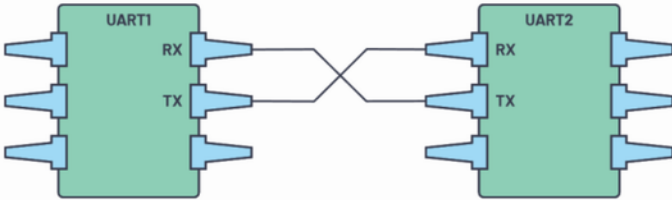


Dasar Teori I2C

UART, atau pemancar-penerima asinkronus universal, adalah salah satu protokol komunikasi perangkat-ke-perangkat yang paling banyak digunakan. Asinkronus berarti tidak ada sinyal clock untuk menyinkronkan bit-bit keluaran dari perangkat pengirim ke pihak penerima.

Ketika dikonfigurasi dengan benar, UART dapat bekerja dengan berbagai jenis protokol serial yang melibatkan transmisi dan penerimaan data serial. Dalam komunikasi serial, data ditransfer sedikit demi sedikit menggunakan satu saluran atau kabel. Dalam komunikasi dua arah, digunakan dua kabel agar transfer data serial berhasil. Tergantung pada aplikasi dan persyaratan sistem, komunikasi serial memerlukan lebih sedikit sirkuit dan kabel, sehingga mengurangi biaya implementasi.

Embedded system, mikrokontroler, dan komputer sebagian besar menggunakan UART sebagai bentuk protokol komunikasi perangkat keras antar perangkat. Di antara protokol komunikasi yang tersedia, UART hanya menggunakan dua kabel untuk transmisi dan penerimaannya, Gambar 1 menunjukkan hal tersebut



**Gambar 1. Kabel pada komunikasi UART**

Dua sinyal dari setiap perangkat UART diberi nama:

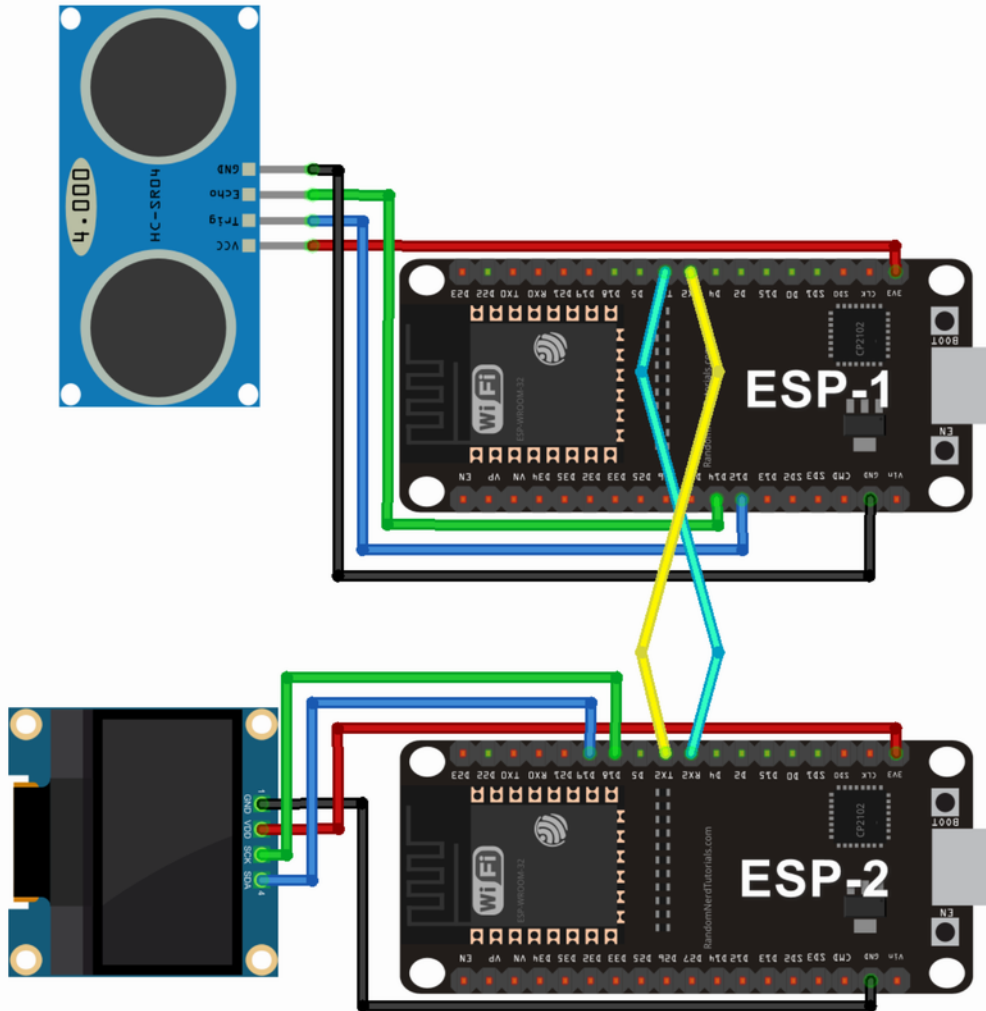
- Pemancar (Tx)
- Penerima (Rx)

Tujuan utama saluran pemancar dan penerima untuk setiap perangkat adalah untuk mengirim dan menerima data serial yang dimaksudkan untuk komunikasi serial.

Pada ESP32, secara default terdapat tiga buah saluran komunikasi UART, yaitu UART 0, UART 1, dan UART 2. Default Pin pada ESP32 ditunjukkan oleh tabel 1.

**Tabel 1. Pin UART pada ESP32**

UART Port	Rx	Tx	Useable
UART0	GPIO3	GPIO1	Yes
UART1	GPIO9	GPIO10	Yes but requires the reassignment of pins
UART2	GPIO16	GPIO17	Yes



1. Tuliskan kode program berikut pada file **main.py** dan simpan kedalam **ESP32-1** :

```
from machine import Pin, UART
from utime import sleep, sleep_us, ticks_us, ticks_diff

trig = Pin(12, Pin.OUT)
echo = Pin(14, Pin.IN)

#UART ke-2 pada kecepatan 9600 bit/detik
uart = UART(2, baudrate = 9600)

while True:
    sleep_us(5)
    trig.value(1)
    sleep_us(10)
    trig.value(0)

    while echo.value() == False:
        pass
    timerStart = ticks_us()

    while echo.value() == True:
        pass
    timerEnd = ticks_us()

    duration = ticks_diff(timerEnd, timerStart)
    dist = (str(duration*0.017))[5:]
    print("jarak benda = {} cm".format(dist))
    uart.write(dist)
    sleep(1)
```

2. Download library oled SSD1306 (lihat link di halaman terakhir)
3. Tuliskan kode program berikut pada kedalam **ESP32-2** dan simpan dengan nama **main.py**

```
from machine import UART, SoftI2C, Pin
from ssd1306 import SSD1306_I2C
from time import sleep

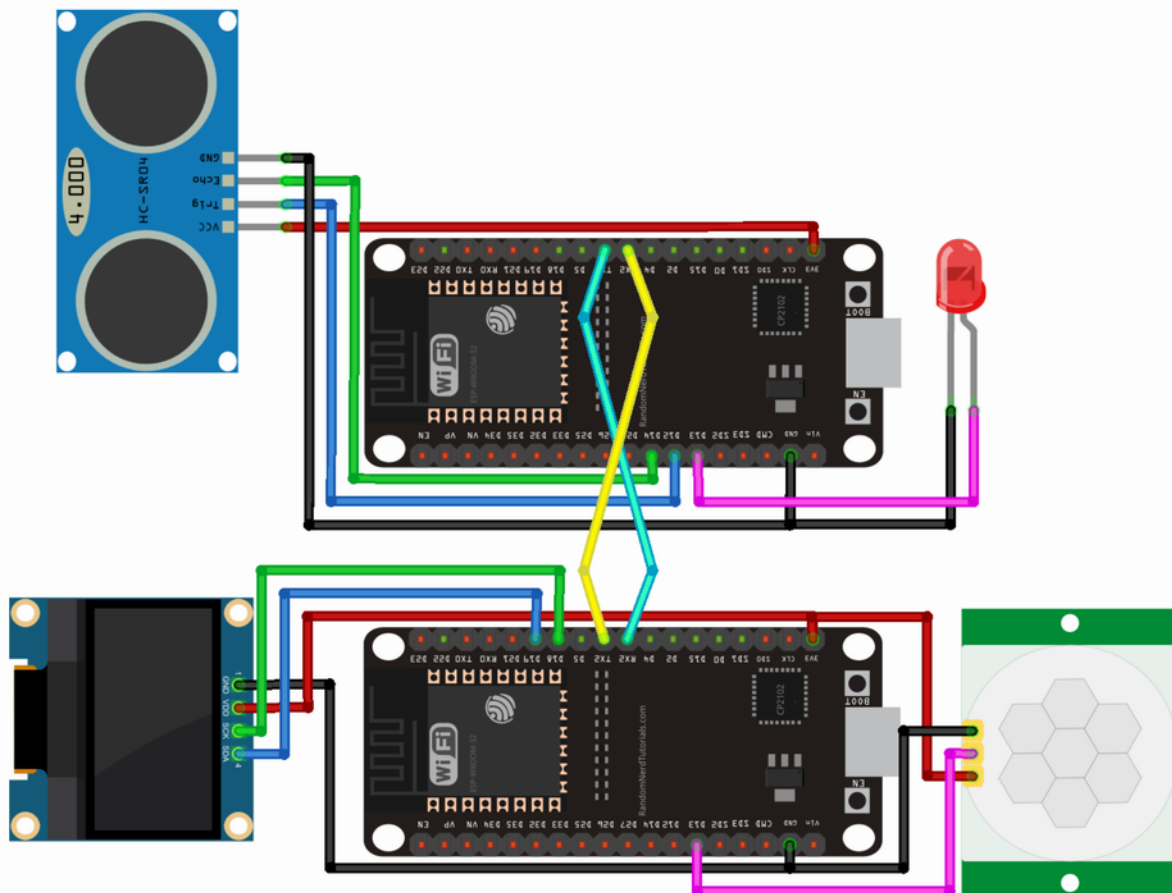
uart = UART(2,9600)
i2c = SoftI2C(scl=Pin(18), sda=Pin(19))

oled_width = 128
oled_height = 32
oled = SSD1306_I2C(oled_width, oled_height, i2c)
oled.fill(False)
oled.text("I2C Comm",0,0)
oled.show()

while True:
    text = ""
    if uart.any()>0:
        text = uart.read().decode('utf-8')
        oled.fill(False)
        oled.text("Jarak = {} cm".format(text),0,0)
        oled.show()
        sleep(1)
```

4. Jalankan kedua buah program tersebut pada kedua ESP tersebut

#### Praktik 2 (HC-SR04, HC-SR501, LED, OLED 1306, ESP32-1 dan ESP32-2)



Pada praktik 2, tambahkan **LED** pada **ESP32-1** dan **HC-SR501** pada **ESP32-2**. Tambahkan fungsi bila HC-SR501 mendeteksi adanya gerakan maka LED pada ESP32-1 akan menyala selama 3 detik, tanpa mengurangi fungsionalitas pada praktik ke-1

### Link download modul/library :

1. Download library oled dari link berikut ini dan masukkan kedalam Micropython dengan nama **ssd1306.py**  
<https://raw.githubusercontent.com/RuiSantosdotme/ESP-MicroPython/master/code/Others/OLED/ssd1306.py>