



ELETRÔNICA EMBARCADA COM MICROCONTROLADORES

ATIVIDADE 1 - TRANSMITINDO DADOS VIA MQTT

PROFESSOR GUSTAVO FERREIRA PALMA

PÓS-GRADUANDA LIZ LENE DO PRADO PINTO

Porto Alegre, 01 de maio de 2024.

ATIVIDADE 1 - TRANSMITINDO DADOS VIA MQTT

Utilizando os conceitos abordados em aula, crie um firmware que seja capaz de enviar a temperatura do núcleo do micro controlador via MQTT para um tópico a sua escolha. O envio deve ser feito sempre que o botão da placa Raspberry Pi Pico for pressionado.

IMPORTANTE!

- Os códigos devem ser desenvolvidos utilizando o SDK padrão C/C++ da Raspberry Pi Pico;
- Apenas o Link para o projeto no github deve ser incluído na tarefa;
- A entrega deve obedecer o prazo estipulado
- O projeto base disponibilizado na página do curso pode ser utilizado como base no desenvolvimento dessa atividade

Com base no exemplo da Aula 02, foi realizada a execução da Atividade 1, integrando o Software Visual Studio Code e o protocolo MQTT com o MQTT Client Toolbox para o envio e recebimento de mensagens pela internet. Onde, a cada clique no botão da Raspberry, a temperatura do núcleo do micro controlador é mostrada via MQTT.

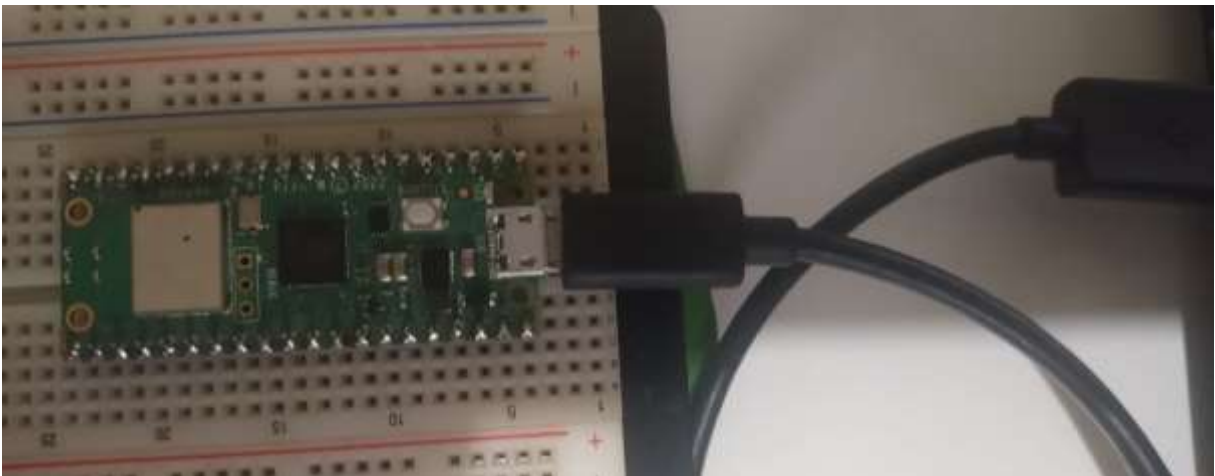


Figura 1 – Raspberry Pi Pico W

1) Descrição do código do arquivo main.c

```
1. #include <stdio.h>
2. #include <string.h>
3. #include "pico/stdlib.h"
4. #include "pico/cyw43_arch.h"
5. #include "bsp/board.h"
6. #include "hardware/gpio.h"
7. #include "hardware/adc.h"
8. #include "hardware/structs/adc.h"
9.
```

```

10. #include "lwip/apps/mqtt.h"
11.
12. #define WIFI_SSID "RESOORTS 2.4G"
13. #define WIFI_PASSWORD "D3&M4@Yi5S"
14. #define MQTT_SERVER "54.146.113.169" //"broker.emqx.io"
15.
16.
17. struct mqtt_connect_client_info_t mqtt_client_info=
18. {
19.     "<RA>/pico_w", /*client id*/
20.     NULL, /* user */
21.     NULL, /* pass */
22.     0, /* keep alive */
23.     NULL, /* will_topic */
24.     NULL, /* will_msg */
25.     0, /* will_qos */
26.     0 /* will_retain */
27. #if LWIP_ALTCP && LWIP_ALTCP_TLS
28.     , NULL
29. #endif
30. };
31.
32. static void mqtt_incoming_data_cb(void *arg, const u8_t *data, u16_t len,
    u8_t flags) {
33.     printf("data: %s\n",data);
34.     /*char led[30];
35.     memset(led, "\0", 30);
36.     strncpy(led, data, len);
37.
38.     if (strncmp(led, "on", 2) == 0) {
39.         cyw43_arch_gpio_put(CYW43_WL_GPIO_LED_PIN, true);
40.     } else if (strncmp(led, "off", 3) == 0) {
41.         cyw43_arch_gpio_put(CYW43_WL_GPIO_LED_PIN, false);
42.     }*/
43. }
44.
45. static void mqtt_incoming_publish_cb(void *arg, const char *topic, u32_t
    tot_len) {
46.     printf("topic %s\n", topic);
47. }
48.
49. static void mqtt_request_cb(void *arg, err_t err) {
50.     printf(("MQTT client request cb: err %d\n", (int)err));
51. }
52.
53. static void mqtt_connection_cb(mqtt_client_t *client, void *arg,
    mqtt_connection_status_t status) {
54.     printf(("Conectado ao Broker MQTT %d\n", (int)status));
55.     /*if (status == MQTT_CONNECT_ACCEPTED) {

```

```

56.     err_t erro = mqtt_sub_unsub(client, "gustavo/led", 0, &mqtt_request_cb,
    NULL, 1);
57.
58.     if (erro == ERR_OK)
59.     {
60.         printf("Inscrito com Sucesso!\n");
61.     } else {
62.         printf("Falha ao Inscrever!\n");
63.     }
64. } else {
65.     printf("Conexão rejeitada!\n");
66. }*/
67. }
68.
69. int main() {
70.
71.     // Inicializa o UART para saída serial
72.     stdio_init_all();
73.     printf("Iniciando...\n");
74.
75.     // Inicializa a biblioteca Pico SDK
76.     adc_init();
77.     adc_set_temp_sensor_enabled(true);
78.     adc_select_input(4);
79.
80.     if (cyw43_arch_init())
81.     {
82.         printf("Falha ao iniciar chip wifi\n");
83.         return 1;
84.     }
85.
86.     cyw43_arch_enable_sta_mode();
87.     printf("Conectando ao %s\n", WIFI_SSID);
88.
89.     if (cyw43_arch_wifi_connect_timeout_ms(WIFI_SSID, WIFI_PASSWORD,
    CYW43_AUTH_WPA2_AES_PSK, 30000))
90.     {
91.         printf("Falha ao conectar\n");
92.         return 1;
93.     }
94.     printf("Conectado ao Wifi %s\n", WIFI_SSID);
95.
96.     ip_addr_t addr;
97.     if(!ip4addr_aton(MQTT_SERVER, &addr)){
98.         printf("ip error\n");
99.         return 1;
100.    }
101.
102.    printf("Conectando ao MQTT\n");

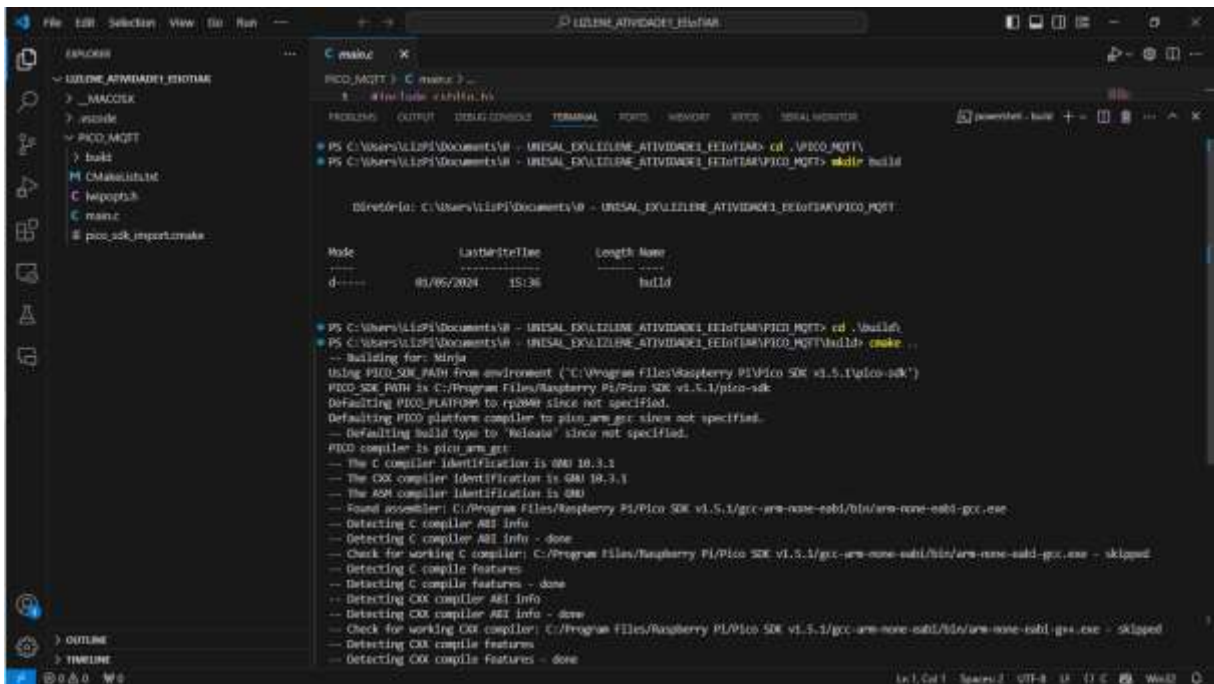
```

```
103.
104.     mqtt_client_t* cliente_mqtt = mqtt_client_new();
105.
106.     mqtt_set_inpub_callback(cliente_mqtt, &mqtt_incoming_publish_cb,
        &mqtt_incoming_data_cb, NULL);
107.
108.     err_t erro = mqtt_client_connect(cliente_mqtt, &addr, 1883,
        &mqtt_connection_cb, NULL, &mqtt_client_info);
109.
110.     if (erro != ERR_OK)
111.     {
112.         printf("Erro de conexão\n");
113.         return 1;
114.     }
115.
116.     printf("Conectado ao MQTT!\n");
117.
118.     while(true) {
119.         //Realiza a leitura do sensor de temperatura
120.         uint16_t raw = adc_read();
121.
122.         // Converte a leitura bruta para temperatura em Celsius
123.         float temperature = (raw / 4095.0f) * 3.3f * 100.0f;
124.
125.         //Converte e imprime o valor da temperatura
126.         char temp_str[20];
127.         snprintf(temp_str, sizeof(temp_str), "Temp_Celsius: %.2f\n",
            temperature);
128.         int bytes = strlen(temp_str);
129.
130.         if (board_button_read()) {
131.             mqtt_publish(cliente_mqtt, "liz_lene/button", temp_str, bytes, 0,
                false, &mqtt_request_cb, NULL);
132.             /*} else {
133.                 mqtt_publish(cliente_mqtt, "liz_lene/button", "off", 3, 0, false,
                    &mqtt_request_cb, NULL);
134.             }*/
135.             sleep_ms(1000); // Espera 2 segundos antes da próxima leitura
136.         }
137.     }
138.
139. }
```

2) Descrição do código do arquivo CMakeLists.txt

```
1) cmake_minimum_required(VERSION 3.12)
2)
3) include(pico_sdk_import.cmake)
4)
5) project(mqtt_pico C CXX ASM)
6)
7) set(CMAKE_C_STANDARD 11)
8) set(CMAKE_CXX_STANDARD 17)
9)
10) #Caso seja a Raspberry Pi Pico W
11) set(PICO_BOARD pico_w)
12)
13) pico_sdk_init()
14)
15) add_executable(mqtt_pico main.c)
16)
17) pico_enable_stdio_usb(mqtt_pico 1)
18) pico_enable_stdio_uart(mqtt_pico 0)
19)
20) target_include_directories(mqtt_pico PRIVATE
21)     ${CMAKE_CURRENT_LIST_DIR}
22) )
23)
24) target_include_directories(mqtt_pico PUBLIC
25)     ${CMAKE_CURRENT_LIST_DIR}/pico-sdk/src
26)     ${CMAKE_CURRENT_LIST_DIR}/pico-sdk/src/rp2_common/hardware_adc
27) )
28)
29) target_link_libraries(mqtt_pico
30)     tinyusb_board
31)     pico_cyw43_arch_lwip_threadsafe_background
32)     pico_stdlib
33)     pico_lwip_mqtt
34) )
35)
36) pico_add_extra_outputs(mqtt_pico)
37)
38) target_link_libraries(mqtt_pico hardware_adc)
39)
```

3) Preparação da estrutura e compilação do código



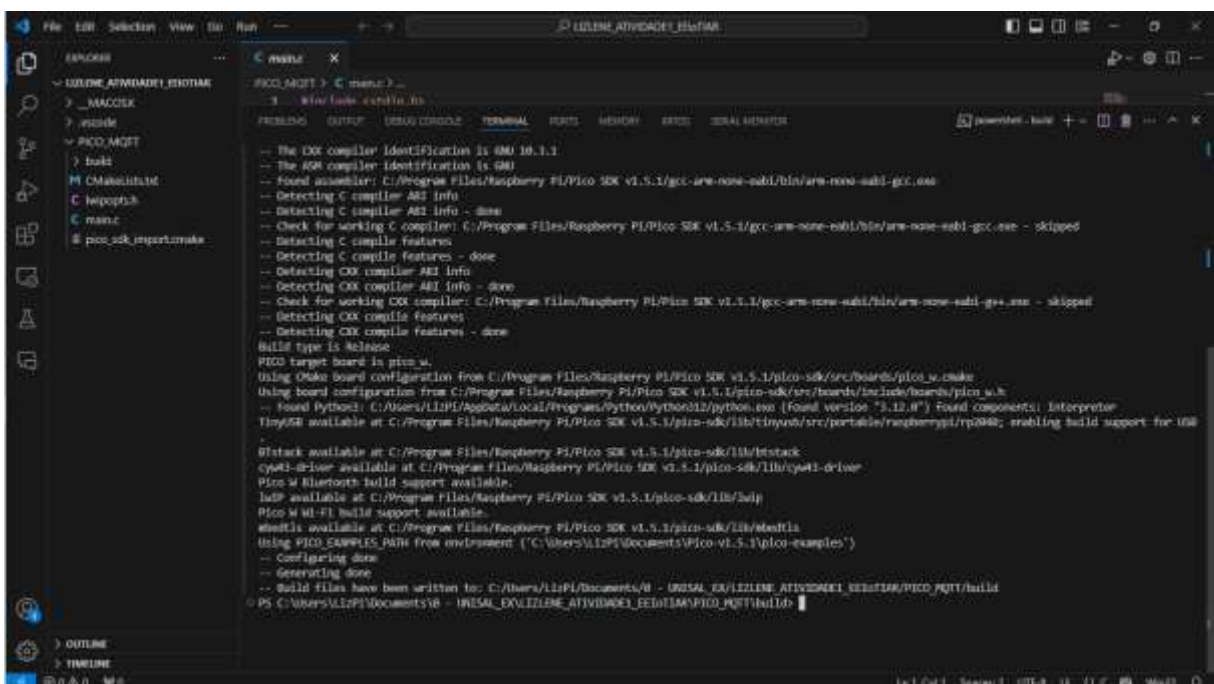
```
PS C:\Users\lilip\Documents\B - UNISA_EX\LINE_ATIVIDADES_FEIOTIM\PICO_MQTT> cd .\PICO_MQTT\
PS C:\Users\lilip\Documents\B - UNISA_EX\LINE_ATIVIDADES_FEIOTIM\PICO_MQTT> make build

Diretório: C:\Users\lilip\Documents\B - UNISA_EX\LINE_ATIVIDADES_FEIOTIM\PICO_MQTT

Mode                LastWriteTime         Length Name
----                -
d-----          01/05/2024      15:36         build

PS C:\Users\lilip\Documents\B - UNISA_EX\LINE_ATIVIDADES_FEIOTIM\PICO_MQTT> cd .\build\
PS C:\Users\lilip\Documents\B - UNISA_EX\LINE_ATIVIDADES_FEIOTIM\PICO_MQTT\build> make ..
-- Building for: Minja
Using PICO_SDK_PATH from environment ("C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk")
PICO_SDK_PATH is C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk
Defaulting PICO_PLATFORM to rp2040 since not specified.
Defaulting PICO_COMPILER to gcc since not specified.
-- Defaulting build type to 'Release' since not specified.
PICO compiler is gcc:
-- The C compiler identification is GNU 10.3.1
-- The CXX compiler identification is GNU 10.3.1
-- The ASM compiler identification is GNU
-- Found assembler: C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\gcc-arm-none-eabi\bin\arm-none-eabi-gcc.exe
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\gcc-arm-none-eabi\bin\arm-none-eabi-gcc.exe - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\gcc-arm-none-eabi\bin\arm-none-eabi-g++.exe - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
```

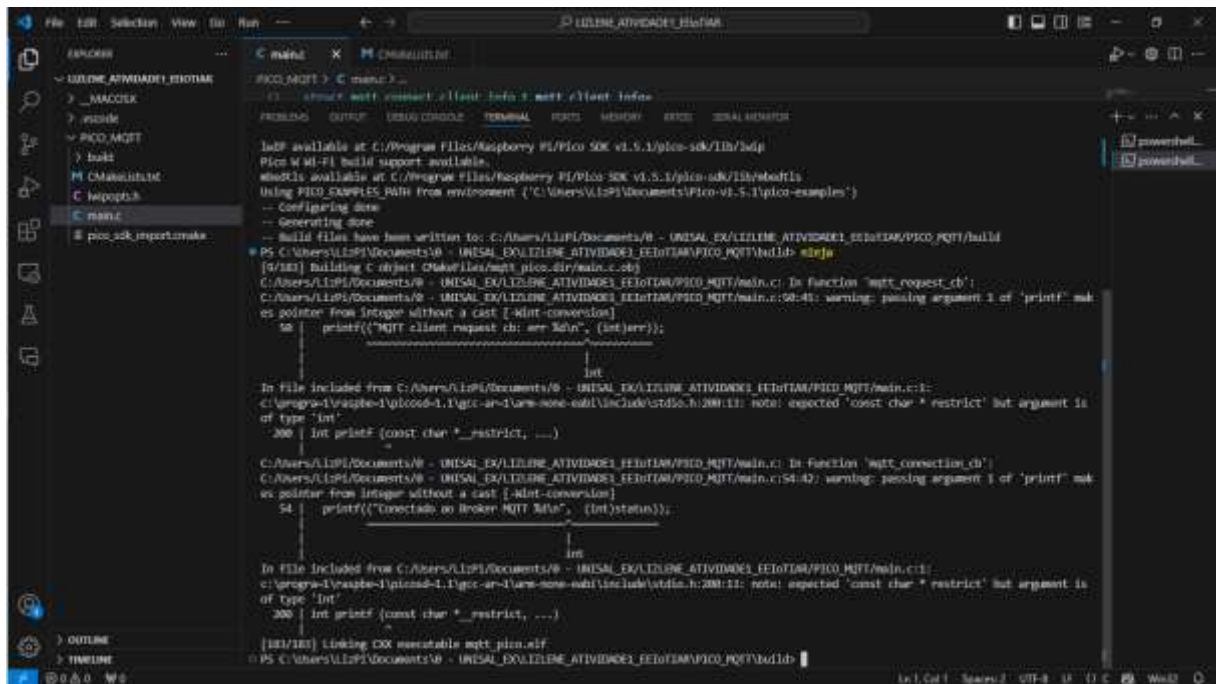
Figura 2 – Compilação



```
-- The CXX compiler identification is GNU 10.3.1
-- The ASM compiler identification is GNU
-- Found assembler: C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\gcc-arm-none-eabi\bin\arm-none-eabi-gcc.exe
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\gcc-arm-none-eabi\bin\arm-none-eabi-gcc.exe - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\gcc-arm-none-eabi\bin\arm-none-eabi-g++.exe - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
Build type is Release
PICO target board is pico w.
Using Make board configuration from C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk/src/boards/pico_w.make
Using board configuration from C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk/src/boards/include/boards/pico_w.h
-- Found Python3: C:/Users/lilip/AppData/Local/Programs/Python/Python312/python.exe (found version "3.12.0") Found components: Interpreter
TinyUSB available at C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk/lib/tinyusb/src/portable/raspberrypi/rp2040; enabling build support for USB
--
Hoststack available at C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk/lib/hoststack
Cyw43-driver available at C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk/lib/cyw43-driver
Pico W Bluetooth build support available.
LWIP available at C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk/lib/lwip
Pico W Wi-Fi build support available.
mbedtls available at C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk/lib/mbedtls
Using PICO_EXAMPLES_PATH from environment ("C:\Users\lilip\Documents\Pico v1.5.1\pico-examples")
-- Configuring done
-- Generating done
-- Build files have been written to: C:\Users\lilip\Documents\B - UNISA_EX\LINE_ATIVIDADES_FEIOTIM\PICO_MQTT\build
PS C:\Users\lilip\Documents\B - UNISA_EX\LINE_ATIVIDADES_FEIOTIM\PICO_MQTT\build
```

Figura 3 – Continuação da compilação

4) Output do código



```
PICO_MQTT > C main.c ...
C:\Program Files\Kasparov PI\Pico SDK v1.5.1/pico-sdk/lib/build
Pico W Wi-Fi build support available.
Pico W Wi-Fi build support available.
Using PICO_EXAMPLES_PATH from environment ('C:\Users\lisp\Documents\Pico-W-5.1/pico-examples')
-- Configuring done
-- Generating done
-- Build files have been written to: C:\Users\lisp\Documents\W - UNISAL_EX\LINE_ATIVIDADES_FEIOTIA\PICO_MQTT/build
PS C:\Users\lisp\Documents\W - UNISAL_EX\LINE_ATIVIDADES_FEIOTIA\PICO_MQTT/build> ninja
[9/10] Building C object C:\Users\lisp\Documents\W - UNISAL_EX\LINE_ATIVIDADES_FEIOTIA\PICO_MQTT/build/main.o
C:\Users\lisp\Documents\W - UNISAL_EX\LINE_ATIVIDADES_FEIOTIA\PICO_MQTT/main.c: In function 'mqtt_request_cb':
C:\Users\lisp\Documents\W - UNISAL_EX\LINE_ATIVIDADES_FEIOTIA\PICO_MQTT/main.c:58:41: warning: passing argument 1 of 'printf' makes
pointer from integer without a cast [-Wint-conversion]
58 | printf("MQTT client request cb: err %d\n", (int)err);
    | ~~~~~^~~~~~
    |          |
    |          int
In file included from C:\Users\lisp\Documents\W - UNISAL_EX\LINE_ATIVIDADES_FEIOTIA\PICO_MQTT/main.c:1:
c:\program-files\kasperov-pi\pico-sdk-1.1\pic-ar-v1\arm-none-eabi\include\stdio.h:200:11: note: expected 'const char * restrict' but argument is
of type 'int'
200 | int printf(const char *__restrict, ...)
    | ~~~~~^~~~~~
C:\Users\lisp\Documents\W - UNISAL_EX\LINE_ATIVIDADES_FEIOTIA\PICO_MQTT/main.c: In function 'mqtt_connection_cb':
C:\Users\lisp\Documents\W - UNISAL_EX\LINE_ATIVIDADES_FEIOTIA\PICO_MQTT/main.c:54:42: warning: passing argument 1 of 'printf' makes
pointer from integer without a cast [-Wint-conversion]
54 | printf("Connected on Broker MQTT %s\n", (int)status);
    | ~~~~~^~~~~~
    |          |
    |          int
In file included from C:\Users\lisp\Documents\W - UNISAL_EX\LINE_ATIVIDADES_FEIOTIA\PICO_MQTT/main.c:1:
c:\program-files\kasperov-pi\pico-sdk-1.1\pic-ar-v1\arm-none-eabi\include\stdio.h:200:11: note: expected 'const char * restrict' but argument is
of type 'int'
200 | int printf(const char *__restrict, ...)
    | ~~~~~^~~~~~
[10/10] Linking CXX executable mqtt_pico.uf2
PS C:\Users\lisp\Documents\W - UNISAL_EX\LINE_ATIVIDADES_FEIOTIA\PICO_MQTT/build>
```

Figura 4 – Utilizando o comando “ninja”, para gerar o arquivo .uf2

5) Testando a integração para verificar se funciona o código

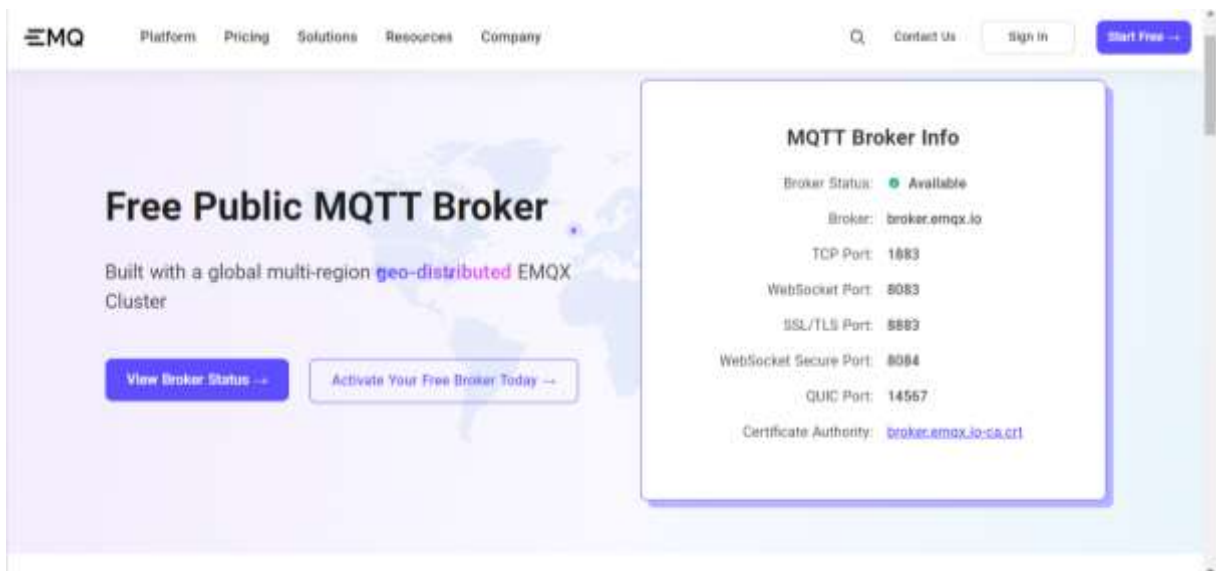


Figura 5 – MQTT Broker Info com os dados necessários para a comunicação na internet

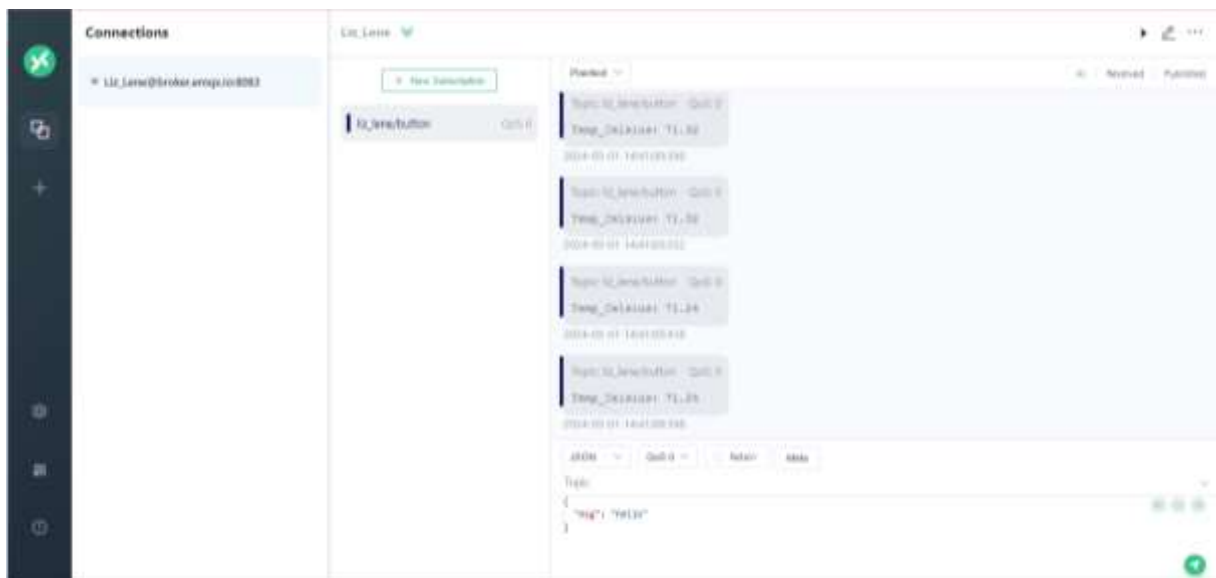


Figura 6 – Emqx.io/online-mqtt-client# mostrando o resultado ao pressionar o botão

6) Referência bibliográfica

6.1) https://ava.ead.unisal.br/pluginfile.php/60694/mod_resource/content/1/Aula-2.pdf

6.2) <https://www.youtube.com/watch?v=ZM6bmV93UvY>

6.3) https://github.com/raspberrypi/pico-examples/blob/master/adc/hello_adc/hello_adc.c

6.4) <https://capsistema.com.br/index.php/2021/07/12/leia-o-valor-do-sensor-de-temperatura-do-raspberry-pi-pico/>

6.5) <https://github.com/Engenharia-entendida/RaspberryPiPico/blob/Aula%2306/main.py>

6.6) <https://elcereza.com/raspberry-pi-pico-vale-a-pena-aprender/>