</b>
--	-------------	---	-------------------

**Fig.2 FiPy (left) send messages and test-shell(right) prints messages**

## 2. Pycom chat over Bluetooth protocol

To implement a FiPy chat system with Bluetooth protocol, FiPy device will be setup as a GATT server and app “BLE scanner” on a smart phone will be run as a GATT client. The additional libraries used in this exercise are “SI7006A20.py” and “pycoproc\_2.py”.

In this chat system, the conversation can only start by GATT client (app BLE scanner). During the design, two problems are found: 1) When “BLE scanner” need to send a message larger than 20 bytes, it will split the message and send in multiple packages. 2) “BLE scanner” can only read at most the first 20 bytes of the payload in a single package. When the payload is larger than 20 bytes, the additional messages will be dropped.

- Solution for problem 1:  
In order to receive the complete message on FiPy chat system, messages that come not later than 0.5 seconds than the previous message will be aggregated together into one message. When no more messages are received after 0.5 seconds form the last message, we assume the transmission of a long message is finished. Then the chat system can perform the analysis and response to the message.
- Solution for problem 2:  
When FiPy chat system need to send a message larger than 20 bytes to “BLE scanner”, it will split the message into 20 bytes and transmit from the last to the first one(to show better in “BLE scanner”). The result is as Fig.3 shows.

For FiPy chat system, it will print the connection status of clients and the received messages in pymark terminal.

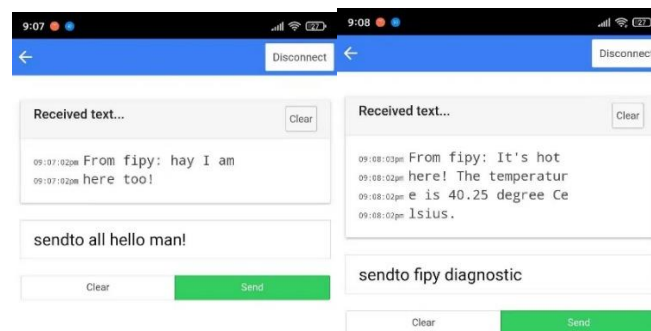


Fig.3 BLE scanner send messages to FiPy

```
Start BLE service
client connected
client disconnected
client connected
Received message: hello man!
Received message: diagnostic
█
```

Fig.4 FiPy prints the connection status and received messages

### 3. Wifi analyzer with telegram

To create a wifi analyzer with FiPy and send message to telegram, we first need to configure the wlan on FiPy to STA\_AP mode. In STA\_AP mode, FiPy can work as a access point for other devices to connect and also connect to another access point in order to access to the Internet.

The analysis algorithm is as following:

- 1) Analyzer retrieve the list of MAC addresses of all devices connected to FiPy access point with methon wlan.ap\_sta\_list() and a current timestamp.
- 2) Insert or update the MAC address and timestamp of a global connected device list (in the format of [{"mac address": "timestamp"}] ) and add the MAC address of new joined device to list called "newDev".
- 3) Check the timestamp of un-updated devices in the global connected device. If the difference of current timestamp with the timestamp of the device is larger than 5 minutes, remove the device form the global connected device list and add it into a list called "inactivateDev".
- 4) Return list "newDev" and "inactivateDev" as the result of the analysis.

In order to send message to telegram, we first need to create a telegram bot named "fipy\_analyzer". For the simplicity, the wifi analyzer will send alert messages to telegram by sending a POST request with chat id and message as parameter to the bot's URL. Since pycom does not provide us the micro-python package of http requests, we need to implement it by ourselve as "urequest.py" in library shows. Then, we can start a chat with the bot with a message "/start". And after that, the result of the analyzer will be sent telegram bot as Fig.5 shows.



**Fig.5 Notifications are sent to telegram bot when device joined FiPy network or found inactive**

## 4. Wifi analyzer with SigFox uplink

To create a wifi analyzer with FiPy and send message with SigFox uplink, we first need to configure the wlan on FiPy to AP mode and configure SigFox to uplink only.

The analysis algorithm will be the same as exercise 3.

After a proper configuration, we can use the send() method of socket to send message to SigFox backend. To notify someone when a new host is detected, we can create an uplink callback on SigFox backend (as Fig.6 shows). The callback is configured to send an e-mail when a data package is uploaded. In this callback configuration, a custom payload config is used to transform the first 12 bytes of raw payload to char which is easier for user to read. With these configurations, when a new host join the FiPy network, its MAC address will be found and sent to SigFox backend. And meanwhile, an e-mail will be sent to administrator's e-mail address. (as Fig.7 shows)

### Device type PYCOM\_DevKit\_1 - Callback edition

Callbacks

Type:

Channel:

Custom payload config:

Recipient:

Multiple emails allowed separated by comma, semicolon or new line

Subject syntax: Subject with device {device}  
Message syntax: Message containing time {time}, key1 {var1}, key2 {var2}...  
Available variables: device, time, data, seqNumber, deviceTypeId  
Custom variables: customData#mac


Subject:

Message:

Fig.6 SigFox uplink callback configuration

Time	Seq Num	Data / Decoding	LQI
2021-07-08 16:07:55	113	343832636130356533313139 ASCII: 482ca05e3119	
2021-07-08 15:58:05	112	343832636130356533313139 ASCII: 482ca05e3119	
2021-07-08 15:53:42	111	343832636130356533313139 ASCII: 482ca05e3119	

**New host connect to network of 4D6EF5**

 SIGFOX <backend-noreply@sigfox.com>  
7/8/2021 4:08 PM

To: jiafish1996@gmail.com

Device with mac: 482ca05e3119.  
Connected to 4D6EF5's network.

Fig.7 Messages sent to SigFox backend and admin's e-mail when a new host join FiPy's network