**Introdução:** O [bridge\_ESP32](https://github.com/RoboCore/RoboCore_SMW-SX1276M0/blob/master/examples/Bridge_ESP32/Bridge_ESP32.ino) foi feito de maneira simples para conectar o computador ao módulo LoRaWAN, utilizando o ESP32 para se comunicar com o módulo LoRaWAN.

**Objetivo:** Através do programa obter informação de configuração do módulo como a região que ele está configurado (Precisa ser a região 1 - AU915), configurar o módulo caso necessário via comandos AT.

CÓDIGO COMENTADO:

#if !defined(ARDUINO\_ESP32\_DEV) // Verifica se a placa é ESP32

#error Use este exemplo com o ESP32

#endif

// --------------------------------------------------

// Bibliotecas

#include "RoboCore\_SMW\_SX1276M0.h" // Inclui a biblioteca específica para o módulo LoRaWAN

// --------------------------------------------------

// Variáveis

#include <HardwareSerial.h> // Inclui a biblioteca para a comunicação serial com hardware

HardwareSerial LoRaSerial(2); // Define o segundo UART (Serial2) do ESP32

#define RXD2 16 // Define o pino de recepção RX para o UART2

#define TXD2 17 // Define o pino de transmissão TX para o UART2

SMW\_SX1276M0 lorawan(LoRaSerial); // Instancia o objeto para comunicação com o módulo LoRaWAN

// --------------------------------------------------

// Função de inicialização

void setup() {

// Inicializa a comunicação UART para o computador

Serial.begin(115200); // Configura a taxa de transmissão em 115200 bps

Serial.println(F("--- SMW\_SX1276M0 Bridge ---")); // Imprime uma mensagem inicial na UART do computador

// Definição do pino de reset do módulo LoRaWAN

lorawan.setPinReset(5); // Configura o pino 5 como o pino de reset do módulo

lorawan.reset(); // Realiza um reset no módulo LoRaWAN

// Inicializa a comunicação UART para o módulo LoRaWAN

LoRaSerial.begin(115200, SERIAL\_8N1, RXD2, TXD2);

// Configura o UART2 com taxa de transmissão de 115200 bps, 8 bits de dados, sem paridade, 1 bit de parada

}

// --------------------------------------------------

// Função principal

void loop() {

// Comunicação do módulo SMW\_SX1276M0 para o computador

if (LoRaSerial.available()) { // Verifica se há dados disponíveis no UART do módulo LoRaWAN

Serial.write(LoRaSerial.read()); // Lê o dado recebido do LoRaWAN e envia para o computador

}

// Comunicação do computador para o módulo SMW\_SX1276M0

if (Serial.available()) { // Verifica se há dados disponíveis na UART do computador

LoRaSerial.write(Serial.read()); // Lê o dado recebido do computador e envia para o LoRaWAN

}

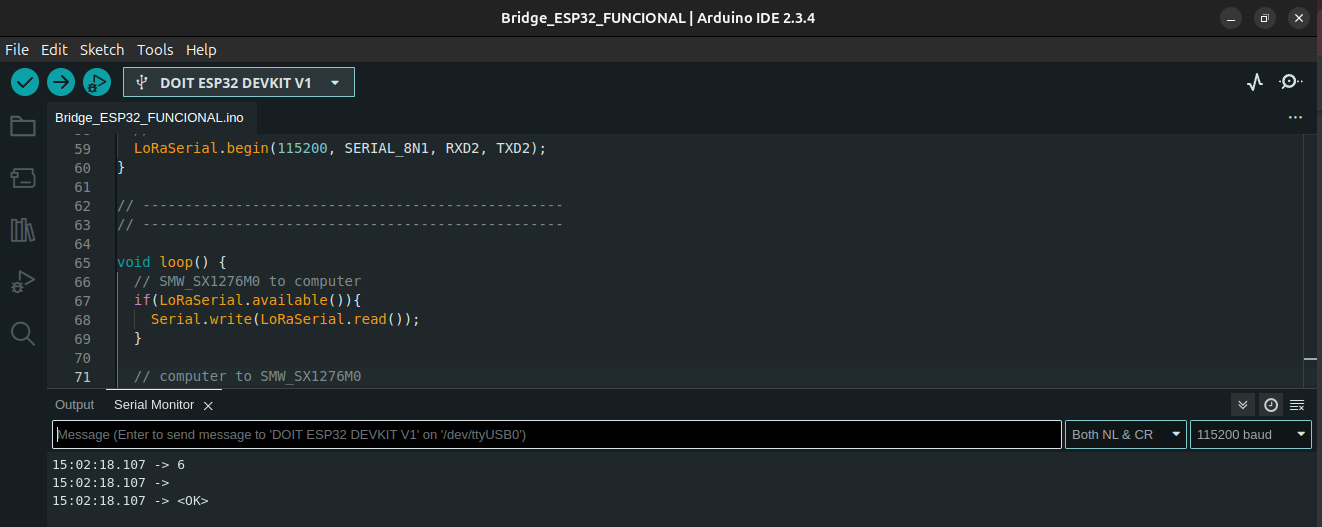
}

// Fim do código

VERIFICANDO REGIÃO

A região é um parâmetro muito importante pois define a faixa de frequência da comunicação do módulo que por Lei no Brasil utiliza-se 915 Mhz (que pelo manual de [comandos do Fabricante](https://d229kd5ey79jzj.cloudfront.net/1239/SMART_LoRa_AT.Command_v1.1_en_v0.5.pdf) [pag23] é REGION\_AU915 = 1).

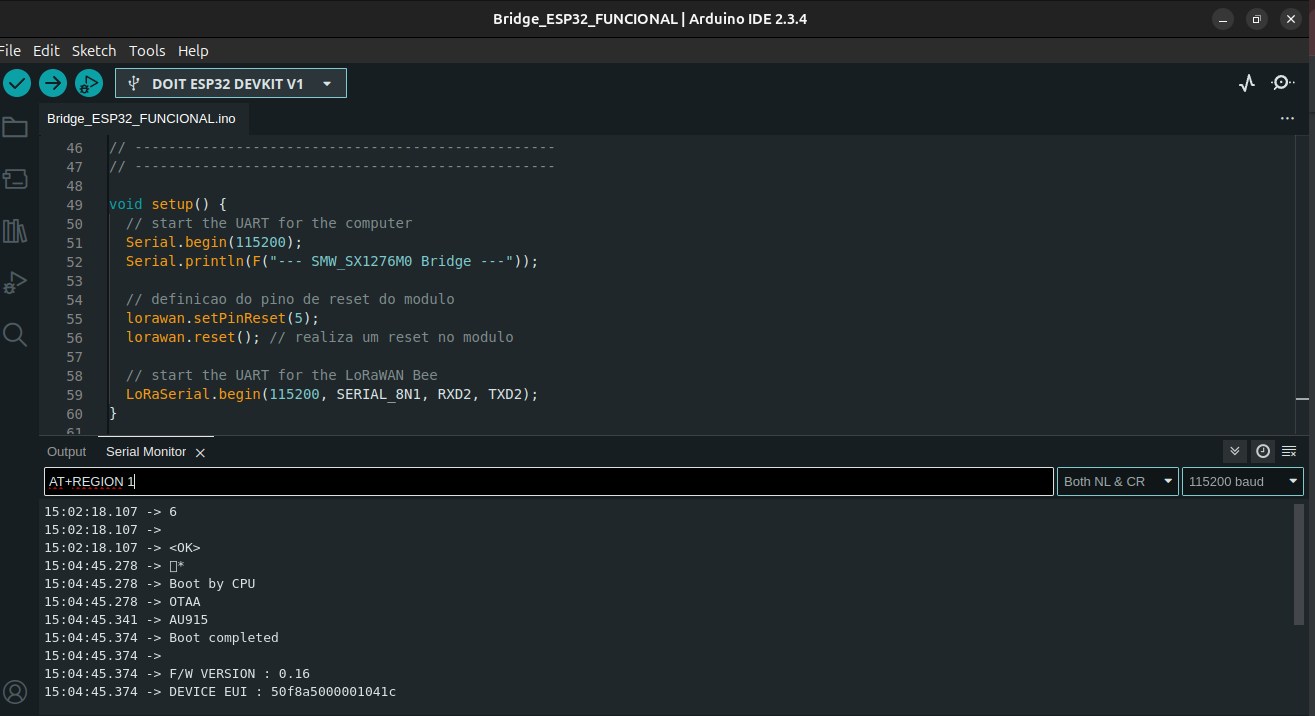
figura 1



Uma das placas estava configurada na região 6 (Quando usado o comando AT+REGION).

Logo, foi possível com o comando AT+REGION 1 configurar a nossa placa para a região 1, como pode ser visto na figura 2.

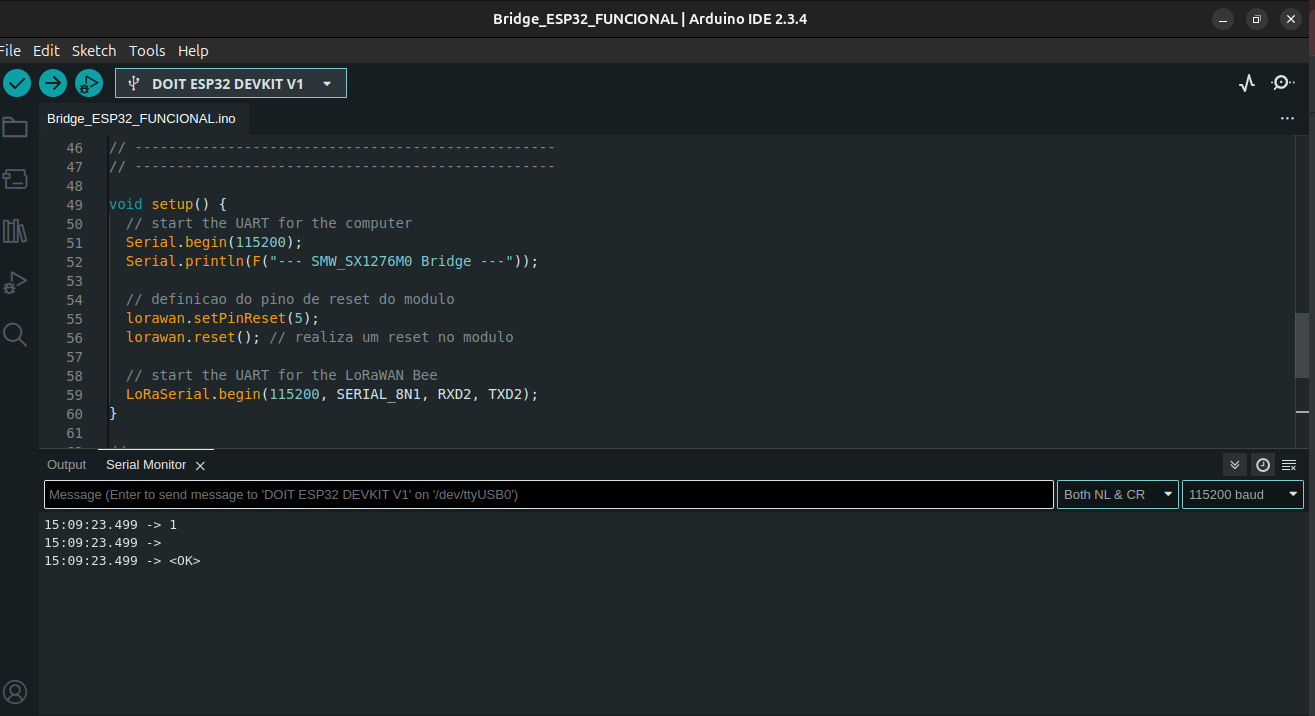
figura 2



Uma das placas passou a estar configurada na região 1 (Quando usado o comando AT+REGION 1).

E então temos a região devidamente configurada como mostra a figura 3.

figura 3



Configurada na região 1 (Quando usado o comando AT+REGION).

Outra forma de configurar a regiao da placa é atravez da função fornecida pela biblioteca <[set\_Region()](https://github.com/RoboCore/RoboCore_SMW-SX1276M0/blob/10382101a6bb926053f6d6d042d007dce4128fa0/src/RoboCore_SMW_SX1276M0.h#L217)> para fazer a definição da região:

// Obtém a região

uint8\_t regiao = 1; // Define a região como 1 (por exemplo, América do Norte, Europa, etc., dependendo da documentação do módulo)

// Configura a região no módulo LoRaWAN usando o método set\_Region().

// Esse método aceita um valor inteiro que representa a região.

response = lorawan.set\_Region(regiao);

// Verifica se o comando foi bem-sucedido

if (response == CommandResponse::OK) { // Se o comando foi executado com sucesso

Serial.print(F("Regiao: ")); // Exibe a mensagem "Regiao:" na interface serial

Serial.println(regiao); // Exibe o valor da região configurada

} else {

// Se houve um erro na configuração da região, exibe uma mensagem de erro

Serial.println(F("Erro ao obter a regiao"));

}

// Após executar este código, a placa já estará configurada para a região 1.

// Isso pode ser testado usando o código "Bridge\_ESP32" e enviando o comando AT+REGION.

// O comando AT+REGION deve retornar a região configurada no módulo.

// ATENÇÃO: Certifique-se de que a região configurada é compatível com as regulamentações locais

// de frequência para evitar problemas legais ou interferências. Consulte a documentação do módulo

// para mais informações sobre os valores disponíveis para regiões e seus respectivos significados.

