



WEBINAR
EMBARCADOS



mouser.com

METODOLOGIA DE DESENVOLVIMENTO IOT: **DEBUG E PROJETOS** **COM ESP32-S3**



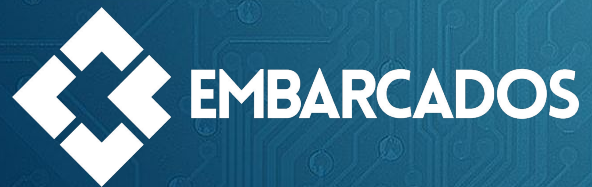
Luiz Pedro Bittencourt

Desenvolvedor de
Firmware, Control ID



Marcelo Barros

Professor, Faculdade de
Engenharia Elétrica da UFU



mouser.com

Agenda

1. Formação em Sistemas Embarcados
2. Aprendizagem baseada em projetos (PBL)
3. Projeto desenvolvido seguindo o modelo PBL

Problemas atuais na formação em sistemas embarcados

- **Os cursos universitários tem dificuldade em preparar o discente para o mercado de trabalho!**
 - Formação mais generalista
 - Natureza multidisciplinar da área
- **Os melhores profissionais geralmente tomam ações adicionais:**
 - Fazem estágios ou trabalham na área de embarcados
 - Desenvolvem projetos próprios
 - Participam de competições ou equipes de competição
 - Tem formação técnica do segundo grau
 - Participam de projetos de pesquisa

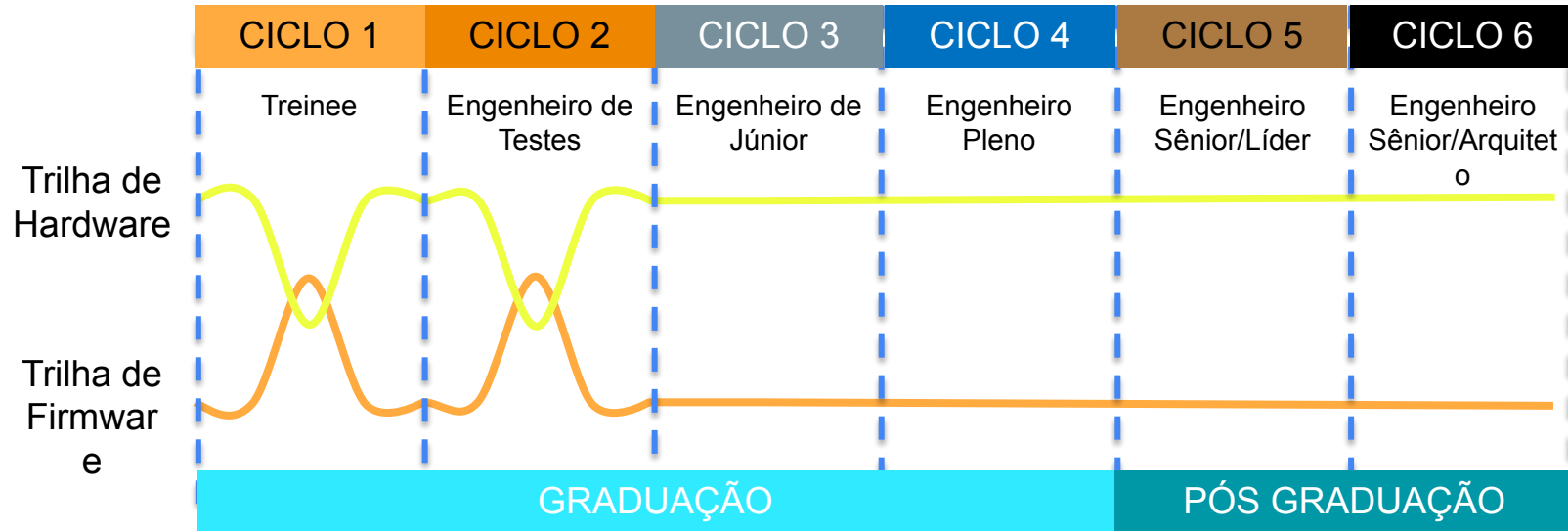
Problemas atuais na formação em sistemas embarcados

- **Possível estratégia: PBL (Project Based Learning) (*)**
 - Permite unir conhecimento de áreas diversas
 - Ataca problemas reais
 - Trabalho em equipe é essencial
- **Difícil implementar na estrutura atual da universidade pública**
- **Mas é uma realidade de vários laboratórios de pesquisa !**

Problemas atuais na formação em sistemas embarcados

- **Pesquisa aplicada e PBL: um caminho viável**
- **Falta apenas a sistematização !**
- **Como estamos implantando:**
 - Um laboratório ativo em pesquisa aplicada
 - Curadoria de disciplinas
 - Disciplinas optativas
 - Trilhas de conhecimento bem definidas

Trilhas de Conhecimento

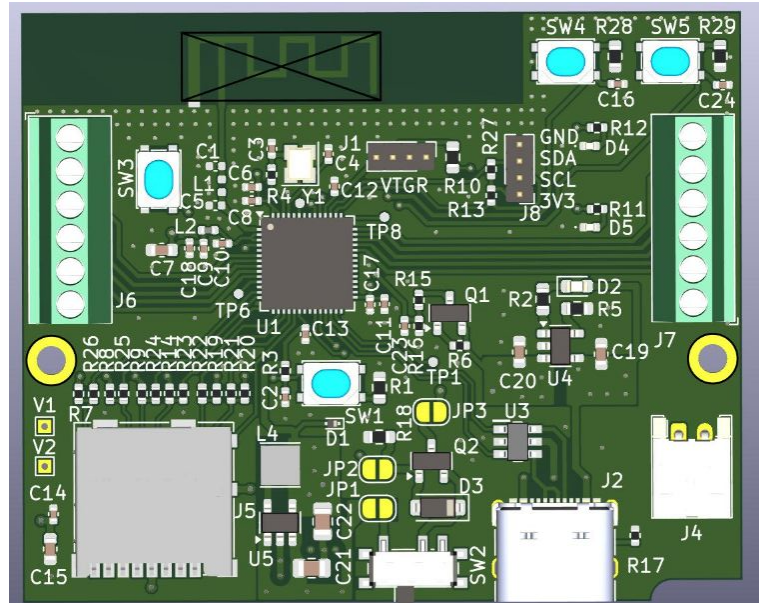


Trilhas de Conhecimento

Competência	Grupo	Tripla	Treinado	Júnior	Pleno	Sênior	Líder	Arquiteto	Evidência
Construção de firmware com Schedulers cooperativos	Arquitetura de Firmware	Firmware	Não	Desenvolvendo	Sim	Sim	Sim	Sim	Projeto
Construção de firmware com Corrotinas	Arquitetura de Firmware	Firmware	Não	Não	Desenvolvendo	Sim	Sim	Sim	Projeto
Separação em camadas (BSP / middleware / aplicação)	Arquitetura de Firmware	Firmware	Não	Não	Desenvolvendo	Sim	Sim	Sim	Projeto
Criação de HALs	Arquitetura de Firmware	Firmware	Não	Não	Desenvolvendo	Sim	Sim	Sim	Projeto
Padrões de projeto para sistemas embarcados	Arquitetura de Firmware	Firmware	Não	Não	Não	Desenvolvendo	Sim	Sim	Projeto
Refactoring e manutenção de firmware legado	Arquitetura de Firmware	Firmware	Não	Não	Não	Desenvolvendo	Sim	Sim	Projeto
Versionamento com Git	Build e DevOps	Firmware e Hardware	Desenvolvendo	Sim	Sim	Sim	Sim	Sim	PBL
Make / Makefile	Build e DevOps	Firmware	Desenvolvendo	Sim	Sim	Sim	Sim	Sim	PBL
CMake	Build e DevOps	Firmware	Não	Desenvolvendo	Sim	Sim	Sim	Sim	PBL
GCC ARM Toolchain	Build e DevOps	Firmware	Não	Desenvolvendo	Sim	Sim	Sim	Sim	PBL
Docker	Build e DevOps	Firmware	Não	Não	Desenvolvendo	Sim	Sim	Sim	Projeto
C/CD para firmware	Build e DevOps	Firmware	Não	Não	Não	Desenvolvendo	Sim	Sim	Projeto
Captura de esquemas simples (folha única)	Captura de Esquemas e Design de PCB	Hardware	Não	Desenvolvendo	Sim	Sim	Sim	Sim	Projeto
Captura de esquemas complexos (múltiplas folhas)	Captura de Esquemas e Design de PCB	Hardware	Não	Não	Desenvolvendo	Sim	Sim	Sim	Projeto
Captura de esquemas avançados (impedância e diferenciais)	Captura de Esquemas e Design de PCB	Hardware	Não	Não	Não	Desenvolvendo	Sim	Sim	Projeto
Geração de BOM padronizada	Captura de Esquemas e Design de PCB	Hardware	Não	Desenvolvendo	Sim	Sim	Sim	Sim	Projeto
Cotação de componentes (Mouser e DigiKey)	Captura de Esquemas e Design de PCB	Hardware	Desenvolvendo	Sim	Sim	Sim	Sim	Sim	Projeto
Criação de símbolos e footprints	Captura de Esquemas e Design de PCB	Hardware	Não	Desenvolvendo	Sim	Sim	Sim	Sim	Projeto
Layout PCB 2 layers	Captura de Esquemas e Design de PCB	Hardware	Não	Não	Desenvolvendo	Sim	Sim	Sim	Projeto
Layout PCB 4 layers com controle de impedância	Captura de Esquemas e Design de PCB	Hardware	Não	Não	Não	Desenvolvendo	Sim	Sim	Projeto
Layout PCB 6+ layers (alta densidade)	Captura de Esquemas e Design de PCB	Hardware	Não	Não	Não	Não	Desenvolvendo	Sim	Projeto
Uso e projeto com JTAG / SWD	Depuração e Diagnóstico	Firmware e Hardware	Desenvolvendo	Sim	Sim	Sim	Sim	Sim	PBL
Depuração com ODB	Depuração e Diagnóstico	Firmware	Desenvolvendo	Sim	Sim	Sim	Sim	Sim	PBL
Trace de execução (ITM, ETM, SEGGER RTT)	Depuração e Diagnóstico	Firmware	Não	Desenvolvendo	Sim	Sim	Sim	Sim	PBL

Projeto desenvolvido e contexto do problema

Ferramenta Embarcada para Depuração e Logs



O Ponto Cego da Engenharia Embarcada

Falhas Intermitentes

Surtem apenas após dias ou semanas de operação real no cliente.

Testes Inviáveis

Testes em campo sem instrumentação geram dados pouco confiáveis e custosos.

Falta de Telemetria

A ausência de registro contínuo deixa os desenvolvedores 'cegos' em relação à causa raiz.

Motivação e Objetivos do Datalogger

Motivação

Falta de ferramentas para depurar sistemas legados sem alterar o firmware original do dispositivo monitorado.

Objetivo Geral

Desenvolver um datalogger transparente para monitoramento, testes assíncronos e telemetria de longo prazo.

Recursos Principais

- Captura de logs UART e GPIO
- Redundância SD local e MQTT
- Controle remoto bidirecional

Arquitetura de Hardware

Microcontrolador

ESP32-S3 Xtensa
Dual-Core 32-bit LX7
operando em até 240
MHz.

Memória Nativa

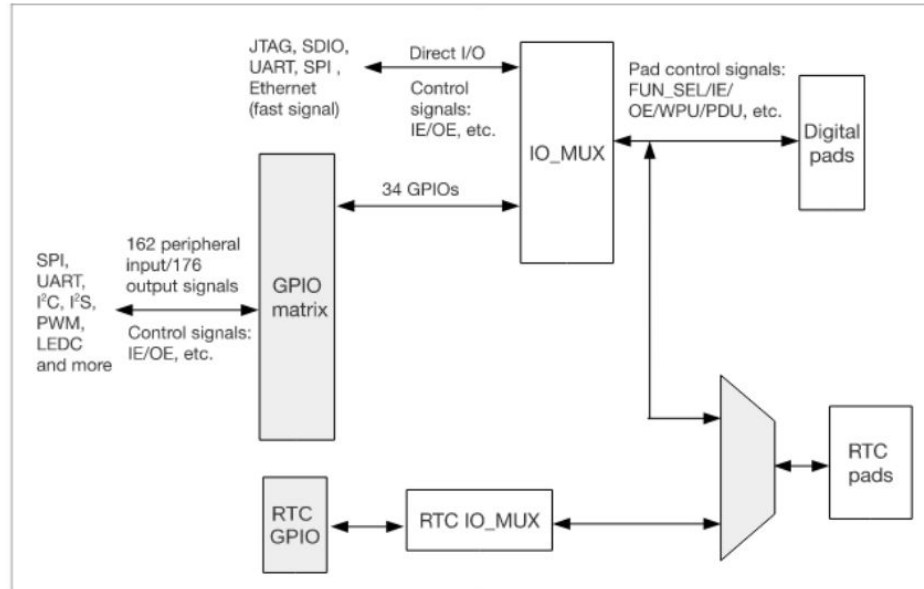
4MB de Memória
Flash interna e 2MB
de PSRAM / SRAM
para buffers.

Conectividade

Wi-Fi e Bluetooth LE
integrados no próprio
SoC, reduzindo
componentes
externos.

Características Fundamentais da Plataforma

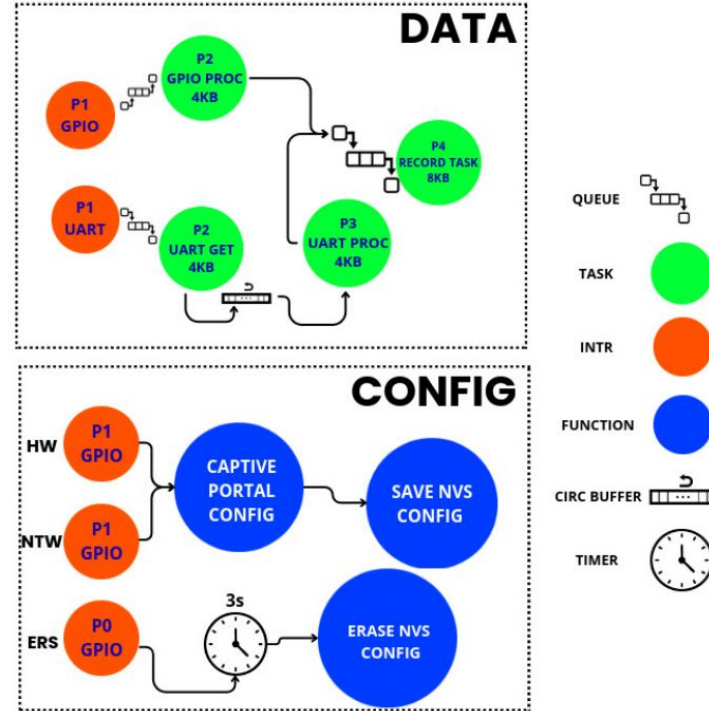
- Wi-Fi integrado
- Matriz IO/MUX permite reatribuir sinais via software
- Framework maduro com RTOS integrado em seus drivers.



Arquitetura de firmware e aplicação

- **Firmware desenvolvido para ser modular e desacoplado:**
 - A falha de um periférico não pode atrapalhar o funcionamento de outro.
 - Um código sempre deve ser idealizado com a ideia de “isso precisa ser fácil de dar manutenção”.
 - Módulos devem ser fáceis de testar e imaginar cenários de testes.
- **Camadas de serviço:**
 - Camada de Conectividade: Wi-Fi e MQTT;
 - Camada de Aquisição: UART e GPIO
 - Camada de Armazenamento: Sistema de Arquivos e Cartão SD
 - Camada de Apoio: FreeRTOS™ e drivers auxiliares (buffers circulares, serializador de dados, etc.)

Arquitetura de firmware e aplicação



Módulo de Wi-Fi e Gerenciamento de Rede

Gerencia a conexão de rede e fornece portal captivo para configuração e controle do hardware.

- Conecta automaticamente utilizando as credenciais do armazenamento não volátil;
- Gerencia portais captivos para configuração do dispositivo.
- Permite reconfigurar aspectos do firmware sem recompilação.

Módulo de Wi-Fi e Gerenciamento de Rede

Portal de configuração de *hardware*

☒ Habilitar UART 2

Baudrate

115200

Terminador

LF ()

TX Pin

3

RX Pin

4

GPIO CONFIGURATION

☒ BOARD_1

Mode

INPUT

Interrupt

ANYEDGE

Pull Up

ENABLE

Pull Down

DISABLE

Portal de configuração de rede

Wi-Fi Setup

☐ Modo Offline (Sem Wi-Fi)

Rede Wi-Fi (SSID)

JURUBEBA_2G (-58dBm)

Senha

Digite a senha da rede

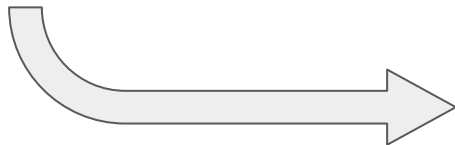
Salvar e Reiniciar

Módulo de MQTT

Gerencia a comunicação assíncrona online, tanto para o monitoramento de telemetrias quanto para comandos remotos.

- O uso explorou os aspectos hierárquicos do protocolo;
- É fácil de separar de onde os dados estão vindo;
- Protocolo leve, baixa latência (importante quando o sistema está em estresse);

Estrutura de tópicos para monitoramento das telemetrias.



```
"datalogger/datalogger_ID/periph/periph_num"  
  
/* -----*/  
  
datalogger/1/uart/2  
datalogger/2/gpio/33  
  
datalogger/3/uart/+
```

Módulo de Registro em SD

O módulo é responsável pelo armazenamento sequencial e confiável dos logs em arquivos binários, gerenciados por um sistema de arquivos.

- Estruturas com atributo de compactação (PACKED) para uso eficiente de memória;
- Cabeçalho com identificação, tipo e timestamp;

Estrutura de mensagem salva em SD

```
typedef struct PACKED sd_log_msg_s
{
    sd_log_hdr_t log_header;
    union
    {
        sd_log_uart_t uart;
        sd_log_gpio_t gpio;
    };
} sd_log_msg_t;
```

Estrutura de cabeçalho

```
#define PACKED __attribute__((packed))

typedef struct PACKED sd_log_hdr_s
{
    uint32_t header;
    uint64_t time_us;
    uint8_t log_type;
    uint8_t periph_num;
} sd_log_hdr_t;

typedef struct PACKED sd_log_uart_s
{
    uint16_t payload_len;
    uint8_t payload[UART_MAX_PAYLOAD_LEN];
} sd_log_uart_t;

typedef struct PACKED sd_log_gpio_s
{
    uint8_t edge;
    uint8_t level;
} sd_log_gpio_t;
```

Módulo de Comunicação UART

Gerencia a recepção serial com buffers circulares, filas e empacotamento de dados.

- Recepção assíncrona com buffers circulares individuais por UART;
- Delimitação de pacotes por caracteres de finalização customizado;

```
while (1)
{
    if(xQueueReceive(uart_queue[uart_num], &event, pdMS_TO_TICKS(10)))
    {
        switch (event.type)
        {
            case UART_DATA:
                int len = uart_read_bytes(uart_num, data, event.size, pdMS_TO_TICKS(100));
                if(len > 0)
                {
                    utl_cbf_put_n(uart_circ_buffers[uart_num], data, len, &written);
                }
            /* ----- CONTINUAÇÃO DA LÓGICA / TRATAMENTO DE OUTROS CASOS ----- */
        }
    }
}
```

Módulo de Comunicação UART



```
while (utl_cbf_bytes_available(uart_circ_buffers[uart_num]))
{
    utl_cbf_get(uart_circ_buffers[uart_num], &uart_byte);

    if (uart_byte == uart_terminator_char[uart_num])
    {
        ESP_LOGI(TAG, "Terminador da UART%d: %d encontrado!", uart_num, uart_char[uart_num]);
        process_uart_packet(uart_num, ctx);
    }

    else
    {
        if (ctx->uart.payload_len < UART_MAX_PAYLOAD_LEN)
        {
            ctx->uart.payload[ctx->uart.payload_len++] = uart_byte;
        }
        else
        {
            ESP_LOGE(TAG, "Buffer Overflow UART%d. Resetando.", uart_num);
            ctx->uart.payload_len = 0;
        }
    }
}
```

Módulo de GPIO

Gerencia pinos digitais com interrupções rápidas e tratamento desacoplado.

- Interrupções de baixa latência;
- Debouce por tempo para evitar ruídos;
- Eventos enfileirados para processamento;

Handler da ISR de GPIO

```
static void IRAM_ATTR gpio_isr_handler(void *args)
{
    uint32_t gpio_idx = (uint32_t)args;

    int64_t now = esp_timer_get_time();

    if ((now - last_evt_time[gpio_idx]) < GPIO_DEBOUNCE_US)
    {
        return;
    }

    last_evt_time[gpio_idx] = now;

    gpio_evt_t evt =
    {
        .gpio = (uint8_t)gpio_idx,
        .level = gpio_get_level(gpio_num),
        .time_us = now,
    };

    xQueueSendFromISR(gpio_evt_queue, &evt, &xHigherPriorityTaskWoken);
}
```

Tarefa de publicação dos dados de GPIO

```
if (xQueueReceive(gpio_evt_queue, &evt, portMAX_DELAY))
{
    uint8_t gpio_logic = evt.gpio; // índice lógico do gpio

    int real_pin = gpio_pins[gpio_logic];

    ESP_LOGI(TAG, "GPIO%d | level=%d | t=%lld us", real_pin, evt.level, evt.time_us);

    sd_log_msg_t gpio_msg = {0};

    gpio_msg.log_header.header = LOG_PACKET_HEADER_INIT;
    gpio_msg.log_header.time_us = evt.time_us;
    gpio_msg.log_header.log_type = SD_LOG_GPIO;
    gpio_msg.log_header.periph_num = real_pin;

    gpio_msg.gpio.edge = evt.level;
    gpio_msg.gpio.level = evt.level;

    sd_log_data(&gpio_msg);
    mqtt_publish_msg(&gpio_msg);
}
```

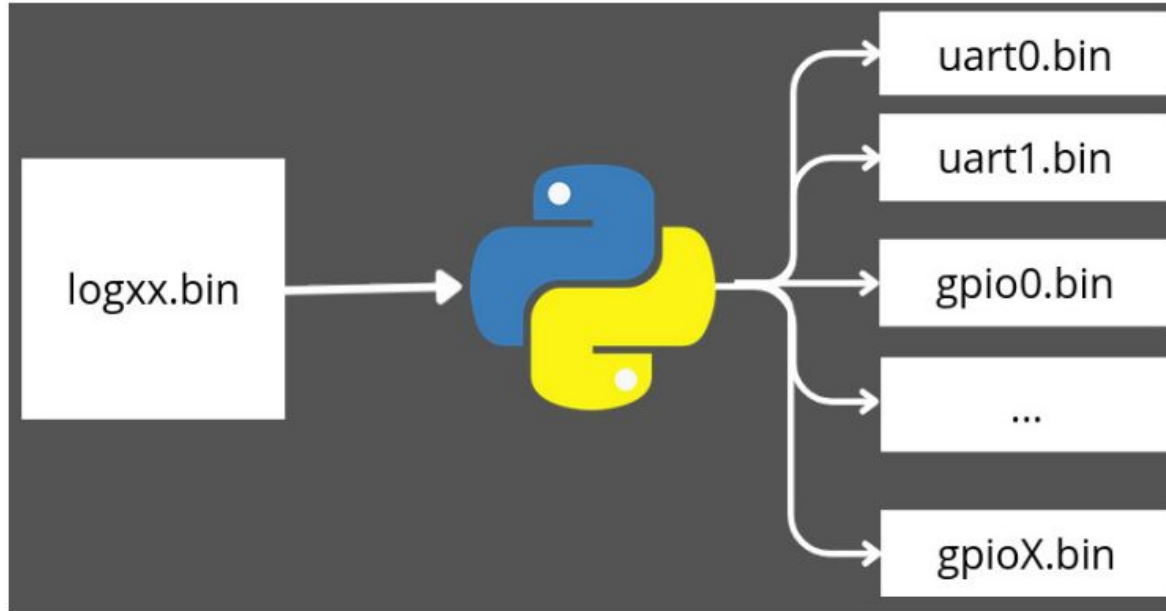
Ferramentas do sistema

O sistema possui scripts auxiliares para servir de interface com o usuário:

- Ferramenta para decodificação dos logs do cartão SD (`log_processor.py`);
- Comandos remotos; ativar/desativar pino de GPIO, enviar mensagem pela UART (`mqtt_remote_cmd.py`);
- Monitorar a telemetria em tempo real por mqtt (`mqtt_monitor.py`);

Ferramentas do sistema

log_processor.py



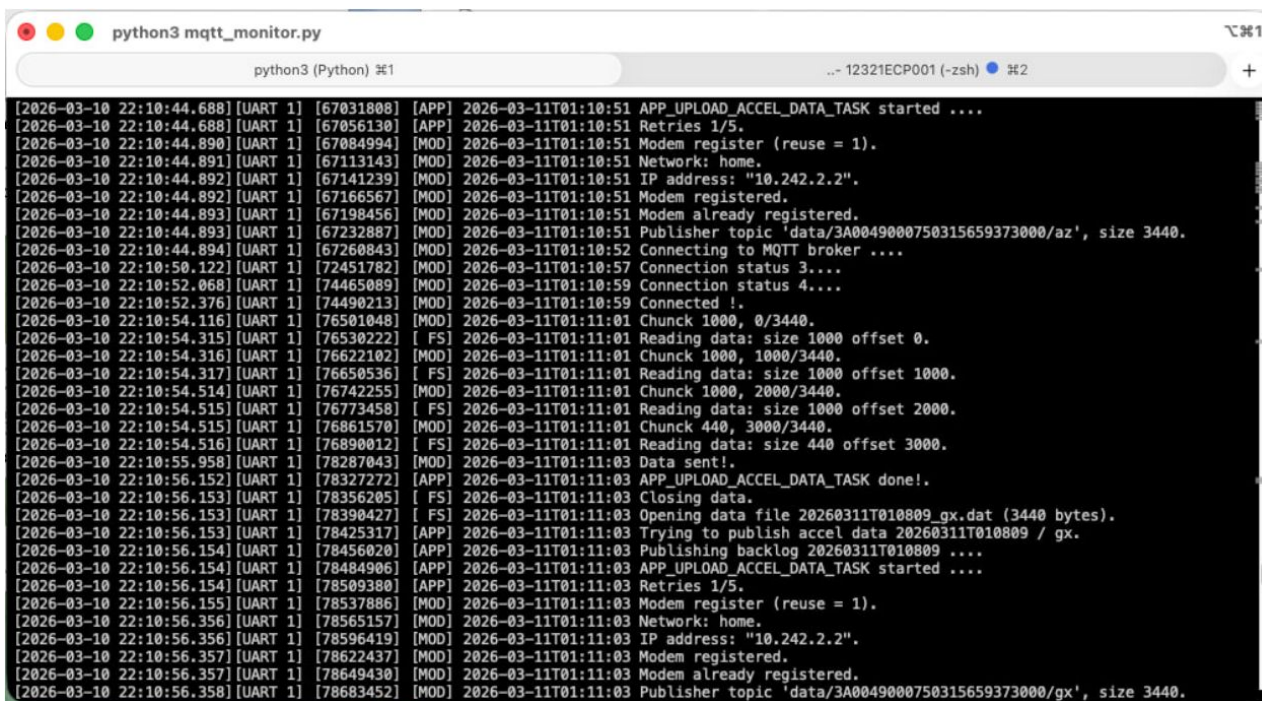
Ferramentas do sistema

mqtt_remote_cmd.py

```
(venv) (t) ~/tcc/software/mqtt/ [main*] python3 mqtt_remote_cmd.py uart 1 "Hello World"
/home/pedro-bittencourt/tcc/software/mqtt/mqtt_remote_cmd.py:108: DeprecationWarning:
  Callback API version 1 is deprecated, update to latest version
  client = mqtt.Client(client_id=client_id, protocol=mqtt.MQTTv311)
Configurando contexto seguro (SSL/TLS)...
Certificado CA carregado de: ../mqtt/emqxsl-ca.crt
Conectando...
Conectado com sucesso a qa717179.ala.us-east-1.emqxsl.com!
[UART] Enviando para 'datalogger/cmd/uart': {"uart_num": 1, "msg": "Hello World"}
-> Comando entregue ao Broker.
(venv) (t) ~/tcc/software/mqtt/ [main*] □
```

Ferramentas do sistema

mqtt_monitor.py



```
python3 mqtt_monitor.py

python3 (Python) #1

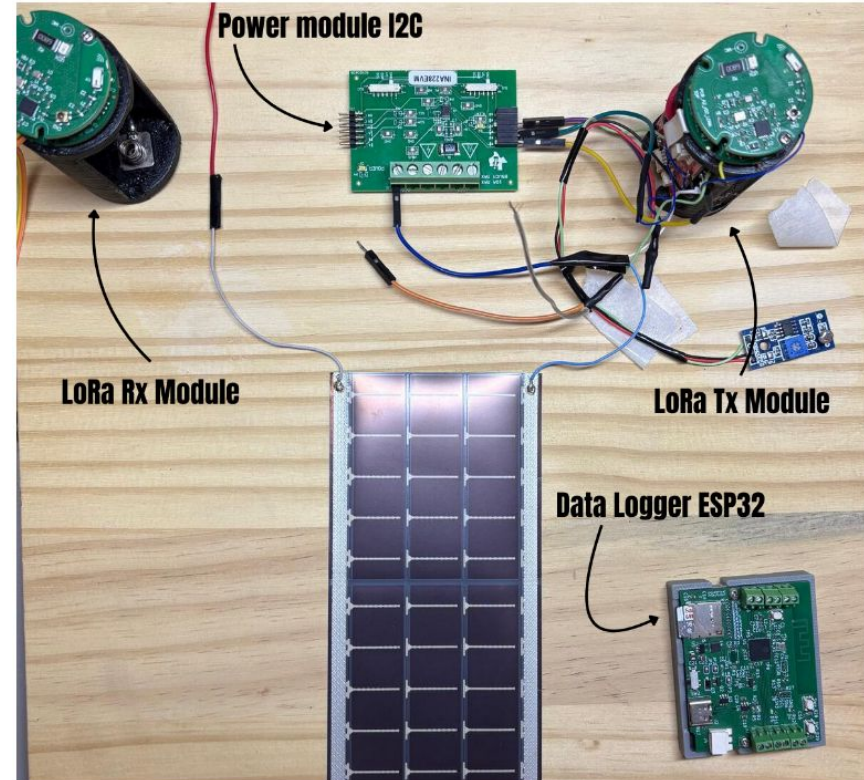
... 12321ECP001 (-zsh) #2

[2026-03-10 22:10:44.688] [UART 1] [67031808] [APP] 2026-03-11T01:10:51 APP_UPLOAD_ACCEL_DATA_TASK started ....
[2026-03-10 22:10:44.688] [UART 1] [67056130] [APP] 2026-03-11T01:10:51 Retries 1/5.
[2026-03-10 22:10:44.890] [UART 1] [67084994] [MOD] 2026-03-11T01:10:51 Modem register (reuse = 1).
[2026-03-10 22:10:44.891] [UART 1] [67113143] [MOD] 2026-03-11T01:10:51 Network: home.
[2026-03-10 22:10:44.892] [UART 1] [67141239] [MOD] 2026-03-11T01:10:51 IP address: "10.242.2.2".
[2026-03-10 22:10:44.892] [UART 1] [67166567] [MOD] 2026-03-11T01:10:51 Modem registered.
[2026-03-10 22:10:44.893] [UART 1] [67198456] [MOD] 2026-03-11T01:10:51 Modem already registered.
[2026-03-10 22:10:44.893] [UART 1] [67232887] [MOD] 2026-03-11T01:10:51 Publisher topic 'data/3A0049000750315659373000/az', size 3440.
[2026-03-10 22:10:44.894] [UART 1] [67260843] [MOD] 2026-03-11T01:10:52 Connecting to MQTT broker ....
[2026-03-10 22:10:50.122] [UART 1] [72451782] [MOD] 2026-03-11T01:10:57 Connection status 3....
[2026-03-10 22:10:52.068] [UART 1] [74465089] [MOD] 2026-03-11T01:10:59 Connection status 4....
[2026-03-10 22:10:52.376] [UART 1] [74490213] [MOD] 2026-03-11T01:10:59 Connected !.
[2026-03-10 22:10:54.116] [UART 1] [76501048] [MOD] 2026-03-11T01:11:01 Chunk 1000, 0/3440.
[2026-03-10 22:10:54.315] [UART 1] [76530222] [ FS] 2026-03-11T01:11:01 Reading data: size 1000 offset 0.
[2026-03-10 22:10:54.316] [UART 1] [76622102] [MOD] 2026-03-11T01:11:01 Chunk 1000, 1000/3440.
[2026-03-10 22:10:54.317] [UART 1] [76650536] [ FS] 2026-03-11T01:11:01 Reading data: size 1000 offset 1000.
[2026-03-10 22:10:54.514] [UART 1] [76742255] [MOD] 2026-03-11T01:11:01 Chunk 1000, 2000/3440.
[2026-03-10 22:10:54.515] [UART 1] [76773458] [ FS] 2026-03-11T01:11:01 Reading data: size 1000 offset 2000.
[2026-03-10 22:10:54.515] [UART 1] [76861570] [MOD] 2026-03-11T01:11:01 Chunk 440, 3000/3440.
[2026-03-10 22:10:54.516] [UART 1] [76890012] [ FS] 2026-03-11T01:11:01 Reading data: size 440 offset 3000.
[2026-03-10 22:10:55.958] [UART 1] [78287043] [MOD] 2026-03-11T01:11:03 Data sent!.
[2026-03-10 22:10:56.152] [UART 1] [78327272] [APP] 2026-03-11T01:11:03 APP_UPLOAD_ACCEL_DATA_TASK done!.
[2026-03-10 22:10:56.153] [UART 1] [78356205] [ FS] 2026-03-11T01:11:03 Closing data.
[2026-03-10 22:10:56.153] [UART 1] [78390427] [ FS] 2026-03-11T01:11:03 Opening data file 20260311T010809_gx.dat (3440 bytes).
[2026-03-10 22:10:56.153] [UART 1] [78425317] [APP] 2026-03-11T01:11:03 Trying to publish accel data 20260311T010809 / gx.
[2026-03-10 22:10:56.154] [UART 1] [78456020] [APP] 2026-03-11T01:11:03 Publishing backlog 20260311T010809 ....
[2026-03-10 22:10:56.154] [UART 1] [78484906] [APP] 2026-03-11T01:11:03 APP_UPLOAD_ACCEL_DATA_TASK started ....
[2026-03-10 22:10:56.154] [UART 1] [78509380] [APP] 2026-03-11T01:11:03 Retries 1/5.
[2026-03-10 22:10:56.155] [UART 1] [78537886] [MOD] 2026-03-11T01:11:03 Modem register (reuse = 1).
[2026-03-10 22:10:56.356] [UART 1] [78565157] [MOD] 2026-03-11T01:11:03 Network: home.
[2026-03-10 22:10:56.356] [UART 1] [78596419] [MOD] 2026-03-11T01:11:03 IP address: "10.242.2.2".
[2026-03-10 22:10:56.357] [UART 1] [78622437] [MOD] 2026-03-11T01:11:03 Modem registered.
[2026-03-10 22:10:56.357] [UART 1] [78649430] [MOD] 2026-03-11T01:11:03 Modem already registered.
[2026-03-10 22:10:56.358] [UART 1] [78683452] [MOD] 2026-03-11T01:11:03 Publisher topic 'data/3A0049000750315659373000/gx', size 3440.
```

Caso de Teste 1: Monitoramento Fotovoltaico com Telemetria LoRa

Objetivo: Validar a operação de um sistema por longas horas em ambiente remoto.

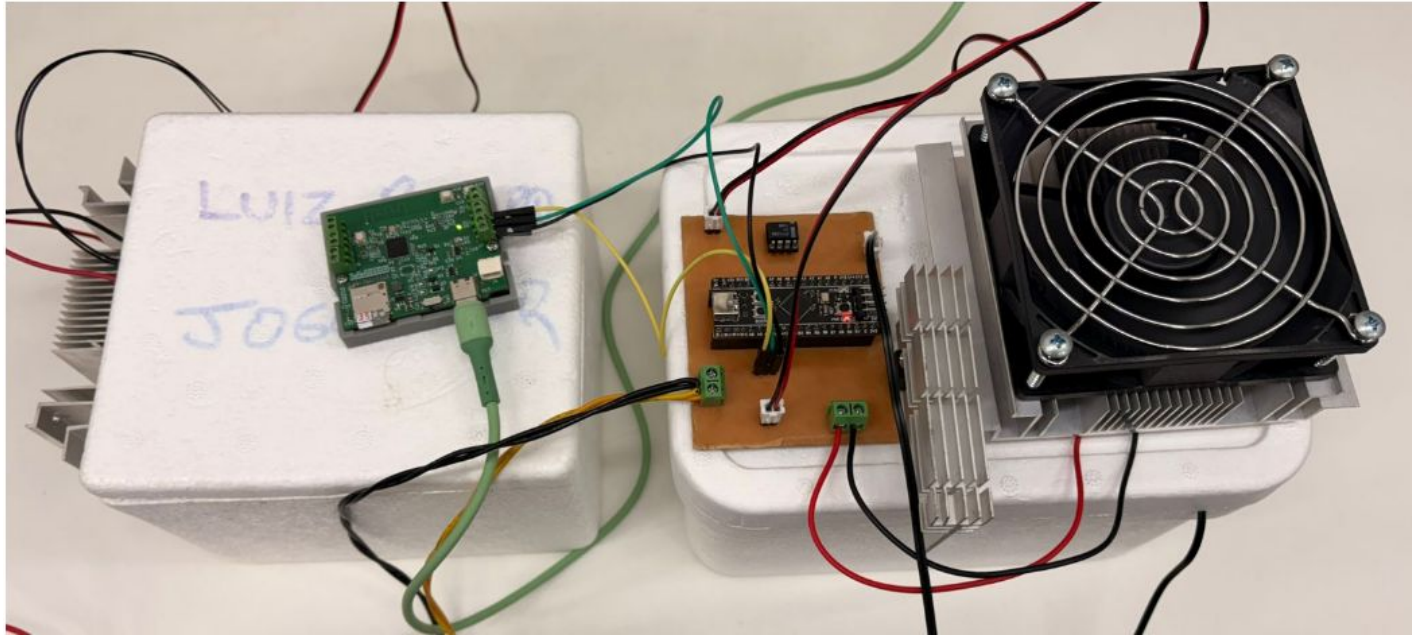
Funcionamento: Recebe dados via LoRa (UART) e registra em SD + MQTT



Caso de Teste 1: Controle e Interação Remota com Câmara Fria

Objetivo: Testar comunicação bidirecional em sistema isolado

Funcionamento: Gateway UART com envio e recepção via MQTT



OBRIGADO!



mouser.com



www.embarcados.com.br



linkedin.com/embarcados



[@portalembarcados](https://www.instagram.com/portalembarcados)



[youtube/Embarcados TV](https://youtube.com/EmbarcadosTV)