**Interfacing Soil NPK Sensor with Arduino**

## Component: JXCT Soil NPK sensor

The soil NPK sensor can detect the levels of nitrogen, phosphorus, and potassium in the soil (not in water. , the NPK sensor cannot be used directly with an Arduino. To communicate with Arduino, you’ll need an RS-485 transceiver module that converts a UART serial stream to RS-485, such as the one shown below.

A circuit board with wires

Description automatically generated

Code:

import serial

import time

# Define the RE and DE pins

RE\_PIN = 8

DE\_PIN = 7

# Modbus RTU requests for reading NPK values

nitro = [0x01, 0x03, 0x00, 0x1e, 0x00, 0x01, 0xe4, 0x0c]

phos = [0x01, 0x03, 0x00, 0x1f, 0x00, 0x01, 0xb5, 0xcc]

pota = [0x01, 0x03, 0x00, 0x20, 0x00, 0x01, 0x85, 0xc0]

# Initialize the serial connection

ser = serial.Serial('COM3', 9600) # Change 'COM3' to your Arduino's port

def setup():

# Define pin modes for RE and DE

ser.write(b'RE,OUTPUT\r')

ser.write(b'DE,OUTPUT\r')

time.sleep(0.5)

def nitrogen():

ser.write(b'DE,HIGH\r')

ser.write(b'RE,HIGH\r')

time.sleep(0.01)

if ser.write(bytearray(nitro))==8:

ser.write(b'DE,LOW\r')

ser.write(b'RE,LOW\r')

values = []

for i in range(7):

values.append(ser.read())

return values[4]

def phosphorous():

ser.write(b'DE,HIGH\r')

ser.write(b'RE,HIGH\r')

time.sleep(0.01)

if ser.write(bytearray(phos))==8:

ser.write(b'DE,LOW\r')

ser.write(b'RE,LOW\r')

values = []

for i in range(7):

values.append(ser.read())

return values[4]

def potassium():

ser.write(b'DE,HIGH\r')

ser.write(b'RE,HIGH\r')

time.sleep(0.01)

if ser.write(bytearray(pota))==8:

ser.write(b'DE,LOW\r')

ser.write(b'RE,LOW\r')

values = []

for i in range(7):

values.append(ser.read())

return values[4]

def loop():

while True:

val1 = nitrogen()

time.sleep(0.25)

val2 = phosphorous()

time.sleep(0.25)

val3 = potassium()

time.sleep(0.25)

print("Nitrogen: {} mg/kg".format(val1))

print("Phosphorous: {} mg/kg".format(val2))

print("Potassium: {} mg/kg".format(val3))

time.sleep(2)

if \_\_name\_\_ == "\_\_main\_\_":

setup()

loop()

# Soil Moisture Sensor

# import serial

# import time

# # Sensor pins

# SENSOR\_POWER\_PIN = 7

# SENSOR\_PIN = 8

# # Initialize the serial connection

# ser = serial.Serial('COM3', 9600) # Change 'COM3' to your Arduino's port

# def setup():

# # Set the sensorPower pin as OUTPUT

# ser.write(b'sensorPower,OUTPUT\r')

# time.sleep(0.5)

# def read\_sensor():

# # Turn the sensor ON

# ser.write(b'sensorPower,HIGH\r')

# time.sleep(0.01) # Allow power to settle

# # Read the analog value from the sensor

# ser.write(b'sensorPin,READ\r')

# val = int(ser.readline())

# # Turn the sensor OFF

# ser.write(b'sensorPower,LOW\r')

# return val

# def loop():

# while True:

# val = read\_sensor()

# print("Digital Output:", val)

# # Determine the status of soil moisture

# if val:

# print("Status: Soil is too dry - time to water!")

# else:

# print("Status: Soil moisture is perfect")

# time.sleep(1) # Take a reading every second for testing

# if \_\_name\_\_ == "\_\_main\_\_":

# setup()

# loop()

A blue circuit board with wires connected to it

Description automatically generated

# DHT11 and DHT22 temperature and humidity Sensors with Arduino

# import Adafruit\_DHT

# import time

# # Define sensor type and pin

# sensor = Adafruit\_DHT.DHT22

# data\_pin = 8

# def setup():

# pass # No setup needed

# def loop():

# while True:

# humidity, temperature = Adafruit\_DHT.read\_retry(sensor, data\_pin)

# if humidity is not None and temperature is not None:

# print("Temperature = {:.1f} C | {:.1f} F".format(temperature, temperature \* 9.0 / 5.0 + 32.0))

# print("Humidity = {:.1f} %".format(humidity))

# print("")

# else:

# print("Failed to retrieve data from sensor")

# time.sleep(2) # Delay for 2 seconds

# if \_\_name\_\_ == "\_\_main\_\_":

# setup()

# loop()

A circuit board with wires

Description automatically generated

A circuit board with wires

Description automatically generated