**********Draft**********

Adding a 2nd socket to Pico_W freertos iperf, MQTT
Mosquitto or Ultibo QEMU or Hardware Rpi Broker
To provide Debug information previously provided by a hard wired connection to the Pico_W UART
04/07/23

**********Draft**********

Mosquitto or Ultibo QEMU or Hardware Rpi Broker

RTC time setting
Build Steps

Mosquitto Broker
diff /usr/share/doc/mosquitto/examples/mosquitto.conf /etc/mosquitto/mosquitto.conf
512c512,522
< #allow_anonymous false

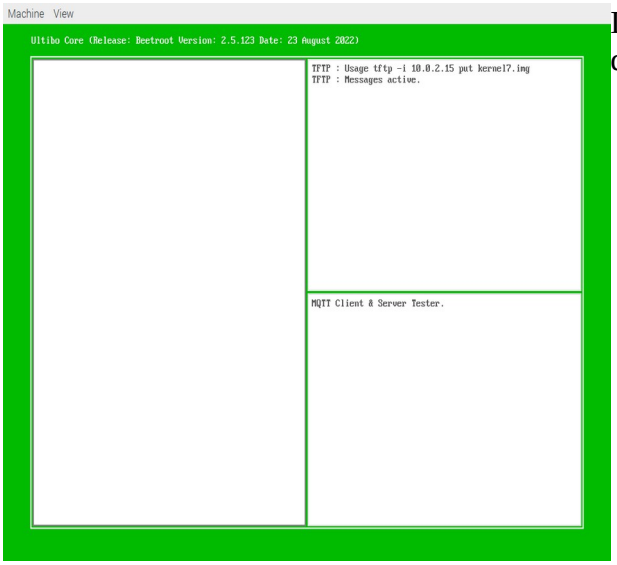
> #listener 8883 192.168.1.211
> #listener 1884 192.168.1.211
> listener 9883
> #listener 9883 192.168.1.175
> listener 1883
> user testuser
> per_listener_settings true
> #password_file /etc/mosqitto/mosquitto-pw
> password_file /home/devel/mosquitto-pw
> #acl_file file /etc/mosquitto/acl_file.conf
> allow_anonymous false
513a524
> #log_dest stdout

mosquitto -c /etc/mosquitto/mosquitto.conf
mosquitto_sub -t 'update/memo' -u 'testuser' -P 'password123'
mosquitto_sub -h pi4-60 -p 1883 -t 'update/memo' -u 'testuser' -P 'password123'

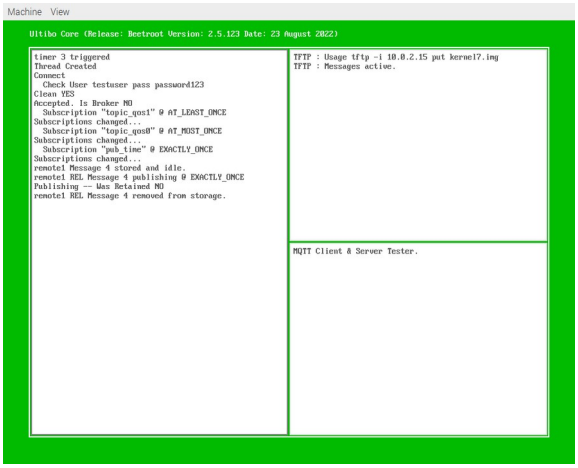
Ultibo QEMU “https://github.com/develone/Ultibo_Projects/tree/master/Pauls-ultibo-mqtt”
u16_t mqtt_port = 9883; instead of default u16_t mqtt_port = 1883;

devel@pi4-50:~/Ultibo_Projects/Pauls-ultibo-mqtt/QEMU $ ./startqemu.sh
qemu-system-arm: -net user,hostfwd=tcp::5080-:80,hostfwd=tcp::5023-:23,hostfwd=tcp::9883-:1883,hostfwd=udp::5069-:69,hostfwd=tcp::6050-:5050: Could not set up host forwarding rule 'tcp::9883-:1883'

Starts a 3 pane Window. First step would be depressing '5' : MQ.Activate (true);



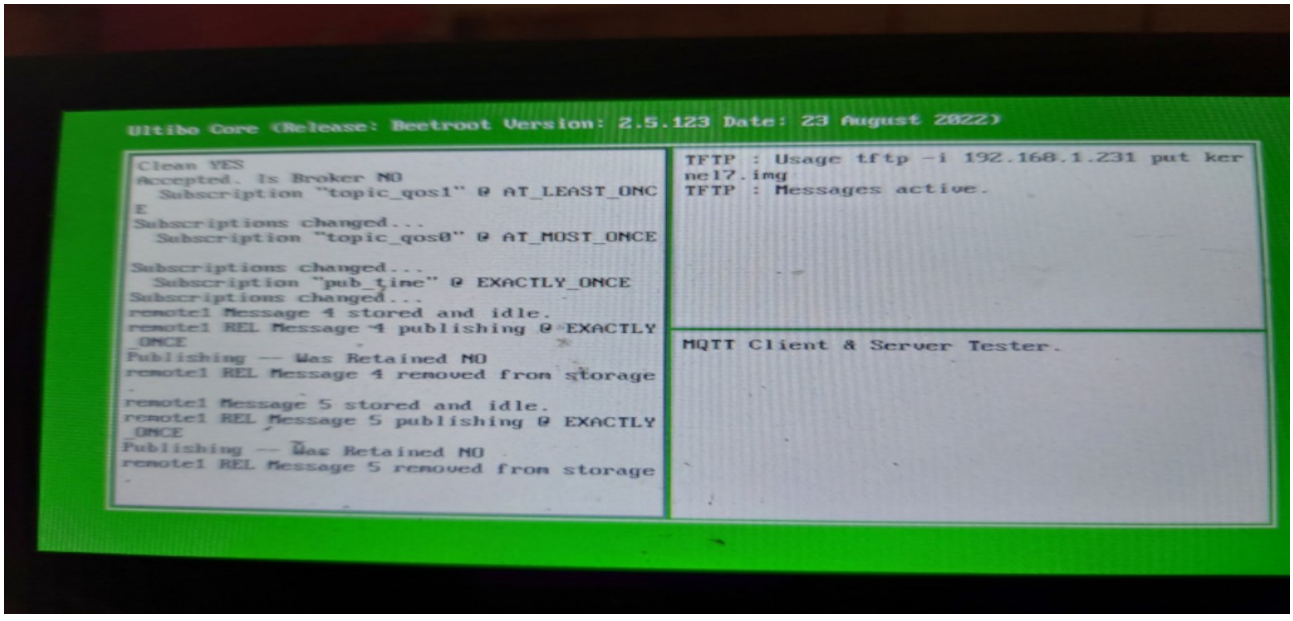
Data starts filling the left pane following the depressing ‘5’



/*192.168.1.212 0xc0a801d4 LWIP_MQTT_EXAMPLE_IPADDR_INIT pi4-50*/
#define LWIP_MQTT_EXAMPLE_IPADDR_INIT = IPADDR4_INIT(PP_HTONL(0xc0a801d4))

/*192.168.1.231 0xc0a801d4 LWIP_MQTT_EXAMPLE_IPADDR_INIT ultibo*/
#define LWIP_MQTT_EXAMPLE_IPADDR_INIT = IPADDR4_INIT(PP_HTONL(0xc0a801e7))

This is a Ultibo RPi3B with 7in display.



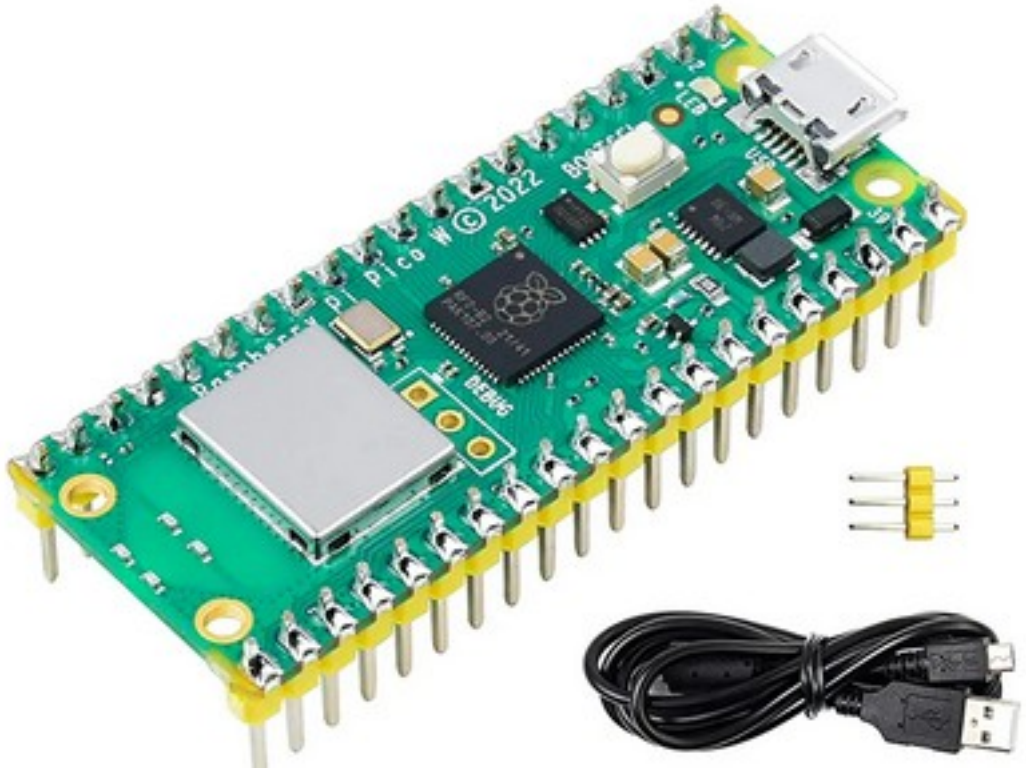
RTC time setting
In the process of converting my “https://github.com/develone/pico_w-remotes.git”.

This version used ntp for setting the RTC in Pico_W. The new version "https://github.com/develone/pico_w-mqtt.git" uses a RPI to publish date and time information to topic 'pub_time' and the Pico_W subscribes to topic 'pub_time'.
../pub-time pi4-50
2023-04-07-05-38-18

In the function "mqtt_incoming_data_cb" parses the received time information and sets the Pico_W RTC.
t 0x0 &t 0x0 *pt 0x200220a0
t_ntp 0x0 &pt_ntp 0x0 *pt_ntp 0x200220dc
2023
04
07
05
38
18
2023-04-07-05-38-18
2023/04/07 05:38:27

Time information is reported to users using tcp_debug socket.
../pi_tcp_tests/cli1
Socket created successfully
Connected with server successfully
Starting FreeRTOS on core 0: ver 0.0.02 remote1
Connecting to Wi-Fi...
Connected. iperf server 192.168.1.176 4001
starting watchdog timer task
mqtt_ip = 0xd401a8c0 mqtt_port = 1883
mqtt_connect 0x0 mqtt_connect 0x1
2023-04-07-05-38-18
2023/04/07 05:38:2nnect 0x1
2023/04/07 05:42:37
mqtt_connect 0x1 mqtt_connect 0x1
40:57
mqtt_connect 0x1 mqtt_connect 0x1
2023/04/07 05:41:22
mqtt ���

I have several pico_w connected to my home Wifi. Currently a 512 byte debug is sent to RPi4-4GB using (cli1, cli2, cli3, cli4, cli5, and cli6).

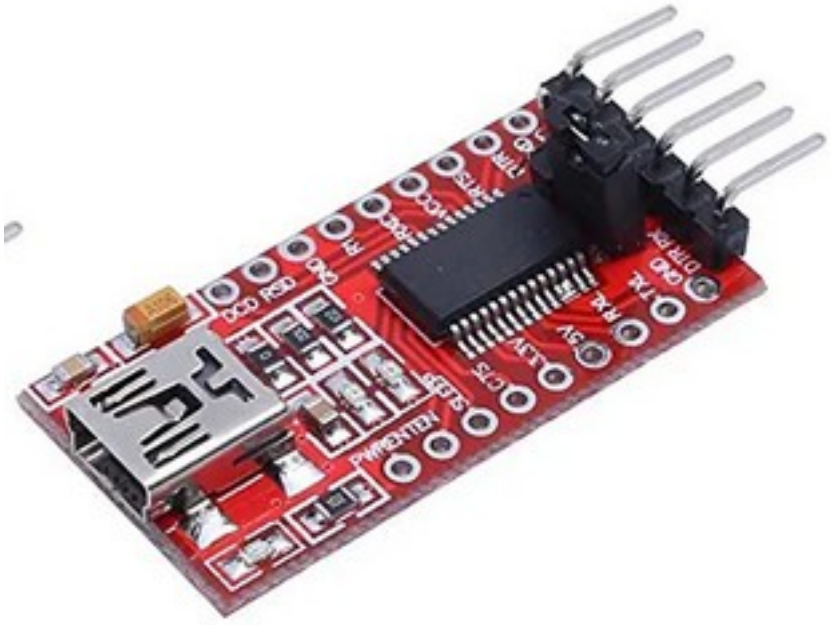


Build Steps
“git clone https://github.com/develone/pico_w-mqtt.git -b dev”
“cd pico_w-mqtt”
Modify the script “6remotes.sh” WIFI_SSID with your SSID and WIFI_PASSWORD with your PASSWORD.
Modify the file “pico_w/wifi/freertos/iperf/picow_freertos_iperf.c” WIFI_PASSWORD with your PASSWORD.
“./6remotes.sh” creates 6 copies of the program “remotex/pico_w/wifi/freertos/iperf/picow_freertos_iperf_server_mqtt.elf” each with a different hostname. In addition copies “exe-ocd.sh” to each of the six folders remotex.
“exe-ocd.sh” uses openocd to program the Pico_W
#!/bin/bash
openocd -f interface/raspberrypi-swd.cfg -f target/rp2040.cfg -c "program pico_w/wifi/freertos/iperf/picow_freertos_iperf_server_mqtt.elf verify reset exit"

It also runs the script “build_cli.sh”.
The script “build_cli.sh” creates 6 programs (cli1, cli2, cli3, cli4, cli5, and cli6) in the folder pi_tcp_tests.

```
#!/bin/bash

cd pi_tcp_tests

rm -f cli1 cli2 cli5 cli6

gcc -v client.c -Drem1  -o cli1
gcc -v client.c -Drem2  -o cli2
gcc -v client.c -Drem3  -o cli3
gcc -v client.c -Drem4  -o cli4

gcc -v client.c -Drem5  -o cli5
gcc -v client.c -Drem6  -o cli6
```

The USB to UART is currently used to see the debug from pico_w. This will be removed and debug will be available using programs (cli1, cli2, cli3, cli4, cli5, and cli6).



and connected
to the RPi4B
4Gb USB to
see the debug output.

Now this can be done with the programs (cli1, cli2, cli3, cli4, cli5, and cli6).

Examples of the programming & debug are found
“https://github.com/develone/pico_w-mqtt/blob/dev/doc/info.txt”.
Modified output “https://github.com/develone/pico_w-mqtt/blob/dev/doc/info_1.txt”.

The buffer now is 512 bytes. The first 256 is used for booting information and the next 256 are used following the connection to WiFi. Note: mqtt_connected 0 then mqtt_connected 1 which is when the connection to the Mosquitto Broker.

```
 devel@pi4-30:~/pico_w-mqtt/remote5 $ ../pi_tcp_tests/cli1
Socket created successfully
Connected with server successfully
Starting FreeRTOS on core 0: ver 0.0.02 remote1
Connecting to Wi-Fi...
Connected. iperf server 192.168.1.176 4001
starting watchdog timer task
mqtt_ip = 0xd401a8c0 mqtt_port = 1883
mqtt_connect 0x0 mqtt_connect 0x1
2023-04-07-05-38-18
2023/04/07 05:38:2nnect 0x1
2023/04/07 05:42:37
mqtt_connect 0x1 mqtt_connect 0x1
40:57
mqtt_connect 0x1 mqtt_connect 0x1
2023/04/07 05:41:22
mqtt ���
```