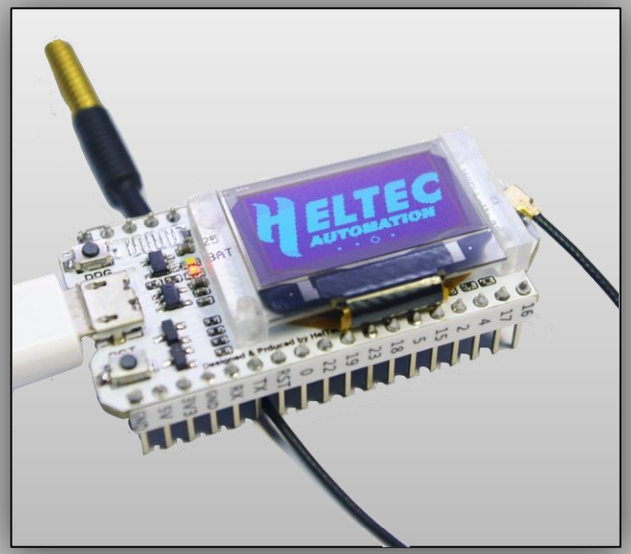
LoRa Send Receive



Por Fernando Koyanagi

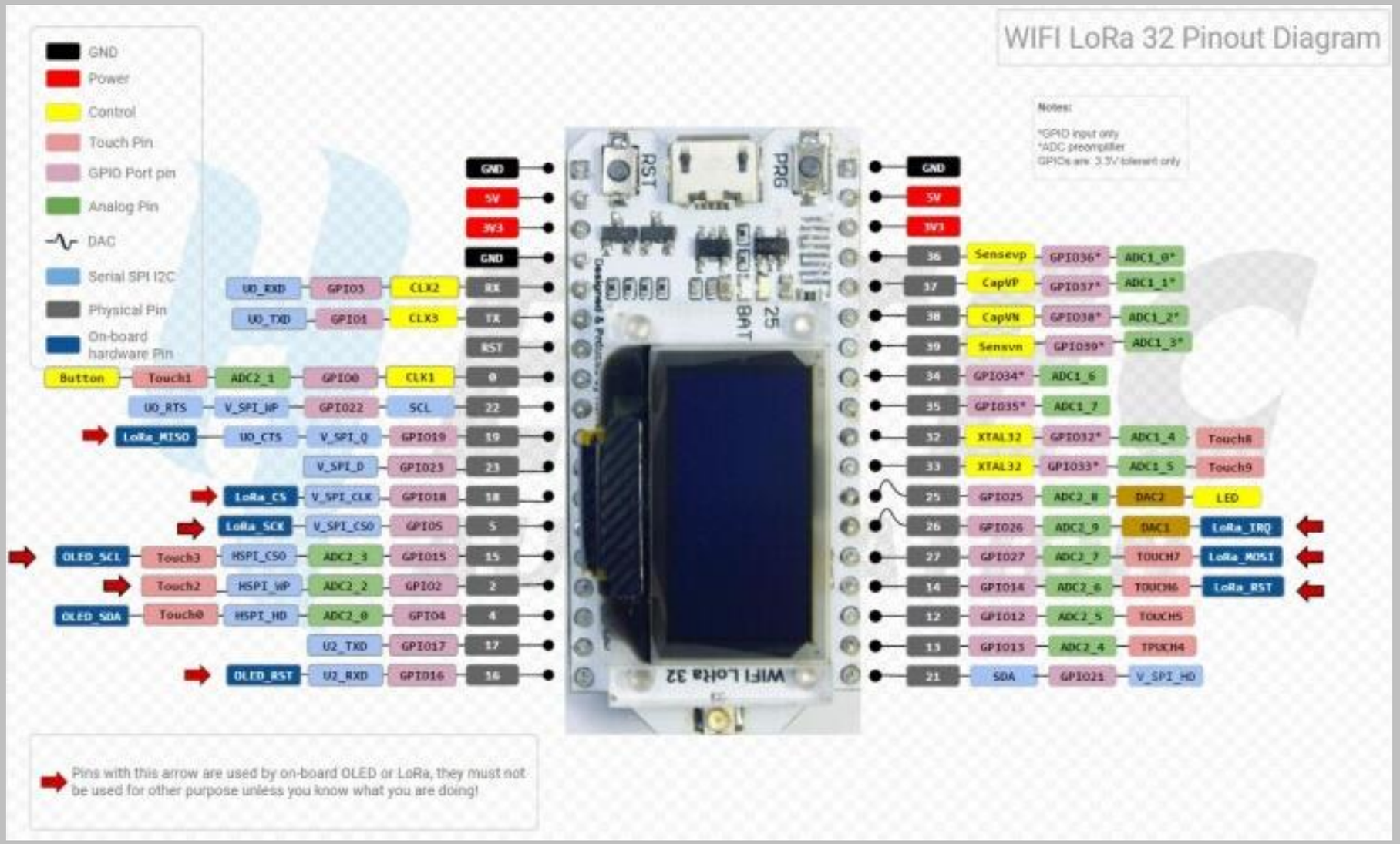
Intenção dessa aula

1. Utilizar LoRa para enviar e receber dados

2. Um dispositivo enviará um pacote avisando que quer receber os dados

3. O outro dispositivo ficará aguardando o comando para enviar os dados de volta para o primeiro dispositivo

LoRa ESP32 Pinout

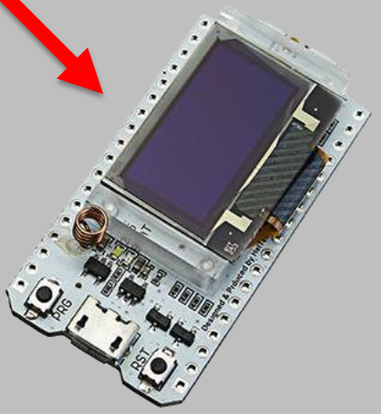


Comunicação

1 - O Master envia um pacote indicando que deseja receber os dados do Slave

2 - O Slave recebe e verifica este pacote

Master

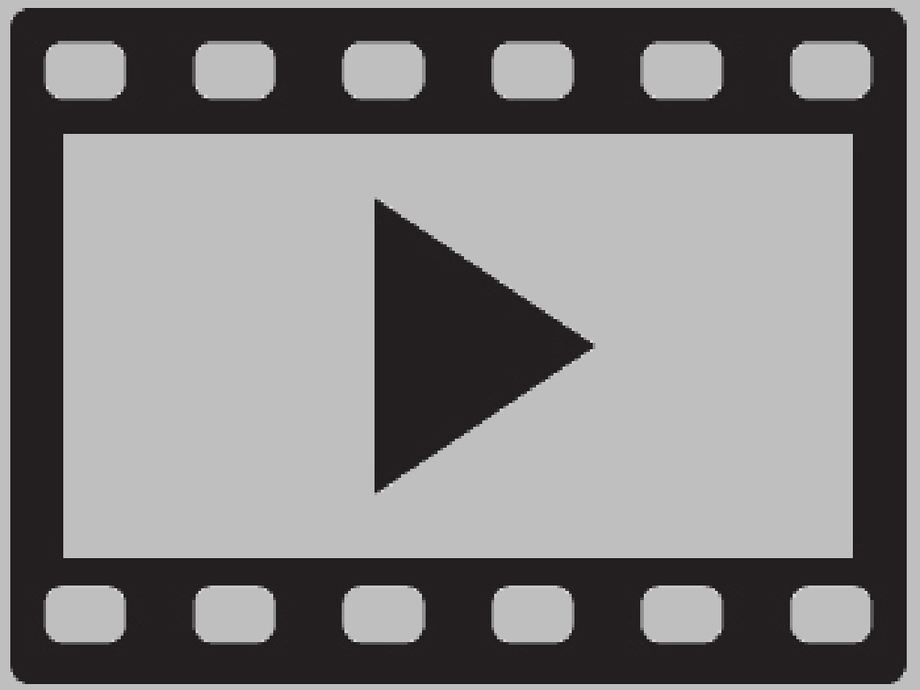


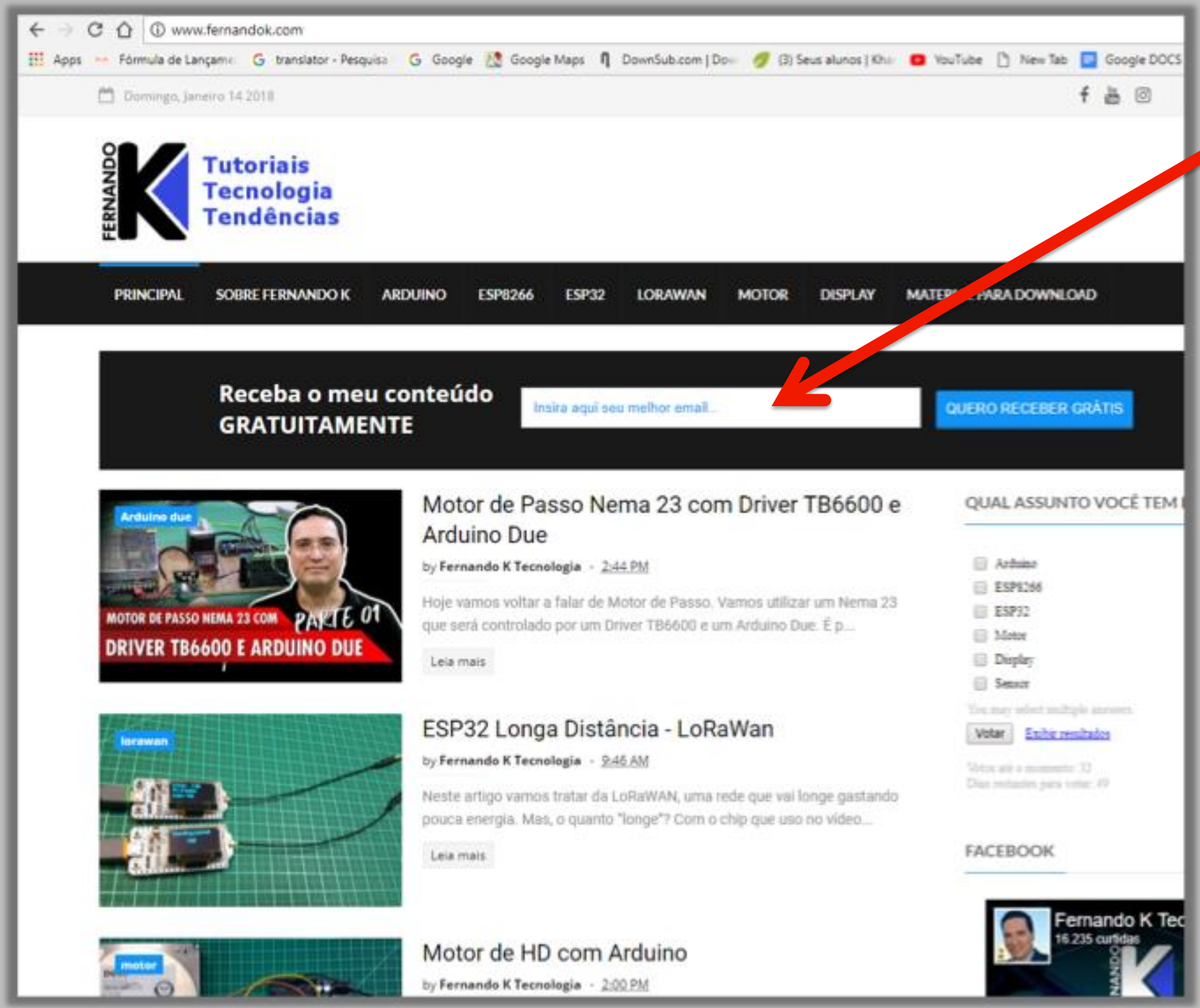
Slave

3 - O Slave envia de volta um pacote com os dados

4 - O Master recebe, verifica o pacote e exibe os dados

Demonstração





Seu e-mail

Inscreva-se

LoRaSendReceive.ino

```
#include <SPI.h>
#include <LoRa.h>
#include <Wire.h>
#include <SSD1306.h>

//Deixe esta linha descomentada para compilar o Master
//Comente ou remova para compilar o Slave
#define MASTER

#define SCK 5    // GPIO5  SCK
#define MISO 19 // GPIO19 MISO
#define MOSI 27 // GPIO27 MOSI
#define SS 18    // GPIO18 CS
#define RST 14  // GPIO14 RESET
#define DI00 26 // GPIO26 IRQ(Interrupt Request)

#define BAND 433E6 //Frequência do radio - exemplo : 433E6, 868E6, 915E6

//Constante para informar ao Slave que queremos os dados
const String GETDATA = "getdata";
//Constante que o Slave retorna junto com os dados para o Master
const String SETDATA = "setdata=";

//Variável para controlar o display
SSD1306 display(0x3c, 4, 15);
```

LoRaSendReceive.ino - setupDisplay

```
void setupDisplay(){
  //O estado do GPIO16 é utilizado para controlar o display OLED
  pinMode(16, OUTPUT);
  //Reseta as configurações do display OLED
  digitalWrite(16, LOW);
  //Para o OLED permanecer ligado, o GPIO16 deve permanecer HIGH
  //Deve estar em HIGH antes de chamar o display.init() e fazer as
    //demais configurações, não inverta a ordem
  digitalWrite(16, HIGH);

  //Configurações do display
  display.init();
  display.flipScreenVertically();
  display.setFont(ArialMT_Plain_10);
  display.setTextAlignment(TEXT_ALIGN_LEFT);
}
```

LoRaSendReceive.ino - setupLoRa

```
//Configurações iniciais do LoRa
void setupLoRa(){
  //Inicializa a comunicação
  SPI.begin(SCK, MISO, MOSI, SS);
  LoRa.setPins(SS, RST, DI00);

  //Inicializa o LoRa
  if (!LoRa.begin(BAND, true)){
    //Se não conseguiu inicializar, mostra uma mensagem no display
    display.clear();
    display.drawString(0, 0, "Erro ao inicializar o LoRa!");
    display.display();
    while (1);
  }

  //Ativa o crc
  LoRa.enableCrc();
  //Ativa o recebimento de pacotes
  LoRa.receive();
}
```

Master.ino

```
//Compila apenas se MASTER estiver definido no arquivo
//principal
#ifdef MASTER

//Intervalo entre os envios
#define INTERVAL 1000

//Tempo do último envio
long lastSendTime = 0;
```

Master.ino - setup

```
void setup(){
  Serial.begin(115200);
  //Chama a configuração inicial do display
  setupDisplay();
  //Chama a configuração inicial do LoRa
  setupLoRa();

  display.clear();
  display.drawString(0, 0, "Master");
  display.display();
}
```

Master.ino - loop

```
void loop(){
  //Se passou o tempo definido em INTERVAL desde o último
  //envio
  if (millis() - lastSendTime > INTERVAL){
     //Marcamos o tempo que ocorreu o último envio
     lastSendTime = millis();
     //Envia o pacote para informar ao Slave que queremos
     //receber os dados
     send();
  }

  //Verificamos se há pacotes para recebermos
  receive();
}
```

Master.ino - receive 1/2

```
void receive(){
  //Tentamos ler o pacote
  int packetSize = LoRa.parsePacket();

  //Verificamos se o pacote tem o tamanho mínimo de caracteres
  //que esperamos
  if (packetSize > SETDATA.length()){
    String received = "";
    //Armazena os dados do pacote em uma string
    while(LoRa.available()){
      received += (char) LoRa.read();
    }
    //Verifica se a string possui o que está contido em
    //“SETDATA”
    int index = received.indexOf(SETDATA);
```

Master.ino - receive 2/2

```
if(index >= 0){
    //Recuperamos a string que está após o "SETDATA",
    //que no caso serão os dados de nosso interesse
    String data = received.substring(SETDATA.length());
    //Tempo que demorou para o Master criar o pacote, enviar o pacote,
    //o Slave receber, fazer a leitura, criar um novo pacote, enviá-lo
    //e o Master receber e ler
    String waiting = String(millis() - lastSendTime);
    //Mostra no display os dados e o tempo que a operação demorou
    display.clear();
    display.drawString(0, 0, "Recebeu: " + data);
    display.drawString(0, 10, "Tempo: " + waiting + "ms");
    display.display();
  }
 }
}

#endif
```

Slave.ino

```
//Compila apenas se MASTER não estiver definido no arquivo principal
#ifndef MASTER

//Contador que irá servir como o dados que o Slave irá enviar
int count = 0;
```

Slave.ino

```
void setup(){
    Serial.begin(115200);
    //Chama a configuração inicial do display
    setupDisplay();
    //Chama a configuração inicial do LoRa
    setupLoRa();
    display.clear();
    display.drawString(0, 0, "Slave esperando...");
    display.display();
}
```

Slave.ino - loop 1/2

```
void loop(){
  //Tenta ler o pacote
  int packetSize = LoRa.parsePacket();

  //Verifica se o pacote possui a quantidade de caracteres que
  //esperamos
  if (packetSize == GETDATA.length()){
    String received = "";

    //Armazena os dados do pacote em uma string
    while(LoRa.available()){
      received += (char) LoRa.read();
    }
```

Slave.ino - loop 2/2

```
if(received.equals(GETDATA)){
      //Simula a leitura dos dados
      String data = readData();
      Serial.println("Criando pacote para envio");
      //Cria o pacote para envio
      LoRa.beginPacket();
      LoRa.print(SETDATA + data);
      //Finaliza e envia o pacote
      LoRa.endPacket();
      //Mostra no display
      display.clear();
      display.drawString(0, 0, "Enviou: " + String(data));
      display.display();
    }
  }
}
```

Slave.ino - loop 2/2

```
//Função onde se faz a leitura dos dados que queira enviar
//Poderia ser o valor lido por algum sensor por exemplo
//Aqui vamos enviar apenas um contador para testes
//mas você pode alterar a função para fazer a leitura de
//algum sensor
String readData(){
  return String(count++);
}

#endif
```

Em www.fernandok.com

Download arquivos PDF e INO do código fonte