

Elektrotechniek

- [Soundboard met DFPlayer Mini / MP3-TF-16P](#)
- [Lijn volg robot](#)
- [ESP32 + Playstation controller](#)
- [Lorawan - RAK3172 + The Things Network \(TTN\)](#)
- [RAK3272S Digital I/O's](#)
- [LoraWan - Geocaching](#)

Soundboard met DFPlayer Mini / MP3-TF-16P

Dit project is een work in progress

Dit idee is ontstaan nadat ik op YouTube een filmpje zag van de **“Seagull Keychain”**. Dit is een kleine pushbutton die het bekende [“Mine, mine, mine!”](#)-geluid uit de film *Finding Nemo* afspeelt. Het idee was meteen: *dat kan ik beter*. En zo ben ik met dit project aan de slag gegaan.

Naarmate ik verder met het project aan de slag ging, ontstond al snel het idee: **waarom zou ik het bij één geluid houden?** Waarom niet meerdere knoppen met verschillende geluiden? Zo veranderde het oorspronkelijke idee al snel in een compleet **soundboard**, waarbij iedere knop zijn eigen geluidsfragment kan afspelen.

De eerste uitdaging was natuurlijk: **wat ga ik hiervoor gebruiken?** Ik had immers een oplossing nodig waarop ik meerdere geluidsfragmenten kon opslaan en afspelen.

Mijn eerste idee was om een **ESP32** te gebruiken in combinatie met een SD-kaartlezer en vanaf daar verder te kijken. Tijdens mijn zoektocht kwam ik echter de **MP3-TF-16P** tegen. Dit is een clone van de **DFPlayer Mini** en bleek eigenlijk alles te bieden wat ik voor dit project nodig had:

- Ondersteuning voor een SD-kaart
- Geluidsfragmenten eenvoudig aan te roepen met knoppen
- Compact en relatief eenvoudig te gebruiken
- Goedkoop
- Een ingebouwde 3 watt DAC-versterker

Hierdoor hoefde ik niet zelf een complete oplossing voor het opslaan en afspelen van audio te bouwen. De MP3-TF-16P nam dit grotendeels voor zijn rekening, waardoor ik me kon focussen op de rest van het project.

Lijn volg robot

Projectje wat ik met mijn zoon aan het maken ben

https://projecthub.arduino.cc/lee_curiosity/building-a-line-following-robot-using-arduino-017dbb

ESP32 + Playstation controller

<https://www.cytron.io/tutorial/esp32-ps4controller-beginner?srsId=AfmBOorz0MuLSKF-G4ilAa7tnePnd7oyDFwj9cswZ3JMjLUdG0eK4yv3>

Lorawan - RAK3172 + The Things Network (TTN)

Wat is LoRaWAN?

LoRaWAN is een draadloos communicatienetwerk voor het verbinden van apparaten over grote(re) afstanden. Het is energiezuinig, betrouwbaar en geschikt voor toepassingen waarbij sensoren kleine hoeveelheden data versturen. Denk bijvoorbeeld aan monitoring van temperatuur, luchtkwaliteit, waterstanden of energieverbruik.

The Things Network (TTN)

Voor LoRaWAN zijn verschillende netwerken en platformen beschikbaar. Voor dit project heb ik gekozen voor The Things Network (TTN). Een belangrijke reden hiervoor is dat TTN gratis te gebruiken is en daarmee een laagdrempelige manier biedt om met LoRaWAN te experimenteren en toepassingen te ontwikkelen.

RAK3172 / RAK3272S

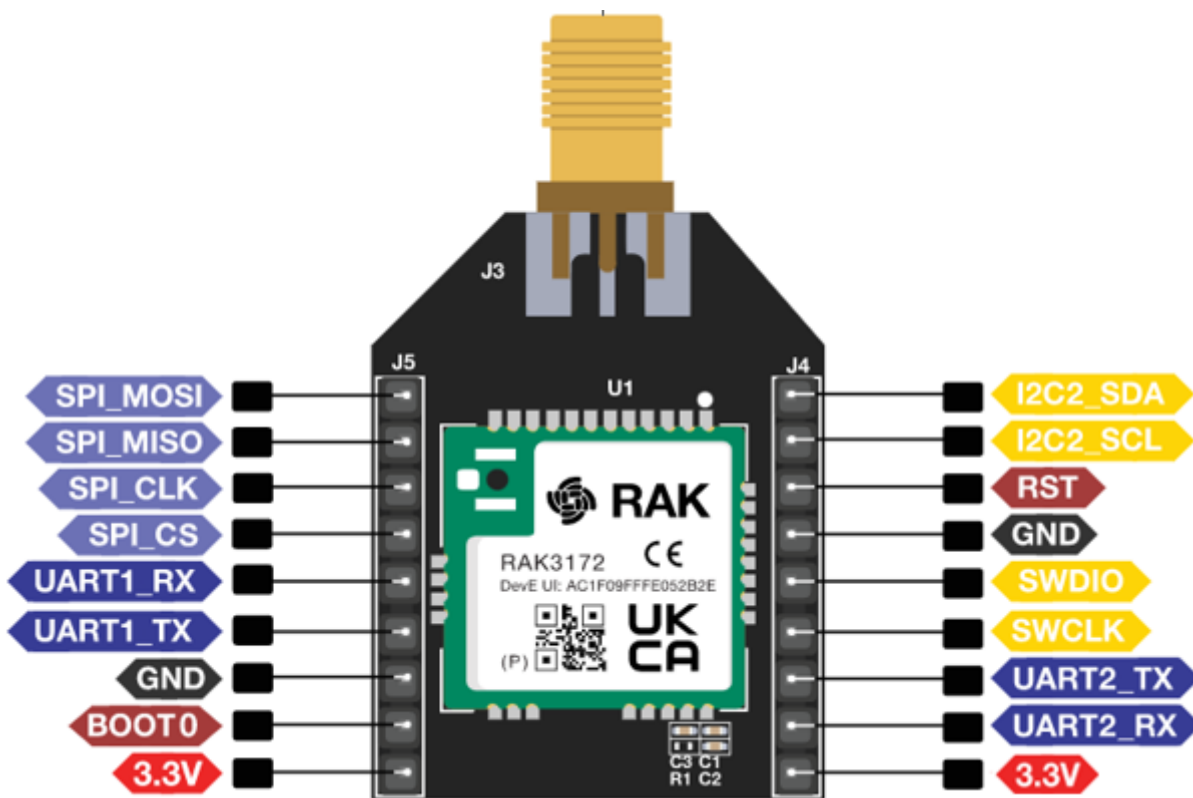
Er wordt gebruikgemaakt van de RAK3172. Omdat dit een SMD-module is, kan ook de RAK3272S Proto Board worden gebruikt. Hierop is de RAK3172 al voormonteerd. Het bord is energiezuinig en eenvoudig te programmeren, waardoor het geschikt is voor het testen van sensoren en IoT-toepassingen.

Zorg ervoor dat de antenne is aangesloten voordat je de RAK3172 inschakelt. Zonder antenne kan de module onherstelbaar beschadigd raken.

De reden dat ik voor de RAK3272S heb gekozen, is omdat deze een microcontroller bevat voor het aansluiten van externe sensoren. Daarnaast heeft het bord een laag energieverbruik en is er duidelijke documentatie beschikbaar.

Kies je voor de RAK3172 om je eigen board te ontwikkelen, dan heb je nog meer programmeerbare pinnen tot je beschikking voor externe toepassingen.

<https://docs.rakwireless.com/product-categories/wisduo/rak3272s-breakout-board/>



Om met de module te communiceren heb je een USB-Serial adapter nodig. Je kan hiervoor de RAKDAP1 gebruiken, maar elke andere USB-Serial adapter met 3.3V uitgang werkt ook.

De module werkt op **3,3 V**. Sluit **nooit 5 V** aan, want hierdoor kan de module onherstelbaar beschadigd raken.

Je sluit het als volgt aan.

USB-Serial	RAK3272S
3.3V	3.3V
GND	GND
TX	UART1_RX
RX	UART1_TX

Hierna is het een kwestie van je favoriete terminalprogramma te openen, bijvoorbeeld **PuTTY** voor Windows of **CuteCom** voor Linux. En gebruik onderstaande instellingen.

Baudrate	115200
Databits	8
Pariteit	Geen (N)
Stopbits	1

Als je nu het commando AT stuurt zou je OK terug moeten krijgen.

Het kan voorkomen dat je geen OK terugkrijgt, maar een melding dat het commando niet wordt herkend. In dat geval staat de module niet in boot mode.

Geef eerst het commando AT+BOOT. Eventueel kun je daarna ATZ als commando gebruiken om de module opnieuw op te starten.

Nu je de module succesvol hebt aangesloten, moeten we deze eerst correct instellen. Voer hiervoor de onderstaande commando's in.

AT+NJM=1	zet de RAK3172 in OTAA-modus (Over-The-Air Activation).
AT+BAND=4	stelt de RAK3172 in op de EU868-frequentieband.
AT+CLASS=A	stelt de RAK3172 in op LoRaWAN Class A.

Alle commando's worden gevolgd door een OK-melding wanneer ze succesvol zijn uitgevoerd.

Nu is het tijd om je module aan te melden bij TTN (The Things Network). Hiervoor heb je eerst twee waarden uit de module nodig. Noteer deze bijvoorbeeld in Kladblok, zodat je ze later eenvoudig kunt terugvinden.

AT+DEVEUI=?	Toont de huidige DevEUI van de RAK3172. Deze heb je nodig voor het DevEUI veld bij TTN
AT+APPEUI=?	Toont de huidige AppEUI van de RAK3172. Deze heb je nodig voor het JoinEUI veld bij TTN

Ik ga er vanuit dat je al een account bij TTN hebt. Zo niet dan kan je die eenvoudig aanmaken via: <https://eu1.cloud.thethings.network/>

Maak als eerst een applicatie aan bij TTN. Je mag zelf hier iets van maken.

[Create application - Console - The Things Network](#)

Wanneer de applicatie is aangemaakt kun je het end device toevoegen.

[Register end device - geocache - The Things Network](#)

Op het moment van schrijven staat de RAK3172 nog niet in de Device Repository. Daarom moet je bij het toevoegen van het apparaat kiezen voor de optie Manual. Geeft onderstaande waardes op.

Frequency plan	Europe 863-870 MHz (SF9 for RX2 - recommended)
LoRaWAN Version	LoRaWAN Specification 1.0.3
Regional Parameters version	RP001 Regional Parameters 1.0.3 revision A

Er wordt nu gevraagd om de JoinEUI. Deze heb je eerder opgevraagd met het commando AT+APPEUI=?. Vul deze waarde in en klik op Confirm. Vervolgens wordt gevraagd om de DevEUI. Deze heb je eerder opgevraagd met AT+DEVEUI=?. Vul deze in en klik bij AppKey op Generate. De overige velden mag je naar eigen wens invullen.

Hierna klik je op Register end device en zie je links jouw device staan.

Let op: sla de gegenereerde AppKey goed op.

Het enige wat je nu nog moet doen is de AppKey in jouw RAK3172 te zetten en het netwerk te joinen. De key voeg je toe met onderstaande commando.

```
AT+APPKEY=hier_jouw_gegenereerde_key
```

Selecteer nu jouw device in TTN en open het tabblad Live data. Hierna voer je onderstaande commando in bij je RAK3172.

```
AT+JOIN
```

Hierna volgt OK en even later +EVT:JOINED wanneer alles goed is verlopen. Je zie ook in het Live data tabblad een join en accept log.

```
14:20:13 Forward join-accept message
14:20:11 Successfully processed join-request
14:20:11 Accept join-request
```

Het kan voorkomen dat je bij het aanmelden voor +EVT:JOINED één of meerdere keren de melding +EVT:JOIN_FAILED_* krijgt. Vooral bij de eerste keer aanmelden kan het even duren voordat alles goed werkt. Wacht daarom een paar minuten en probeer het daarna opnieuw. In de meeste gevallen werkt het vervolgens gewoon.

Krijg je de melding +EVT:JOIN_FAILED_RX_TIMEOUT, dan is er aan één van de twee kanten iets verkeerd ingevuld. Het kan ook zijn dat er simpelweg geen of onvoldoende bereik is. Kijk dan of je even buiten de join kan uitvoeren.

Nu je jouw device hebt aangemeld kunnen we daadwerkelijk data versturen. Gebruik hiervoor bijvoorbeeld het onderstaande commando:

```
AT+SEND=1:48656C6C6F
```

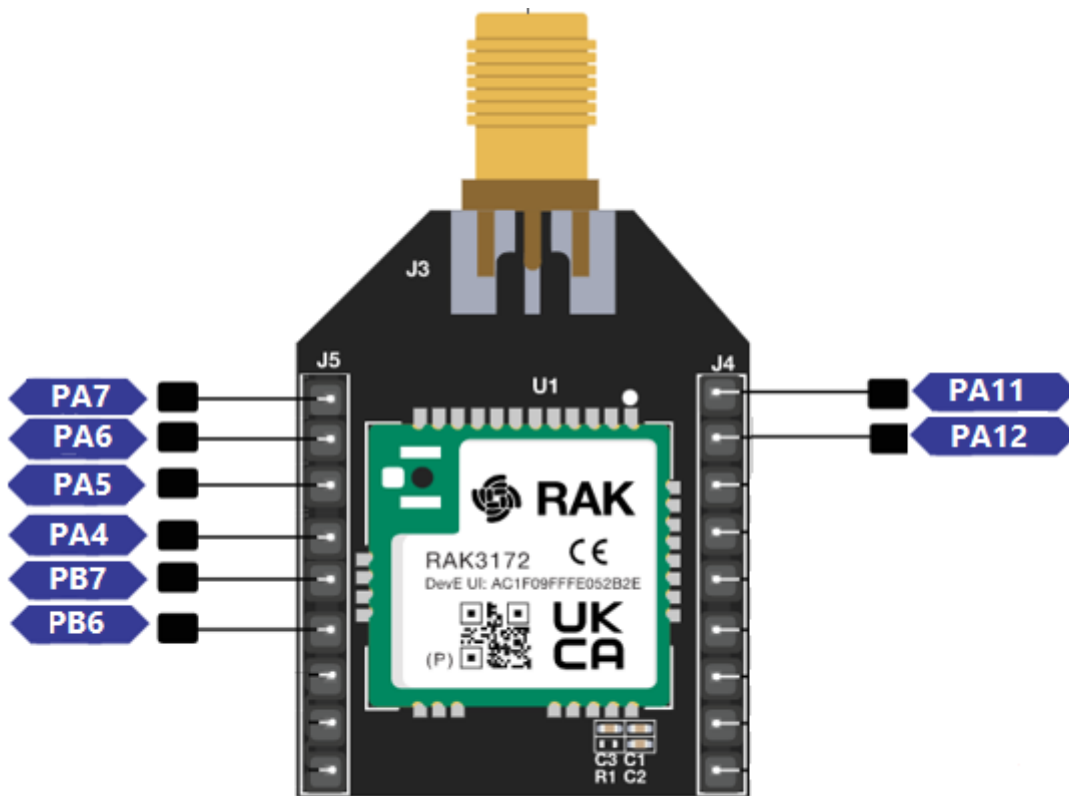
Hiermee verstuur je de hex-data 48 65 6C 6C 6F, oftewel "Hello". Het succesvol versturen van de data wordt vervolgens bevestigd met de melding +EVT:TX_DONE. In het Live data tabblad van TTN krijg je nou jouw berichtje binnen.

```
Forward uplink data message DevAddr * 48656C6C6F FPort:1 Data rate SF10BW125 SNR -6 RSSI -105
```

RAK3272S Digital I/O's

<https://docs.rakwireless.com/product-categories/wisduo/rak3272s-breakout-board/quickstart/#how-to-use-digital-io>

Pin Name	J Connector pin
PA4	J5 pin 4
PA5	J5 pin 3
PA6	J5 pin 2
PA7	J5 pin 1
PA8	J4 pin 1
PA9	J4 pin 2
PB6	J5 pin 5
PB7	J5 pin 6



```
void setup()
{
    pinMode(PA7, OUTPUT); //Change the P0_04 to any digital pin you want. Also, you can set this
    to INPUT or OUTPUT
}

void loop()
{
    digitalWrite(PA7,HIGH); //Change the PA0 to any digital pin you want. Also, you can set this
    to HIGH or LOW state.
    delay(1000); // delay for 1 second
    digitalWrite(PA7,LOW); //Change the PA0 to any digital pin you want. Also, you can set this
    to HIGH or LOW state.
    delay(1000); // delay for 1 second
}
```

LoraWan - Geocaching

Het doel van dit project is om een geocache te ontwikkelen die uit twee delen bestaat. De speler moet eerst de eerste puzzel op de eerste locatie oplossen. Wanneer deze puzzel correct is opgelost, wordt via een LoRaWAN-verbinding de tweede puzzel ontgrendeld. Deze tweede puzzel bevindt zich op een andere locatie.

LoRaWAN wordt gebruikt om de verschillende onderdelen van de geocache met elkaar en met de centrale besturing te laten communiceren. Hierdoor kan de status van de eerste puzzel op afstand worden doorgegeven en kan de tweede puzzel pas worden geactiveerd wanneer aan de juiste voorwaarden is voldaan.

De onderstaande code is momenteel alleen bedoeld als testcode om bepaalde functionaliteiten en de communicatie via LoRaWAN uit te proberen. De uiteindelijke implementatie moet nog verder worden uitgewerkt, uitgebreid en ontwikkeld.

```
#include <Arduino.h>
#include <TinyGPSPlus.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

// =====
// OLED CONFIGURATIE
// =====

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET -1
#define OLED_ADDRESS 0x3C

// Standaard I2C pinnen ESP32
#define I2C_SDA 21
#define I2C_SCL 22

Adafruit_SSD1306 display(
    SCREEN_WIDTH,
    SCREEN_HEIGHT,
```

```

    &Wire,
    OLED_RESET
);

// =====
// PIN CONFIGURATIE
// =====

// GPS - UART2
#define GPS_RX_PIN 16      // ESP32 RX <- GPS TX
#define GPS_TX_PIN 17      // ESP32 TX -> GPS RX

// RAK3172 - UART1
#define RAK_RX_PIN 26      // ESP32 RX <- RAK TX
#define RAK_TX_PIN 27      // ESP32 TX -> RAK RX

// UART baudrates
#define GPS_BAUD 9600
#define RAK_BAUD 115200

// =====
// GPS DETECTIE
// =====

// Hoe lang we bij het opstarten naar GPS-data zoeken
#define GPS_DETECT_TIMEOUT 10000

// GPS object
TinyGPSPlus gps;

// Hardware UARTs
HardwareSerial GPS_Serial(2);
HardwareSerial RAK_Serial(1);

// =====
// LORAWAN INSTELLINGEN
// =====

```

```

// OTAA AppEUI
String APP_EUI = "##";

// OTAA AppKey
String APP_KEY = "##";

// DevEUI
String DEV_EUI = "##";


// =====
// STATUS
// =====

bool lorawanJoined = false;

// Geeft aan of er daadwerkelijk een GPS-module gevonden is
bool gpsModuleFound = false;

unsigned long lastSendTime = 0;

// Iedere 60 seconden
const unsigned long SEND_INTERVAL = 60000;


// =====
// OLED FUNCTIES
// =====

void oledClear()
{
    display.clearDisplay();
    display.setTextColor(SSD1306_WHITE);
    display.setTextSize(1);
    display.setCursor(0, 0);
}

void oledShow()
{
    display.display();

```

```

}

void oledMessage(String line1)
{
    oledClear();

    display.println(line1);

    oledShow();
}

void oledMessage(String line1, String line2)
{
    oledClear();

    display.println(line1);
    display.println();
    display.println(line2);

    oledShow();
}

// =====
// GEOCACHING OPSTARTANIMATIE
// =====

void oledStartup()
{
    oledClear();

    // -----
    // Fase 1: losse blokjes laten verschijnen
    // -----

    // Midden van het scherm
    const int cx = 64;
    const int cy = 25;

```

```

// Blokjes die langzaam richting logo bewegen
int blocks[][2] = {
    {10, 10}, {118, 12}, {20, 48}, {108, 50},
    {35, 5}, {92, 7}, {5, 32}, {123, 35},
    {28, 22}, {100, 25}, {18, 58}, {110, 58}
};

for (int i = 0; i < 12; i++)
{
    display.fillRect(blocks[i][0], blocks[i][1], 4, 4,
                      SSD1306_WHITE);

    oledShow();
    delay(90);
}

// -----
// Fase 2: blokjes rond het logo
// -----

oledClear();

for (int i = 0; i < 8; i++)
{
    display.fillRect(30 + i * 9, 18, 5, 5,
                      SSD1306_WHITE);
    display.fillRect(30 + i * 9, 38, 5, 5,
                      SSD1306_WHITE);

    oledShow();
    delay(70);
}

// -----
// Fase 3: cache / symbool opbouwen
// -----

oledClear();

// Vereenvoudigd cache-symbool
// geschikt voor 128x64 monochroom OLED

```

```
display.drawCircle(64, 25, 15, SSD1306_WHITE);
oledShow();
delay(120);

display.fillCircle(64, 25, 8, SSD1306_WHITE);
oledShow();
delay(120);

// Uitsparing in het midden
display.fillRect(61, 18, 6, 14, SSD1306_BLACK);
oledShow();
delay(150);

// -----
// Fase 4: tekst verschijnt
// -----

display.setTextSize(1);
display.setTextColor(SSD1306_WHITE);

display.setCursor(35, 46);
display.print("GEOCACHING");

oledShow();
delay(3000);

// -----
// Fase 5: korte volledige logo-weergave
// -----

delay(700);

// Fade-achtig effect door het logo enkele keren
// opnieuw te tekenen
for (int i = 0; i < 2; i++)
{
    display.invertDisplay(true);
    delay(100);

    display.invertDisplay(false);
```

```

        delay(100);
    }

    delay(500);

    // -----
    // Daarna normale startup
    // -----

    oledClear();

    display.setTextSize(1);
    display.println("Geocache - xXxXxX");
    display.println();
    display.println("Systeem starten...");
    display.println();
    display.println("Even geduld");

    oledShow();

    delay(1500);
}

```

```

void oledGPSStatus()
{
    oledClear();

    display.println("GPS controleren...");
    display.println();

    if (gpsModuleFound)
    {
        display.println("NEO-8M gevonden");

        if (gps.location.isValid())
        {
            display.println();
            display.print("LAT: ");
            display.println(gps.location.lat(), 6);
        }
    }
}

```

```
        display.print("LON: ");
        display.println(gps.location.lng(), 6);
    }
    else
    {
        display.println();
        display.println("Wachten op fix...");
    }
}
else
{
    display.println("GPS NIET gevonden");
}

oledShow();
}
```

```
void oledRAKStatus(String status)
{
    oledClear();

    display.println("RAK3172");
    display.println();

    display.println(status);

    oledShow();
}
```

```
void oledJoinStatus(String status)
{
    oledClear();

    display.println("LoRaWAN");
    display.println();

    display.println(status);
}
```

```

oledShow();
}

void oledGPSPosition()
{
    oledClear();

    display.setTextSize(1);

    if (gps.location.isValid())
    {
        display.println("GPS POSITIE");
        display.println();

        display.print("LAT: ");
        display.println(gps.location.lat(), 6);

        display.print("LON: ");
        display.println(gps.location.lng(), 6);

        if (gps.satellites.isValid())
        {
            display.println();
            display.print("SAT: ");
            display.println(gps.satellites.value());
        }
    }
    else
    {
        display.println("GPS POSITIE");
        display.println();
        display.println("Geen geldige fix");
    }

    oledShow();
}

void oledSending()
{

```

```
oledClear();

display.println("TTN VERZENDEN");
display.println();

if (gps.location.isValid())
{
    display.print("LAT: ");
    display.println(gps.location.lat(), 6);

    display.print("LON: ");
    display.println(gps.location.lng(), 6);
}

display.println();
display.println("Bericht wordt");
display.println("verzonden...");

oledShow();
}
```

```
void oledSent()
{
    oledClear();

    display.println("TTN VERZONDEN");
    display.println();

    display.println("GPS data");
    display.println("naar TTN gestuurd.");

    oledShow();
}
```

```
void oledMainStatus()
{
    oledClear();

    display.println("GPS + LoRaWAN");
```

```

    if (lorawanJoined)
    {
        display.println("LoRa: JOINED");
    }
    else
    {
        display.println("LoRa: NIET JOINED");
    }

    display.println();

    if (gps.location.isValid())
    {
        display.print("LAT ");
        display.println(gps.location.lat(), 6);

        display.print("LON ");
        display.println(gps.location.lng(), 6);
    }
    else
    {
        display.println("GPS: geen fix");
    }

    oledShow();
}

// =====
// FUNCTIE: stuur AT commando
// =====

void sendRAKCommand(String command)
{
    Serial.print("RAK >> ");
    Serial.println(command);

    RAK_Serial.println(command);
}

```

```

// =====
// FUNCTIE: lees RAK3172 UART
// =====

void readRAK()
{
    while (RAK_Serial.available())
    {
        String response = RAK_Serial.readStringUntil('\n');

        response.trim();

        if (response.length() == 0)
            continue;

        Serial.print("RAK << ");
        Serial.println(response);

        // -----
        // JOIN SUCCESS
        // -----

        if (response.indexOf("+EVT:JOINED") >= 0)
        {
            Serial.println();
            Serial.println("=====");
            Serial.println(" LoRaWAN JOIN SUCCESS!");
            Serial.println("=====");
            Serial.println();

            lorawanJoined = true;

            oledJoinStatus("JOINED!");

            delay(1500);

            // Eerste GPS-bericht direct na JOIN
            lastSendTime = millis() - SEND_INTERVAL;
        }
    }
}

```

```

// -----
// JOIN FAILED
// -----

if (response.indexOf("+EVT:JOIN FAILED") >= 0)
{
    Serial.println("LoRaWAN JOIN FAILED!");

    lorawanJoined = false;

    oledJoinStatus("JOIN FAILED");

    delay(1500);
}

// -----
// SEND SUCCESS
// -----

if (response.indexOf("+EVT:SEND_CONFIRMED_OK") >= 0)
{
    Serial.println("TTN bericht bevestigd.");

    oledSent();
}

// -----
// SEND SUCCESS / UNCONFIRMED
// -----

if (response.indexOf("+EVT:TX_DONE") >= 0)
{
    Serial.println("LoRaWAN verzending voltooid.");

    oledSent();
}
}

```

```

}

// =====
// FUNCTIE: lees GPS
// =====

void readGPS()
{
    while (GPS_Serial.available())
    {
        char c = GPS_Serial.read();

        gps.encode(c);

        // Als TinyGPSPlus daadwerkelijk geldige GPS-data
        // heeft ontvangen, weten we dat er een module
        // aanwezig is en data verstuurt.
        if (gps.passedChecksum() > 0)
        {
            if (!gpsModuleFound)
            {
                gpsModuleFound = true;

                Serial.println();
                Serial.println("=====");
                Serial.println(" GPS MODULE GEVONDEN!");
                Serial.println(" NEO-8M NMEA data ontvangen.");
                Serial.println("=====");
                Serial.println();

                oledGPSStatus();
            }
        }
    }
}

// =====
// FUNCTIE: controleer of GPS-module aanwezig is
// =====

```

```

bool detectGPSTModule()
{
    Serial.println();
    Serial.println("=====");
    Serial.println(" GPS MODULE CONTROLLEREN");
    Serial.println("=====");
    Serial.println();

    Serial.println("Wachten op NEO-8M GPS data...");

    oledClear();
    display.println("GPS MODULE");
    display.println();
    display.println("Controlleren...");
    display.println();
    display.println("Wachten op NEO-8M");
    display.println("NMEA data...");
    oledShow();

    unsigned long startTime = millis();

    while (millis() - startTime < GPS_DETECT_TIMEOUT)
    {
        // GPS UART uitlezen
        readGPS();

        // RAK UART ondertussen ook blijven uitlezen
        readRAK();

        if (gpsModuleFound)
        {
            Serial.println();
            Serial.println("GPS controle succesvol.");
            Serial.println("NEO-8M module reageert.");

            if (gps.satellites.isValid())
            {
                Serial.print("Satellieten zichtbaar: ");
                Serial.println(gps.satellites.value());
            }
        }
    }
}

```

```

    }

    Serial.println();

    oledClear();
    display.println("GPS MODULE");
    display.println();
    display.println("NEO-8M GEVONDEN!");

    if (gps.satellites.isValid())
    {
        display.println();
        display.print("Satellieten: ");
        display.println(gps.satellites.value());
    }

    oledShow();

    delay(1500);

    return true;
}

delay(10);
}

// Geen geldige NMEA-data ontvangen
gpsModuleFound = false;

Serial.println();
Serial.println("=====");
Serial.println(" GPS MODULE NIET GEVONDEN!");
Serial.println("=====");
Serial.println();

Serial.println("Controleer:");
Serial.println("- Voeding van de NEO-8M");
Serial.println("- GPS TX -> ESP32 RX");
Serial.println("- GPS RX -> ESP32 TX");
Serial.println("- GND");

```

```

Serial.println("- Baudrate (9600)");
Serial.println();

oledClear();
display.println("GPS MODULE");
display.println();
display.println("NIET GEVONDEN!");
display.println();
display.println("Controleer:");
display.println("TX/RX/GND/voeding");
display.println("Baudrate 9600");
oledShow();

delay(2000);

return false;
}

// =====
// FUNCTIE: GPS positie tonen
// =====

void printGPS()
{
    Serial.println();
    Serial.println("===== GPS =====");

    if (!gpsModuleFound)
    {
        Serial.println("GPS module niet gevonden.");
        Serial.println("=====");
        Serial.println();

        oledMessage("GPS module niet", "gevonden.");

        return;
    }

    if (gps.location.isValid())

```

```
{
    Serial.print("Latitude : ");
    Serial.println(gps.location.lat(), 6);

    Serial.print("Longitude: ");
    Serial.println(gps.location.lng(), 6);

    oledGPSPosition();
}
else
{
    Serial.println("GPS module gevonden.");
    Serial.println("GPS positie nog niet beschikbaar.");
    Serial.println("Wachten op satellietfix...");

    oledClear();

    display.println("GPS GEVONDEN");
    display.println();
    display.println("Geen GPS fix");
    display.println();
    display.println("Wachten op");
    display.println("satellieten...");

    oledShow();
}

if (gps.satellites.isValid())
{
    Serial.print("Satellites: ");
    Serial.println(gps.satellites.value());
}

if (gps.hdop.isValid())
{
    Serial.print("HDOP      : ");
    Serial.println(gps.hdop.hdop());
}
```

```

    Serial.println("=====");
    Serial.println();
}

// =====
// FUNCTIE: GPS positie naar HEX payload
// =====
//
// Byte 0-3 = latitude
// Byte 4-7 = longitude
//
// Latitude en longitude worden vermenigvuldigd met
// 1.000.000 en als signed 32-bit integer verstuurd.
// =====

String createGPSPayload()
{
    if (!gpsModuleFound)
    {
        return "";
    }

    if (!gps.location.isValid())
    {
        return "";
    }

    int32_t latitude =
        (int32_t)(gps.location.lat() * 1000000);

    int32_t longitude =
        (int32_t)(gps.location.lng() * 1000000);

    uint8_t payload[8];

    // Latitude
    payload[0] = (latitude >> 24) & 0xFF;
    payload[1] = (latitude >> 16) & 0xFF;
    payload[2] = (latitude >> 8) & 0xFF;
    payload[3] = latitude & 0xFF;

```

```

// Longitude
payload[4] = (longitude >> 24) & 0xFF;
payload[5] = (longitude >> 16) & 0xFF;
payload[6] = (longitude >> 8) & 0xFF;
payload[7] = longitude & 0xFF;

// Omzetten naar HEX string
char hexPayload[17];

for (int i = 0; i < 8; i++)
{
    sprintf(&hexPayload[i * 2], "%02X", payload[i]);
}

hexPayload[16] = '\0';

return String(hexPayload);
}

// =====
// FUNCTIE: GPS naar TTN sturen
// =====

void sendGPS()
{
    Serial.println();
    Serial.println("=====");
    Serial.println(" GPS DATA VERZENDEN");
    Serial.println("=====");

    // Eerst controleren of GPS-module aanwezig is
    if (!gpsModuleFound)
    {
        Serial.println("GPS module niet gevonden.");
        Serial.println("Bericht wordt niet verzonden.");

        oledMessage(
            "TTN NIET VERZONDEN",

```

```

        "GPS niet gevonden"
    );

    return;
}

// Daarna controleren of er een geldige positie is
if (!gps.location.isValid())
{
    Serial.println("GPS module is gevonden.");
    Serial.println("Maar er is nog geen geldige GPS positie.");
    Serial.println("Bericht wordt niet verzonden.");

    oledMessage(
        "TTN NIET VERZONDEN",
        "Geen GPS fix"
    );

    return;
}

double latitude = gps.location.lat();
double longitude = gps.location.lng();

Serial.print("Latitude : ");
Serial.println(latitude, 6);

Serial.print("Longitude: ");
Serial.println(longitude, 6);

String payload = createGPSPayload();

if (payload.length() == 0)
{
    Serial.println("GPS payload is leeg.");
    Serial.println("Bericht wordt niet verzonden.");
}

```

```

        oledMessage(
            "TTN NIET VERZONDEN",
            "Payload leeg"
        );

        return;
    }

    Serial.print("Payload  : ");
    Serial.println(payload);

    // OLED: aangeven dat TTN-bericht wordt verstuurd
    oledSending();

    // RAK3172:
    // AT+SEND=<port>:<hex payload>
    String command = "AT+SEND=1:" + payload;

    sendRAKCommand(command);

    Serial.println("GPS data naar LoRaWAN gestuurd.");
    Serial.println();

    // Meteen tonen dat opdracht naar RAK is gestuurd
    oledSent();
}

// =====
// SETUP
// =====

void setup()
{
    // -----
    // USB Serial
    // -----

```

```
Serial.begin(115200);

delay(1000);

// -----
// OLED STARTEN
// -----

Wire.begin(I2C_SDA, I2C_SCL);

if (!display.begin(
    SSD1306_SWITCHCAPVCC,
    OLED_ADDRESS))
{
    Serial.println("OLED NIET GEVONDEN!");

    // Zonder OLED verdergaan
}
else
{
    oledStartup();

    delay(2000);
}

// -----
// STARTMELDING
// -----

Serial.println();
Serial.println("=====");
Serial.println(" ESP32 GPS + RAK3172 LoRaWAN");
Serial.println("=====");
Serial.println();

oledClear();
display.println("ESP32 GPS + LoRaWAN");
```

```
display.println();
display.println("Systeem starten...");
oledShow();

delay(1000);

// -----
// GPS UART
// -----

GPS_Serial.begin(
    GPS_BAUD,
    SERIAL_8N1,
    GPS_RX_PIN,
    GPS_TX_PIN
);

Serial.println("GPS UART gestart.");

oledClear();
display.println("GPS UART");
display.println();
display.println("Gestart");
display.print("Baud: ");
display.println(GPS_BAUD);
oledShow();

delay(1000);

// -----
// RAK UART
// -----

RAK_Serial.begin(
    RAK_BAUD,
    SERIAL_8N1,
    RAK_RX_PIN,
    RAK_TX_PIN
);
```

```
Serial.println("RAK3172 UART gestart.");

oledClear();
display.println("RAK3172 UART");
display.println();
display.println("Gestart");
display.print("Baud: ");
display.println(RAK_BAUD);
oledShow();

delay(1000);

// -----
// GPS MODULE CONTROLEREN
// -----

if (!detectGPSPModule())
{
    Serial.println();
    Serial.println("WAARSCHUWING:");
    Serial.println("Geen NEO-8M GPS module gevonden.");
    Serial.println("Het programma gaat wel verder.");
    Serial.println("LoRaWAN kan dus nog steeds worden");
    Serial.println("opgestart.");
    Serial.println();

    oledClear();
    display.println("WAARSCHUWING");
    display.println();
    display.println("Geen GPS gevonden");
    display.println();
    display.println("LoRaWAN wordt");
    display.println("wel gestart.");

    oledShow();

    delay(2000);
}
```

```
// -----  
// RAK3172 testen  
// -----  
  
Serial.println();  
Serial.println("RAK3172 testen...");  
  
oledMessage(  
    "RAK3172 testen...",  
    "AT commando"  
);  
  
sendRAKCommand("AT");  
  
delay(1000);  
  
readRAK();  
  
// -----  
// RAK3172 resetten  
// -----  
  
Serial.println();  
Serial.println("RAK3172 reset...");  
  
oledMessage(  
    "RAK3172 reset",  
    "Even wachten..."  
);  
  
sendRAKCommand("AT+RESET");  
  
delay(3000);  
  
readRAK();  
  
// -----  
// LoRaWAN configureren
```

```

// -----

Serial.println();
Serial.println("LoRaWAN configureren...");

oledClear();
display.println("LoRaWAN");
display.println();
display.println("Configureren...");
display.println();
display.println("OTAA");
oledShow();

delay(1000);

// OTAA
sendRAKCommand("AT+NJM=1");

delay(500);

// AppEUI
sendRAKCommand("AT+APPEUI=" + APP_EUI);

delay(500);

// AppKey
sendRAKCommand("AT+APPKEY=" + APP_KEY);

delay(500);

// -----
// JOIN
// -----

Serial.println();
Serial.println("=====");
Serial.println(" LoRaWAN JOIN wordt gestart...");

```

```

Serial.println(" Wachten op +EVT:JOINED");
Serial.println("=====");
Serial.println();

oledClear();
display.println("LoRaWAN JOIN");
display.println();
display.println("JOIN wordt");
display.println("gestart...");
display.println();
display.println("Wachten...");
oledShow();

sendRAKCommand("AT+JOIN");
}

// =====
// LOOP
// =====

void loop()
{
    // -----
    // GPS altijd uitlezen
    // -----

    readGPS();

    // -----
    // RAK3172 altijd uitlezen
    // -----

    readRAK();

    // -----
    // Alleen data sturen als LoRaWAN verbonden is

```

```

// -----

if (lorawanJoined)
{
    if (millis() - lastSendTime >= SEND_INTERVAL)
    {
        lastSendTime = millis();

        printGPS();

        sendGPS();
    }
    else
    {
        // GPS positie continu beschikbaar houden
        // op het display.
        static unsigned long lastOLEDUpdate = 0;

        if (millis() - lastOLEDUpdate >= 1000)
        {
            lastOLEDUpdate = millis();

            oledMainStatus();
        }
    }
}

// Kleine pauze
delay(10);
}

```