

## Hardware

### 1. Introdução

Este documento descreve o projeto de telemetria para o veículo da equipe **FSAE-Unicamp**, com foco na coleta, transmissão e análise de dados em tempo real. O sistema utiliza tecnologias IoT (Internet das Coisas) e comunicação sem fio LoRa para garantir robustez e alcance adequado em ambientes dinâmicos.

### 2. Componentes do Hardware

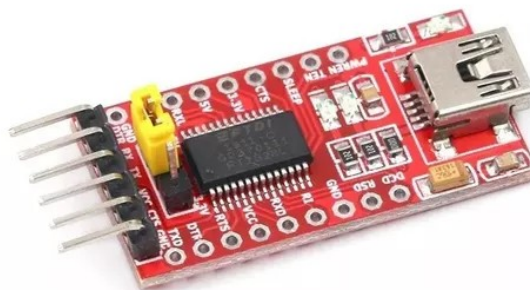
1. [2x CDEBYTE E22-900M30S:](#)



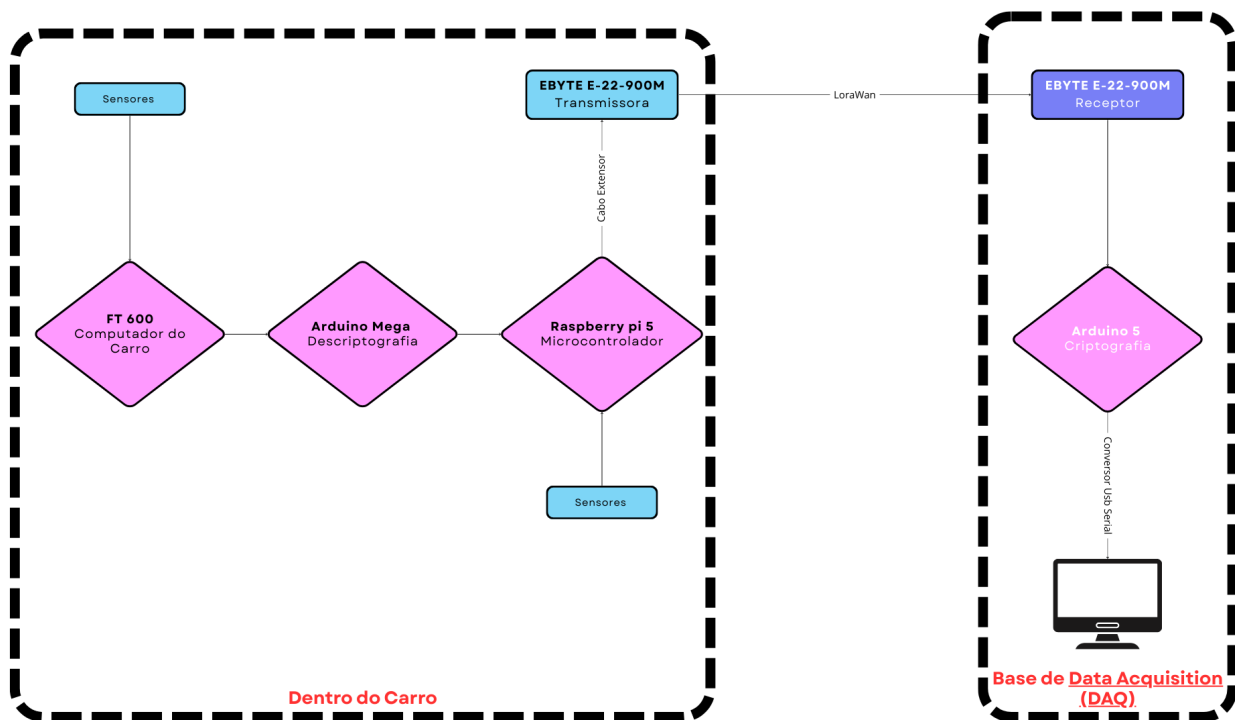
2. [2x Antena Lora:](#)



3. [Conversor USB Serial:](#)



## Diagrama em blocos:



## Função de cada bloco:

### 1. Sensores

#### Conexão:

- Sensores convencionais: conectados diretamente ao FT 650 (computador de bordo).
- Sensores da rede CAM: transmitem dados criptografados ao Arduino Mega.

#### Função:

- Coletam dados críticos (ex: RPM, temperatura, pressão de óleo, aceleração).
- Sensores da rede CAM operam em protocolo seguro, enviando dados criptografados para descriptografia.

### 2. FT 600 (Computador de Bordo)

#### Conexão:

- Recebe dados de sensores convencionais.
- Encaminha dados criptografados da rede CAM ao Arduino Mega.
- Envia dados processados (sensores convencionais + alertas) à Raspberry Pi 5.

#### Função:

- Processamento primário de dados brutos (ex: cálculos de eficiência térmica, ajustes de injeção de combustível).
- Prioriza alertas críticos (ex: superaquecimento, falha elétrica) para envio imediato à Raspberry Pi.

### 3. Arduino Mega (Descriptografia)

#### Conexão:

- Recebe dados criptografados da rede CAM via FT 600.
- Envia dados descriptografados à Raspberry Pi 5.

#### Função:

- Descriptografa dados da rede CAM usando protocolos específicos para garantir segurança.
- Valida a integridade dos dados antes de repassá-los à Raspberry Pi.

### 4. Raspberry Pi 5 (Microcontrolador Central)

#### Conexão:

- Recebe dados processados do FT 600 e dados descriptografados do Arduino Mega.
- Conecta-se ao Cartão de Memória e à Antena Transmissora.

Função:

- Backup: Armazena dados estruturados no cartão de memória, incluindo dados descriptografados da rede CAM.
- Transmissão: Envia pacotes compactados e criptografados via Antena Transmissora (LoRaWAN para longo alcance).
- Pré-processamento: Filtra ruídos, aplica algoritmos de suavização (ex: média móvel) e combina dados de múltiplas fontes (FT 600 + Arduino).

5. Cartão de Memória

Função:

- Armazena backups locais com redundância, garantindo recuperação de dados em caso de falha na transmissão ou perda de conexão.

6. Antena Transmissora

Função:

- Transmite dados em tempo real para a equipe remota.

Protocolos:

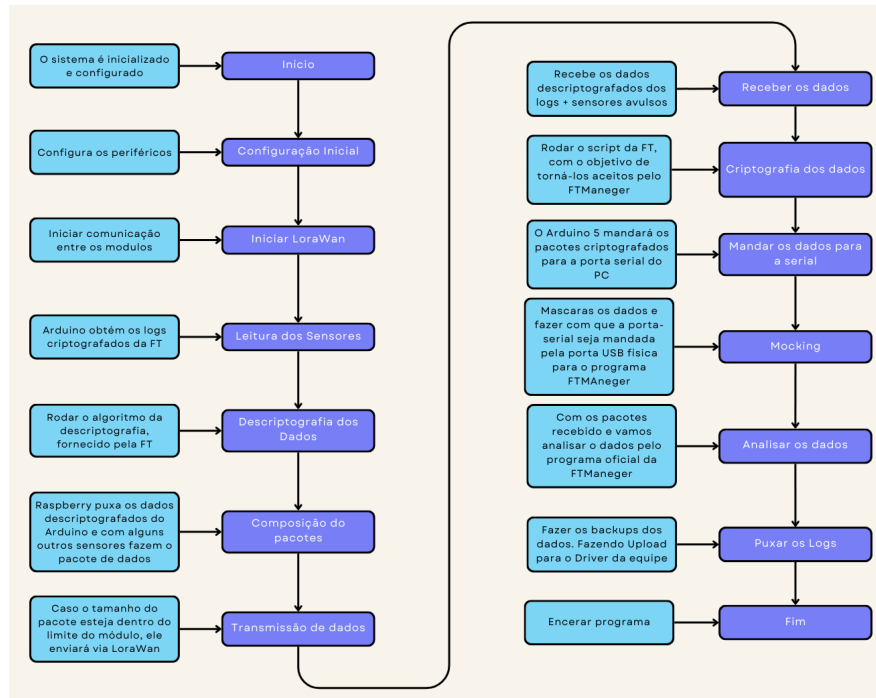
- **LoRaWAN**: Priorizado para comunicação de longo alcance e baixo consumo energético.

7. Antena Receptora + Computador

Função:

- Recebe dados via Antena Receptora.
- Gera dashboards em tempo real (ex: Power BI) para monitoramento remoto.
- Armazena dados em nuvem para análise histórica e machine learning.

## Fluxograma:



## Outra opção seria:

### 1. INICIALIZAÇÃO DO SISTEMA

- | — Configurar Periféricos
  - | | — Iniciar Raspberry Pi 5 (Controlador Principal)
  - | | — Conectar Arduino Mega (Descritografia) e Arduino S (Criptografia)
  - | | — Inicializar EBYTE E22-900M (Transmissor/Receptor LoRaWAN)
- | — Estabelecer Comunicação
  - | | — Conectar FT600 (Computador do Carro) via Cabo Estesque
  - | | — Iniciar DAQ (Base de Aquisição de Dados)
- | — Configurar Sensores
  - | | — Calibrar sensores (ABE, etc.)
  - | | — Definir parâmetros de coleta (taxa de amostragem, formato)

### 2. COLETA DE DADOS

- | — Arduino Mega lê logs criptografados da FT600
- | — Executar Algoritmo de Descritografia (fornecido pela FT)
- | — Combinar dados com leituras de sensores avulsos
- | — Validar integridade dos dados (checksum)

### 3. PREPARAÇÃO DE PACOTES

- |— Verificar tamanho do pacote
- | |— Se  $\leq 242$  bytes (limite LoRaWAN) → Prosseguir
- | |— Se exceder → Fragmentar ou compactar
- |
- |— Criptografar dados no Arduino S (AES-128)
- |— Formatar pacote (cabeçalho + payload + CRC)

#### 4. TRANSMISSÃO LoRaWAN

- |— Enviar pacote via EBYTE E22-900M (915 MHz)
- |— Aguardar ACK (confirmação de recebimento)
- | |— Recebido → Registrar sucesso
- | |— Falha → Retransmitir (máx. 3 tentativas)
- |— Atualizar logs de transmissão

#### 5. RECEPÇÃO E PROCESSAMENTO

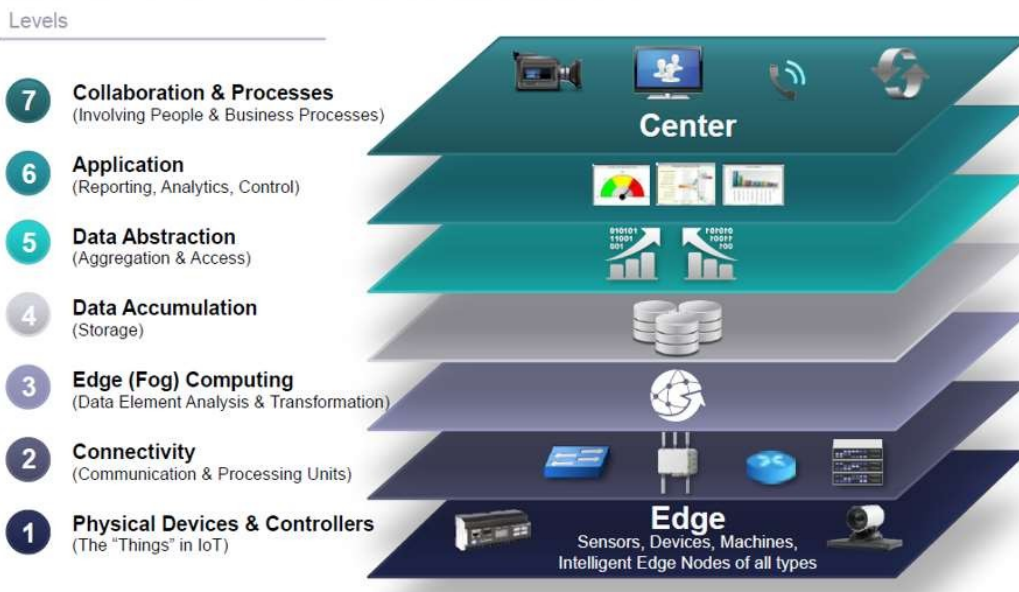
- |— Base DAQ recebe dados via EBYTE E22-900M
- |— Decodificar pacote (Arduino S)
- | |— Verificar CRC
- | |— Descriptografar dados
- |
- |— FTManager processa dados
- | |— Gerar visualizações em tempo real
- | |— Armazenar dados brutos (SQLite)
- | |— Exportar dados processados (CSV/Excel)
- |
- |— Backup automático para nuvem (Google Drive)

#### 6. ENCERRAMENTO

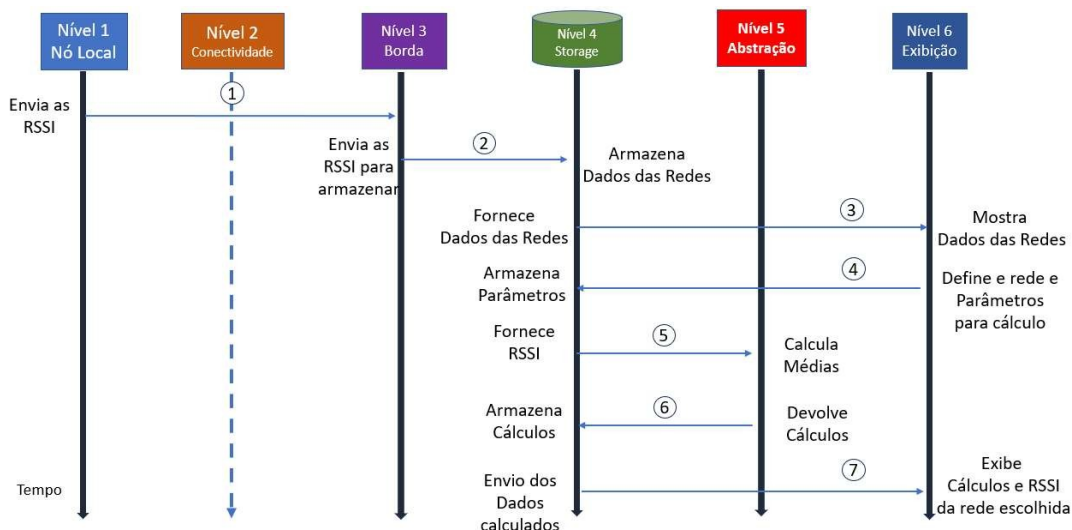
- |— Finalizar conexões seriais
- |— Gerar relatório de operação
- | |— Estatísticas de transmissão
- | |— Alertas de erro (se aplicável)
- |— Entrar em modo low-power (Raspberry Pi)

### 3. Arquitetura IoT por Níveis (Modelo CISCO)

#### Internet of Things Reference Model

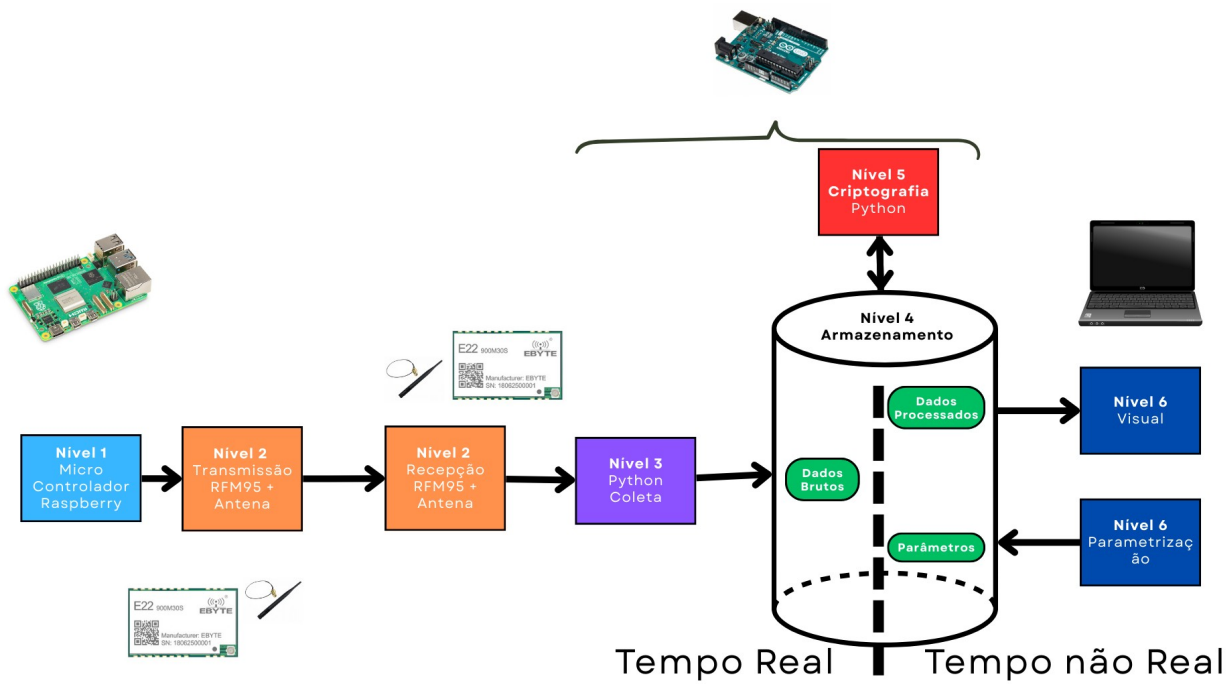


- Nível 1 – trata dos dispositivos físicos e controladores locais
- Nível 2 – trata da conexão dos processos do Nível 1 com o Nível 3.
- Nível 3 – faz a coleta dos dados oriundos de vários sensores.
- Nível 4 – armazenamento dos dados.
- Nível 5 – abstração dos dados.
- Nível 6 – aplicações para analisar.
- Nível 7 – envolvimento das pessoas.



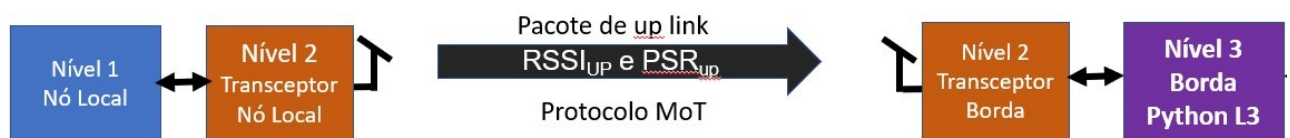
Nível	Função	Componentes/Exemplo
1	Dispositivos Físicos	Sensores, RFM95, Arduino Mega.
2	Conectividade	LoRa, WiFi, Protocolo MoT.
3	Processamento na Borda (Edge)	Raspberry Pi (filtragem de ruído).
4	Armazenamento	Cartão de memória, backup local.
5	Abstração de Dados	Cálculo de médias móveis.
6	Aplicação e Visualização	Dashboards em Power BI.

## Diagrama MACRO:

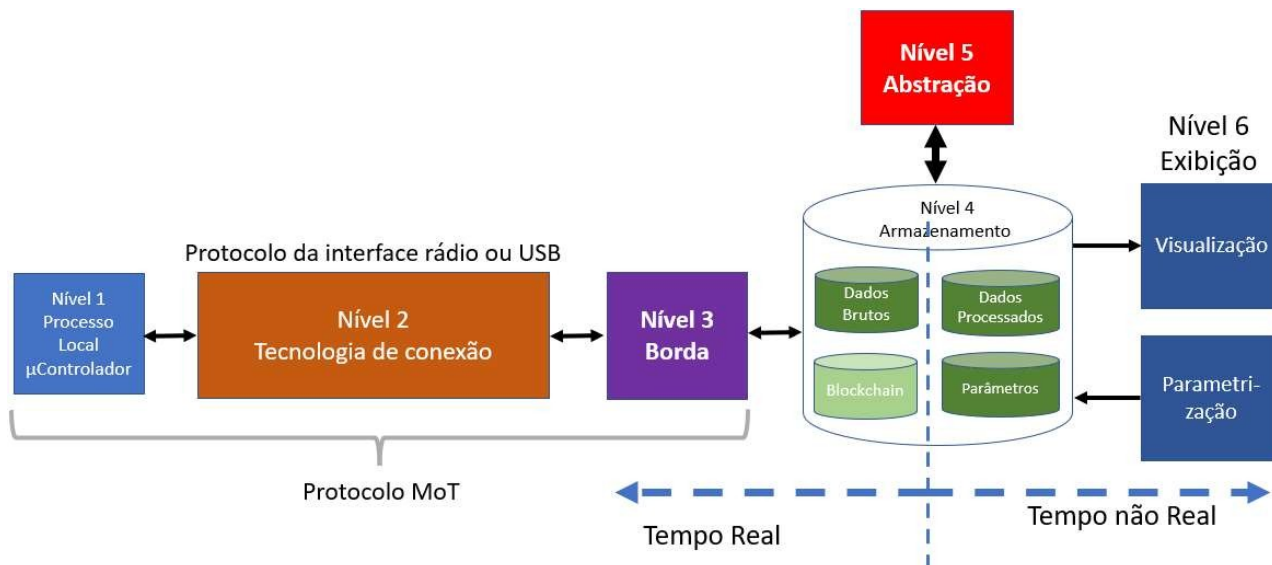


**Observação:** No nível 6, a compatibilização com o software FTManager é fundamental para garantir que os dados coletados e estruturados sejam aceitos no fluxo serial da aplicação proprietária.

## Trajeto de Pacotes entre os níveis 1 e 3:



#### 4. Protocolo de Comunicação: MoT (Management over Tunneling)

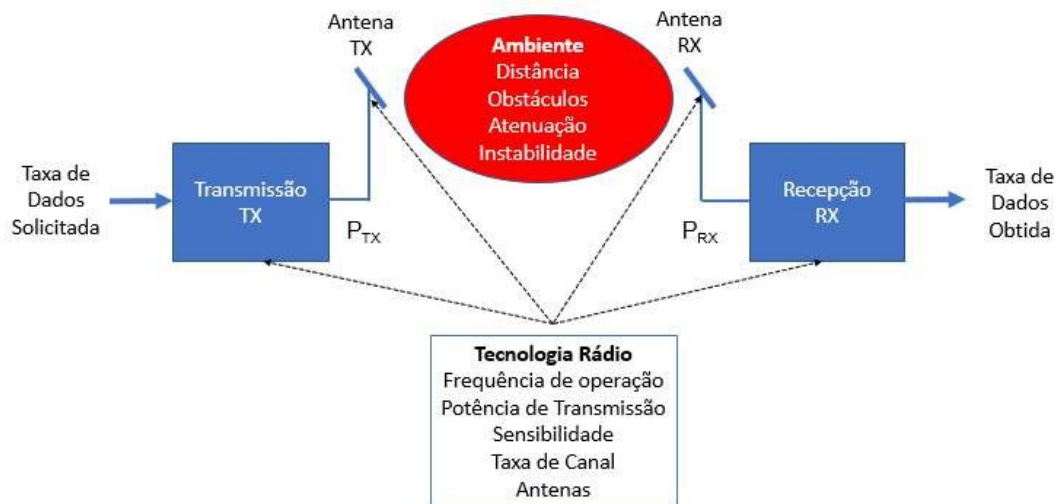


MoT é empregado como protocolo encapsulador de dados sensoriais, oferecendo segurança e gerenciamento de tráfego. Permite autenticação e priorização em nível de pacote, estruturando headers com:

- Identificador único do sensor
- Carimbo temporal (timestamp)
- Tipo e unidade do dado
- Valor bruto ou processado
- Verificador de integridade (checksum)



## 5. Análise de Sinais e Antenas



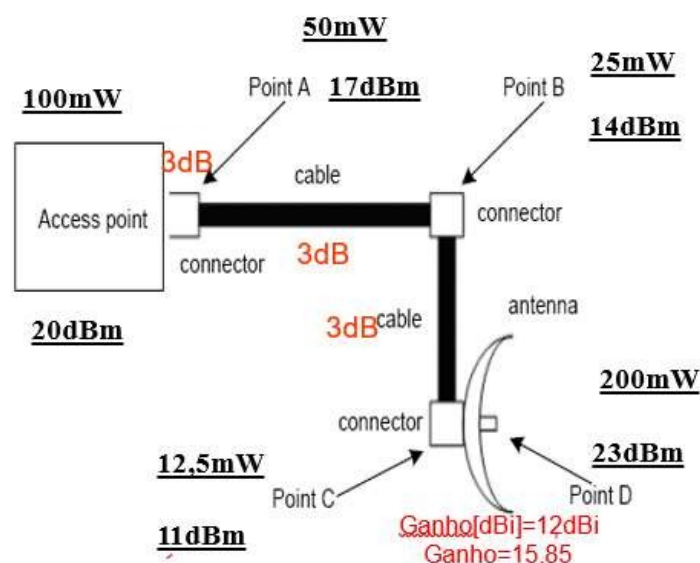
### Mecanismos de Atenuação

A propagação do sinal de rádio é impactada por:

- Obstruções físicas (carroceria, vegetação, estruturas metálicas)
- Efeitos de reflexão e interferência por múltiplos caminhos (multipath)
- Interferências eletromagnéticas no espectro de 915 MHz
- Alterações na geometria do enlace em função da velocidade do veículo

### Parâmetros Radiométricos

- **dBm**: Representa a potência absoluta em relação a 1 mW, em escala logarítmica
- **dBi**: Denota o ganho da antena em relação a uma fonte isotrópica ideal



### dBm vs. dBi:

- **dBm**: Potência absoluta (ex: 20 dBm = 100 mW).

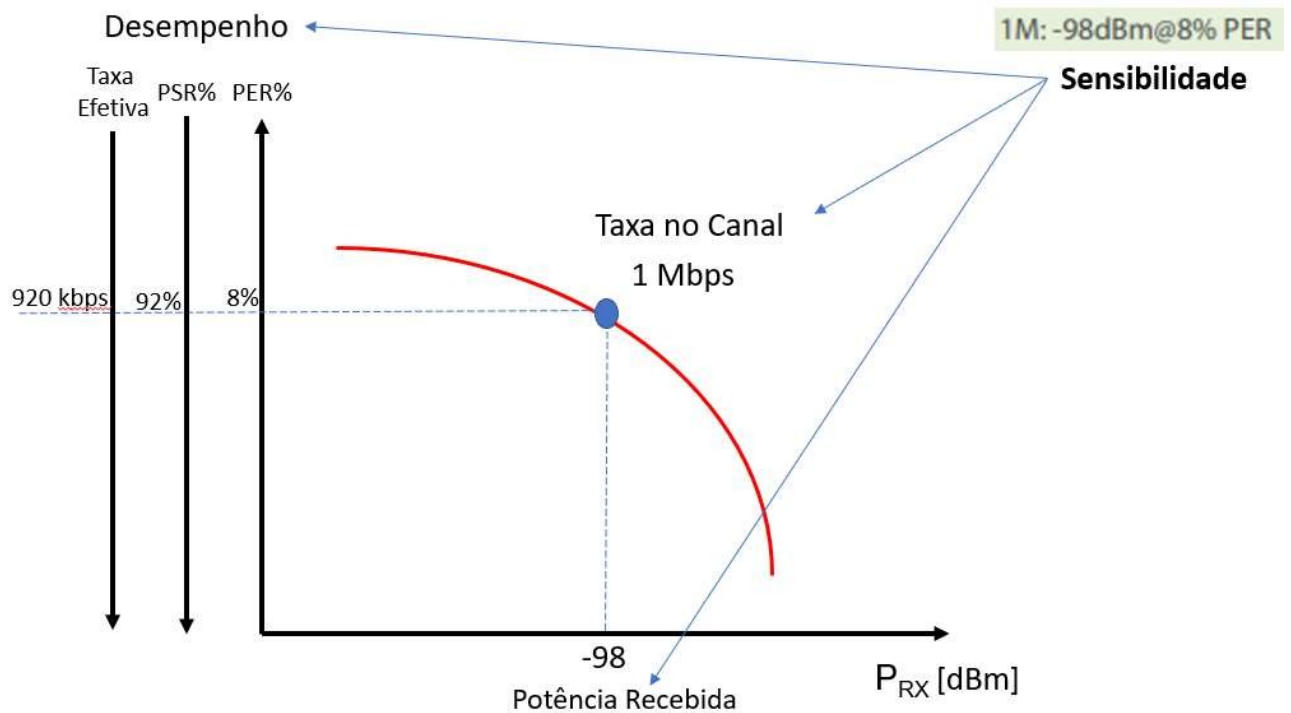
- **dB*i*:** Ganho relativo à antena isotrópica (ex: Laird FG8960: 9 dBi).
- **Cálculo de Potência Recebida (Friis):**

$$PRX = PTX + GTX + GRX - LEL - \text{Perdas (cabos, conectores)}$$

**Exemplo:**

- $PTX = 20 \text{ dBm}$ ,  $GTX = GRX = 9 \text{ dBi}$ ,  $LEL = 99.63 \text{ dB}$  (2 km, 915 MHz).
- $PRX = 20 + 9 + 9 - 99.63 = -61.63 \text{ dBm}$ .

Sensibilidade do receptor:



A sensibilidade é a menor potência detectável com integridade estatística aceitável. Para o RFM95W, baseada no chip Semtech SX1276, observam-se os seguintes valores:

Especificações do RFM95W LoRa, baseado no chip SX1276:

- **Potência de saída:** ajustável entre +5 dBm e +20 dBm, com capacidade de até 100 mW, configurável via software.
- **Consumo de corrente:**
  - Aproximadamente 100 mA durante a transmissão a +20 dBm.
  - Cerca de 30 mA durante a recepção ativa.
- **Alcance:** aproximadamente 2 km em linha de visada com antenas simples de fio; podendo alcançar até 20 km com antenas direcionais e ajustes específicos.
- **Interface:** conexão SPI para comunicação com microcontroladores.
- **Tensão de operação:** compatível com sistemas de 3 V e 5 V, graças a um regulador de tensão e conversores de nível integrados.

Sensibilidade do modulo LoRa:

SF	BW (kHz)	CR	Taxa Efetiva (kbps)	Sensibilidade (dBm)	BER Típico	PER Típica	Aplicação
SF5	500	4/5	~62.5	-112 dBm	10 <sup>-4</sup>	10 <sup>-2</sup>	Alta velocidade (industrial)
SF6	500	4/5	~37.5	-117 dBm	10 <sup>-5</sup>	10 <sup>-3</sup>	Curta distância, alta taxa
SF7	250	4/5	~11.7	-120 dBm	10 <sup>-5</sup>	10 <sup>-4</sup>	Balanceado (urbano)
SF7	125	4/5	~5.47	-124 dBm	10 <sup>-6</sup>	10 <sup>-4</sup>	LoRaWAN padrão
SF8	125	4/5	~3.13	-127 dBm	10 <sup>-6</sup>	10 <sup>-5</sup>	Alcance moderado
SF9	125	4/5	~1.76	-130 dBm	10 <sup>-7</sup>	10 <sup>-6</sup>	Áreas rurais/suburbanas
SF10	125	4/5	~0.98	-133 dBm	10 <sup>-7</sup>	<10 <sup>-6</sup>	Longo alcance (baixa interferência)
SF11	125	4/5	~0.49	-136 dBm	10 <sup>-8</sup>	<10 <sup>-7</sup>	Sensibilidade extrema
SF12	125	4/5	~0.29	-139 dBm	10 <sup>-8</sup>	<10 <sup>-8</sup>	Comunicação crítica (IoT rural)
SF12	7.8	4/8	~0.018	-148 dBm	10 <sup>-9</sup>	<10 <sup>-8</sup>	Sensibilidade máxima (LPWAN)

Diferenças Entre Parâmetros:

**Spreading Factor (SF):**

- ↑ SF:
  - **Vantagem:** Melhora a sensibilidade.
  - **Desvantagem:** Reduz drasticamente a taxa de dados.

**Bandwidth (BW):**

- ↓ BW:
  - **Vantagem:** Aumenta a sensibilidade.
  - **Desvantagem:** Reduz a taxa de dados.

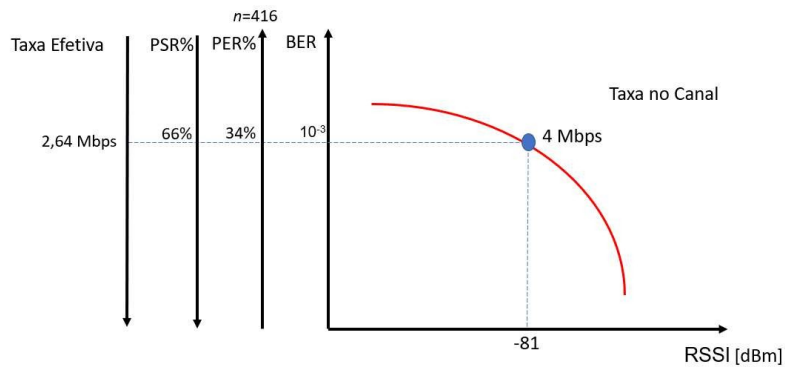
**Coding Rate (CR):**

- ↑ CR (ex: 4/5 → 4/8):
  - **Vantagem:** Reduz o BER.
  - **Desvantagem:** Diminui a taxa efetiva.

A fórmula simplificada para LoRa é:

$$DR \text{ (bps)} = \frac{SF \cdot BW}{2^{SF}} \cdot \frac{4}{4 + CR}$$

Diagrama com o BER e taxa efetiva:



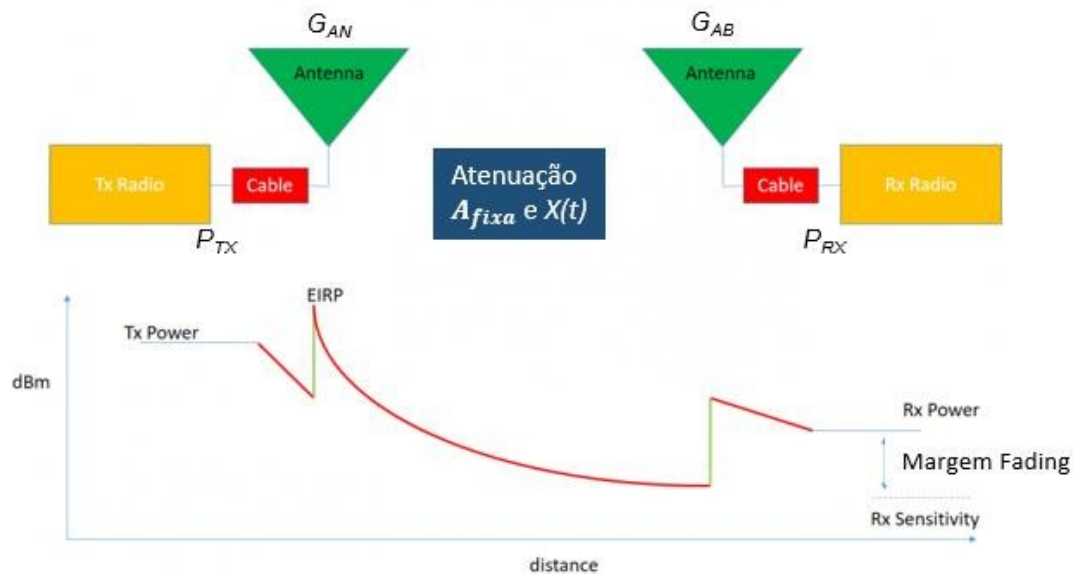
A **taxa efetiva** é a velocidade real de transmissão de dados útil (excluindo cabeçalhos e retransmissões). Em LoRa, é afetada por:

- **Spreading Factor (SF):** SF alto aumenta alcance, mas reduz taxa (ex: SF7 = 5 kbps, SF12 = 0.3 kbps).
- **Largura de Banda (BW):** BW maior aumenta taxa, mas reduz sensibilidade (ex: BW = 125 kHz → 21.9 kbps).
- **PER (Packet Error Rate):** Pacotes perdidos exigem retransmissão, reduzindo taxa efetiva.

**Cálculo:**

$$\text{Taxa Efetiva} = \text{Taxa Bruta} \times (1 - \text{PER}) (\text{bps})$$

## 6. Link Budget e Planejamento



### Fórmula de Friis para Espaço Livre:

$$PRX(t) = PTX + GAN + GAB - Afixa + X(t) - MF - Outros Fatores$$

Onde:

- $PTX$ : Potência de transmissão (ex: 20 dBm).
- $GTX/GRX$ : Ganho das antenas (ex: 9 dBi cada).
- $LEL$ : Atenuação no espaço livre, calculada por:

$$L_{EL}[dB] = 10 \times \log \left( \frac{4\pi d}{\lambda} \right)^2$$

- $d$ : Distância (ex: 2000 m).
- $\lambda$ : Comprimento de onda ( $\lambda = fc$ ,  $f = 915$  MHz).

### Exemplo Prático:

- $PTX = 20$  dBm,  $GTX = GRX = 9$  dBi,  $d = 2000$  m.
- $LEL = 20 \log_{10}(0.3284\pi \times 2000) \approx 99.63$  dB.
- $PRX = 20 + 9 + 9 - 99.63 = -61.63$  dBm.

### Fatores Adicionais:

- **Afixa (Atenuação Fixa)**: Perdas por obstáculos (ex: 10 dB para paredes).
- **X(t)**: Variações de RSSI devido a movimento (ex:  $\pm 5$  dB em ambientes urbanos).
- **MF (Margem de Fading)**: 10-15 dB para compensar instabilidades.



<b>Frequência</b>	<b>915E+06</b>	<b>Hz</b>
Comprimento de onda	0,33	m
Distância	2000	m
Atenuação no espaço livre	99,63	dB
Potência de transmissão	20	dBm
Ganho antena transmissão	5	dB
Ganho antena recepção	5	dB
Potência de recepção	-69,63	dBm

Locais de treino:



Frequência de cada sensor:

Sensores principais:

Sensor	Quantidade	Frequência(Hz)	Total de Frequência(Hz)
RPM	1	200	200
Map	1	200	200
Engine Temp	1	25	25
Oil Pressure	1	25	25
Fuel Pressure	1	25	25
Battery Tension	1	200	200
Steering	1	25	25
G Force	1	50	50
Wheel Speed	4	25	100
<b>Total</b>	12		850

Sensor	Quantidade	Frequência(Hz)	Total de Frequência(Hz)
TPS	1	50	50
Break Temp	4	25	100
Break Pressure	2	25	50
Susp Travel	4	25	100
Gear Position	1	25	25
Oil Temp	1	1	1
Water Temp	4	5	20
<b>Total</b>	17		346

## 8. Cronograma de Implementação


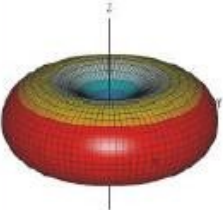
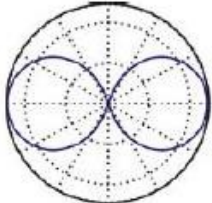
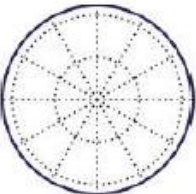
Fase	Ações Detalhadas	Entregáveis
1. Validação	- Medir interferência em 915 MHz com analisador de espectro.	Relatório de interferência e mapa de ruído.
2. Compra	- Aquisição de componentes homologados (E22-900M30S, conversor serial e antenas).	Lista de materiais adquiridos com datasheet.
3. Protótipo	- Montagem do sistema embarcado (Raspberry + Arduino + E22-900M).	Protótipo funcional com transmissão LoRa.
4. Validação de Distância	- Medir RSSI e PER em distâncias crescentes (100m a 2.5km). - Testar antenas LoRa na área de treinamento (alcance, ganho e padrão de radiação). - Medir taxa de canal real em campo (meta: 62,875kbps em 2km).	Tabela de RSSI x Distância e gráfico de PER.  Relatório de validação das antenas.
5. Estruturação de Código	- Desenvolver fila prioritária por sensor (ex: temperatura > RPM). - Desenvolver pacotes para taxa de canal descoberta. - Configurar receptor e prints no serial para teste - Reunião com FTManager para script de criptografia e descriptografia. - Configuração do Mocking	Código em Python/Arduino com comentários. Código para a formação de pacotes adaptado para LoRaWAN. Código para configurar o receptor e imprimir no serial do PC. Documento com os scripts da FTManager. Código para redirecionar os dados para o FTManeger
6. Teste de Bancada	- Integração física dos sensores (RPM, temperatura) ao FT 600. - Simular perda de sinal, falhas de energia e interferência em laboratório.	Diagrama de conexões atualizado.  Relatório de testes com soluções propostas.
7. Case do Receptor	- Projetar caixa à prova d'água para receptor.	Protótipo físico do case.
8. Teste no Carro	- Instalação no veículo e testes dinâmicos (aceleração, curvas, frenagem).	Vídeos e logs de telemetria em tempo real.
9. FSAE	- Ajustes finais e acelerar.	Certificação de funcionamento em evento.



## 9. Futuras Melhorias

Antena Omnidirecionais:

Omnidirecional com Dipolos Empilhados(Dentro do Carro):

Tipo de Antena	Diagrama Tridimensional	Diagrama Vertical ou de Elevação	Diagrama Horizontal ou de Azimute
			
Dipolo de Meia Onda	$G = 2,15 \text{ dBi}$ $G = 0 \text{ dBd}$	Plano Elétrico	Plano Magnético

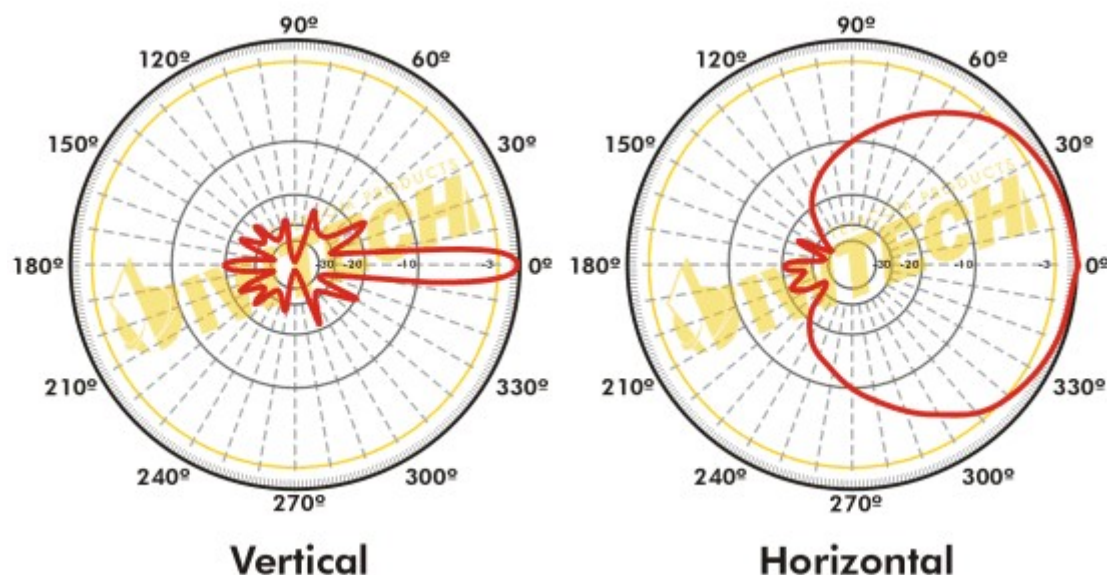
### Antena Laird FG9026 – Especificações Técnicas

- **Frequência de operação:** 902–928 MHz (sintonizada em 915 MHz)
- **Ganho:** 6 dBi
- **Tipo de antena:** Omnidirecional colinear de alto desempenho
- **Polarização:** Vertical
- **Padrão de radiação:** 360° (omnidirecional)
- **Largura de banda:** 26 MHz
- **VSWR (Relação de Onda Estacionária):**  $\leq 1.5:1$
- **Potência máxima suportada:** 100 watts
- **Impedância:** 50 ohms
- **Conector:** N-Fêmea (necessita de adaptador para SMA, se necessário)
- **Material do radome:** Fibra de vidro resistente a UV
- **Comprimento total:** aproximadamente 1,65 metros
- **Peso:** cerca de 1,1 kg
- **Temperatura de operação:** -40°C a +85°C
- **Montagem:** Inclui suporte para mastro (diâmetro de 1,25" a 2")
- **Aplicações típicas:** LoRaWAN, IoT, SCADA, automação industrial, redes ISM 900 MHz

Link de compra: <https://www.arcantenna.com/products/laird-fg9026-65-inch-outdoor-rated-900mhz-fiberglass-omni-antenna-with-fixed-n-female-connector>

Outra opção, ANT-916-CW-HWR-SMA: <https://br.mouser.com/ProductDetail/TE-Connectivity-Linx-Technologies/ANT-916-CW-HWR-SMA?qs=K5ta8V%252BWhtZqsDF9HiOBpg%3D%3D>

Antena Setorial Painei(Base de dados):



## Especificações Técnicas da Laird FG8960

Parâmetro	Valor
Faixa de Frequência	896–960 MHz
Ganho Máximo	9 dBi
Ângulo de Abertura (HPBW)	65° (Horizontal e Vertical)
VSWR	≤ 1.5:1
Impedância	50 Ω
Polarização	Vertical
Conector	N-type (N-fêmea)
Potência Máxima (CW)	100 W
Material do Radome	Polycarbonato UV-resistente
Temperatura de Operação	-40°C a +70°C
Dimensões (A x L x P)	318 x 318 x 85 mm
Peso	1,36 kg
Montagem	Suporte para poste/mastro (incluído)
Certificações	CE, RoHS

Onde comprar: <https://www.arcantenna.com/products/rugged-outdoor-rated-from-896-940-mhz-fiberglass-omnidirectional-antenna-with-fixed-n-female-connector>

Outra opção, OA-915M09-NF: <https://ft-rf.com.tw/915mhz-9dbi-omni-directional-outdoor-antenna>

## 10. Referências

Placa para Estação Meteorológica Colubris: <https://embarcacoes.ic.unicamp.br/colubris/>

Meu repositório: <https://github.com/Faraoh-Back/ProjetoFinalBitDogLab>

Inspiração: <https://embarcados.com.br/lora-arduino-raspberry-pi-shield-dragino/>

Documentação EBYTE E22-900M30S:

<https://www.cdebyte.com/products/E22-900M30S/1#Parameters>

Calculadora Potencia recebida :<https://www.rfwireless-world.com/calculators/wireless-and-cellular/wlan-wifi-range-calculator>

## 11. Sobre o Modulo Lora E22-900M30s:

Pino	Nome	Função	Conexão no Arduino Uno R3
1,2,3,4,5,11,12,20,22	<b>GND</b>	Referência de terra.	Conecte todos ao <b>GND</b> do Arduino.
9,10	<b>VCC</b>	Alimentação (2.5V–5.5V). Recomenda-se usar <b>5V</b> do Arduino.	Conecte ao <b>5V</b> do Arduino.
16	<b>MISO</b>	Saída de dados SPI (Módulo → Arduino). Não usado em transmissão.	Conecte ao <b>MISO (pino 12)</b> .
17	<b>MOSI</b>	Entrada de dados SPI (Arduino → Módulo). <i>Não usado em recepção</i> .	Pode ser deixado desconectado.
18	<b>SCK</b>	Fornece o sinal de clock para sincronizar a comunicação SPI entre o Arduino (mestre) e o módulo E22 (escravo).	Conecte ao pino <b>SCK (13)</b> do Arduino.
19	<b>NSS</b>	Ativa o módulo E22 para comunicação SPI (ativo em LOW).	Conecte a um pino digital (ex: <b>D10</b> ).
14	<b>BUSY</b>	Indica se o módulo está ocupado (HIGH = ocupado, LOW = pronto).	Conecte a um pino digital (ex: <b>D9</b> ).
15	<b>NRST</b>	Reset do módulo (ativo em LOW).	Conecte a um pino digital (ex: <b>D8</b> ).
7	<b>TXEN</b>	Controle de transmissão (LOW para recepção).	Conecte ao <b>GND</b> (modo receptor).
6	<b>RXEN</b>	Controle de recepção (HIGH para recepção).	Conecte ao <b>5V</b> (modo receptor).
8	<b>DIO2</b>	Usado para interrupções (ex: avisar quando dados são recebidos).	Deixe desconectado ou use conforme necessidade.
13	<b>DIO1</b>	Pode ser configurado para eventos secundários (ex: detecção de timeout).	Conecte a um pino digital (ex: <b>D2</b> ).

**Exemplos de uso:**

**a) SDK(Serial Clock) – Pino 18**

```
#include <SPI.h>

void setup() {
    SPI.begin(); // Inicia a comunicação SPI (SCK é configurado automaticamente)
}
```

**b) NSS (Slave Select) – Pino 19**

```
#define NSS_PIN 10

void setup() {
    pinMode(NSS_PIN, OUTPUT);
    digitalWrite(NSS_PIN, HIGH); // Desativa o módulo inicialmente
}

void loop() {
    digitalWrite(NSS_PIN, LOW); // Ativa o módulo
    // Comunicação SPI aqui...
    digitalWrite(NSS_PIN, HIGH); // Desativa o módulo
}
```

**c) BUSY – Pino 14**

```
#define BUSY_PIN 9

void setup() {
    pinMode(BUSY_PIN, INPUT);
}

void loop() {
    if (digitalRead(BUSY_PIN) == LOW) {
        // Módulo está pronto para receber dados
    }
}
```

**d) NRST(Reset) – Pino 15**

```
#define NRST_PIN 8

void resetModule() {
    digitalWrite(NRST_PIN, LOW);
    delay(10);
    digitalWrite(NRST_PIN, HIGH);
    delay(100); // Tempo para reinicialização
}
```

```
}
```

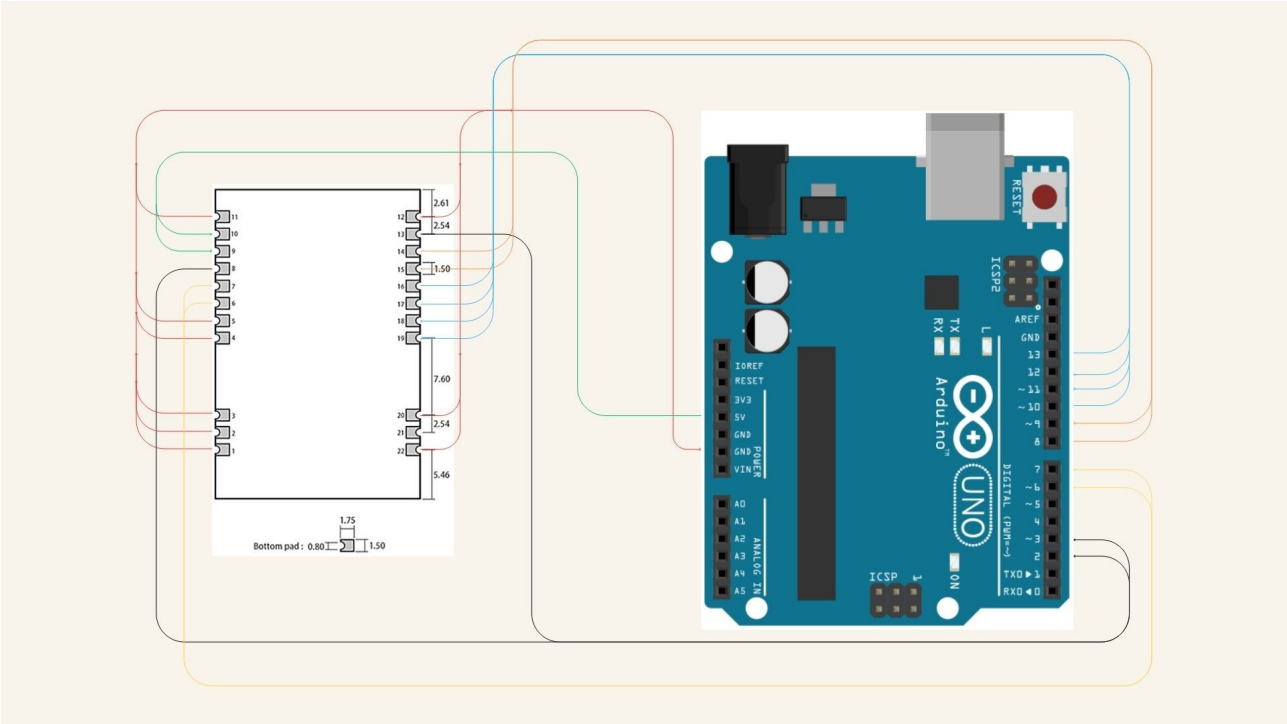
**e) TXEN e RXEN – Pinos 7 e 6**

```
void setup() {  
    pinMode(7, OUTPUT); // TXEN  
    pinMode(6, OUTPUT); // RXEN  
    digitalWrite(7, LOW); // Desativa TX  
    digitalWrite(6, HIGH); // Ativa RX  
}
```

**f) DIO1 e DIO2 – Pinos 13 e 8**

```
#define DIO1_PIN 2  
  
volatile bool dataReceived = false;  
  
void setup() {  
    pinMode(DIO1_PIN, INPUT);  
    attachInterrupt(digitalPinToInterrupt(DIO1_PIN), onDataReceived, RISING);  
}  
  
void onDataReceived() {  
    dataReceived = true; // Sinaliza que dados estão prontos  
}  
  
void loop() {  
    if (dataReceived) {  
        // Lê dados via SPI  
        dataReceived = false;  
    }  
}
```

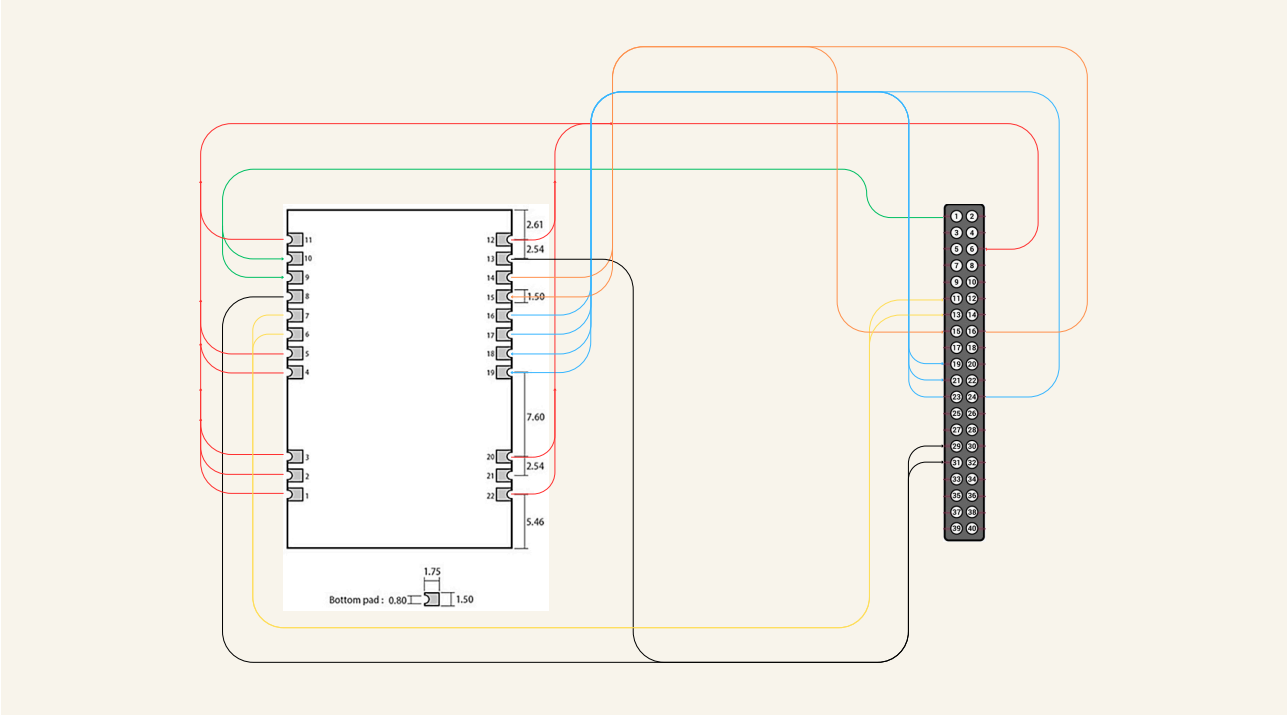
Diagrama receptor:



Legenda

Cores	Função	Pino E22	Pino Arduino
Verde	Alimentação (5V)	9,10	5V
Vermelho	Alimentação (GND)	1,2,3,4,5,11,12,20,22	GND
Azul	Comunicação SPI	16	12
		17	11
		18	13
		19	10
Amarelo	Controle de Modo	7	6
		6	7
Laranja	Monitoramento de Estado	14	9
		15	8
Preto	Interrupção de Recepção (Opcional)	13	2
		8	3

Diagrama Transmissor



Legenda

Cores	Função	Pino E22	Pino Arduino
Verde	Alimentação (5V)	9,10	3,3V
Vermelho	Alimentação (GND)	1,2,3,4,5,11,12,20,22	GND
		16	21
Azul	Comunicação SPI	17	19
		18	23
		19	24
Amarelo	Controle de Modo	7	11
		6	13
Laranja	Monitoramento de Estado	14	15
		15	16
Preto	Interrupção de Recepção (Opcional)	13	29
		8	31

Dicas para Eficiência e Integridade do Sistema:

1. Alimentação Estável:

- Use capacitores cerâmicos (100nF) entre **VCC** e **GND** próximos ao módulo E22 para filtrar ruídos.
- Evite compartilhar a fonte de alimentação com motores ou componentes de alta potência.

## 2. Controle de Modo (TXEN/RXEN):

- Nunca ative **TXEN** e **RXEN** simultaneamente.
- Adicione um delay de **10 ms** após alternar os modos para estabilização.

## 3. Comunicação SPI:

- Mantenha o **NSS (Slave Select)** em **HIGH** quando não estiver em uso para evitar conflitos no barramento.
- Use taxas de clock SPI moderadas (ex: 1 MHz) para garantir estabilidade em ambientes ruidosos.

## 4. Antena:

- Posicione a antena longe de superfícies metálicas para evitar interferências.
- Use cabos coaxiais curtos e de alta qualidade para conexões.

## 5. Reset e Monitoramento:

- Implemente um watchdog timer para reiniciar o módulo automaticamente em caso de falha (via pino **NRST**).
- Monitore o pino **BUSY** para evitar sobrecarga do módulo.

# Código 1: Raspberry Pi 5 (Transmissor Default)

## Funcionalidades:

- Envia dados periódicos via LoRa.
- Alterna para modo receptor se receber um comando de alerta.
- Usa interrupção via **DIO1** para recepção.

```
import spidev
import RPi.GPIO as GPIO
import time

# Configuração de pinos
TXEN_PIN = 17    # GPIO 17 (Transmissão)
RXEN_PIN = 27    # GPIO 27 (Recepção)
BUSY_PIN = 22    # GPIO 22 (Estado)
NRST_PIN = 23    # GPIO 23 (Reset)
DIO1_PIN = 5     # GPIO 5 (Interrupção)
BUTTON_PIN = 24  # GPIO 24 (Botão para teste)
```



```
# Inicialização SPI
spi = spidev.SpiDev()
spi.open(0, 0)
spi.max_speed_hz = 1000000

def setup_gpio():
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(TXEN_PIN, GPIO.OUT)
    GPIO.setup(RXEN_PIN, GPIO.OUT)
    GPIO.setup(BUSY_PIN, GPIO.IN)
    GPIO.setup(NRST_PIN, GPIO.OUT)
    GPIO.setup(DI01_PIN, GPIO.IN, pull_up_down=GPIO.PUD_UP)
    GPIO.setup(BUTTON_PIN, GPIO.IN, pull_up_down=GPIO.PUD_UP)
    GPIO.add_event_detect(DI01_PIN, GPIO.FALLING,
callback=receive_interrupt, bouncetime=100)

def set_transmitter_mode():
    GPIO.output(TXEN_PIN, GPIO.HIGH)
    GPIO.output(RXEN_PIN, GPIO.LOW)
    time.sleep(0.01)

def set_receiver_mode():
    GPIO.output(TXEN_PIN, GPIO.LOW)
    GPIO.output(RXEN_PIN, GPIO.HIGH)
    time.sleep(0.01)

def reset_module():
    GPIO.output(NRST_PIN, GPIO.LOW)
    time.sleep(0.1)
    GPIO.output(NRST_PIN, GPIO.HIGH)
    time.sleep(0.5)
```

```

def send_data(data):
    if GPIO.input(BUSY_PIN) == GPIO.LOW:
        set_transmitter_mode()
        spi.xfer2(data)
        set_receiver_mode()

def receive_interrupt(channel):
    if GPIO.input(BUSY_PIN) == GPIO.LOW:
        received = spi.readbytes(10)
        print(f"Alerta recebido: {bytes(received).decode()}")

# Main
setup_gpio()
reset_module()
set_receiver_mode()

try:
    while True:
        # Envia dados a cada 5 segundos
        send_data([0x48, 0x65, 0x6C, 0x6C, 0x6F]) # "Hello"
        time.sleep(5)

        # Verifica botão de teste
        if GPIO.input(BUTTON_PIN) == GPIO.LOW:
            send_data([0x41, 0x4C, 0x45, 0x52, 0x54]) # "ALERT"
            time.sleep(1)
except KeyboardInterrupt:
    spi.close()
    GPIO.cleanup()

```

## Código 2: Arduino Uno R3 (Receptor Default)

### Funcionalidades:

- Recebe dados e aciona alertas.
- Alterna para transmissor se um botão for pressionado.
- Usa interrupção via **DIO1** para recepção.

```
#include <SPI.h>
```

```
// Definição de pinos
```

```
#define TXEN_PIN 6
```

```
#define RXEN_PIN 7
```

```
#define BUSY_PIN 9
```

```
#define NRST_PIN 8
```

```
#define DIO1_PIN 2
```

```
#define BUTTON_PIN 3
```

```
volatile bool dataReceived = false;
```

```
void setTransmitterMode() {  
    digitalWrite(TXEN_PIN, HIGH);  
    digitalWrite(RXEN_PIN, LOW);  
    delay(10);  
}
```

```
void setReceiverMode() {  
    digitalWrite(TXEN_PIN, LOW);  
    digitalWrite(RXEN_PIN, HIGH);  
    delay(10);  
}
```

```
void resetModule() {  
    digitalWrite(NRST_PIN, LOW);  
    delay(100);  
}
```

```

    digitalWrite(NRST_PIN, HIGH);
    delay(500);
}

void onDataReceived() {
    dataReceived = true;
}

void setup() {
    pinMode(TXEN_PIN, OUTPUT);
    pinMode(RXEN_PIN, OUTPUT);
    pinMode(BUSY_PIN, INPUT);
    pinMode(NRST_PIN, OUTPUT);
    pinMode(DIO1_PIN, INPUT);
    pinMode(BUTTON_PIN, INPUT_PULLUP);
    attachInterrupt(digitalPinToInterrupt(DIO1_PIN), onDataReceived,
FALLING);

    SPI.begin();
    setReceiverMode();
    resetModule();
    Serial.begin(9600);
}

void loop() {
    // Recepção de dados
    if (dataReceived && digitalRead(BUSY_PIN) == LOW) {
        digitalWrite(SS, LOW);
        byte data[10];
        for (int i = 0; i < 10; i++) {
            data[i] = SPI.transfer(0x00);
        }
    }
}

```

```
digitalWrite(SS, HIGH);  
Serial.print("Dados recebidos: ");  
for (byte b : data) Serial.write(b);  
Serial.println();  
dataReceived = false;  
}
```

```
// Botão para envio de alerta  
if (digitalRead(BUTTON_PIN) == LOW) {  
    setTransmitterMode();  
    digitalWrite(SS, LOW);  
    SPI.transfer(0x41); // 'A'  
    SPI.transfer(0x4C); // 'L'  
    SPI.transfer(0x45); // 'E'  
    SPI.transfer(0x52); // 'R'  
    SPI.transfer(0x54); // 'T'  
    digitalWrite(SS, HIGH);  
    setReceiverMode();  
    delay(1000);  
}  
}
```

**Referencias:**

Placa para Estação Meteorológica Colubris: <https://embarcacoes.ic.unicamp.br/colubris/>

Meu repositório: <https://github.com/Faraoh-Back/ProjetoFinalBitDogLab>

Inspiração: <https://embarcados.com.br/lora-arduino-raspberry-pi-shield-dragino/>

Documentação Adafruit RFM65W : <https://learn.adafruit.com/adafruit-rfm69hcw-and-rfm96-rfm95-rfm98-lora-packet-padio-breakouts>

DataSheet – SX1276:

[https://semtech.my.salesforce.com/sfc/p/#E0000000JelG/a/2R00000001Rbr/6EfVZUorrpoKFfvaF\\_Fkpgp5kzjiNyiAbqcpqh9qSjE](https://semtech.my.salesforce.com/sfc/p/#E0000000JelG/a/2R00000001Rbr/6EfVZUorrpoKFfvaF_Fkpgp5kzjiNyiAbqcpqh9qSjE)

Calculadora Potencia recebida :<https://www.rfwireless-world.com/calculators/wireless-and-cellular/wlan-wifi-range-calculator>

