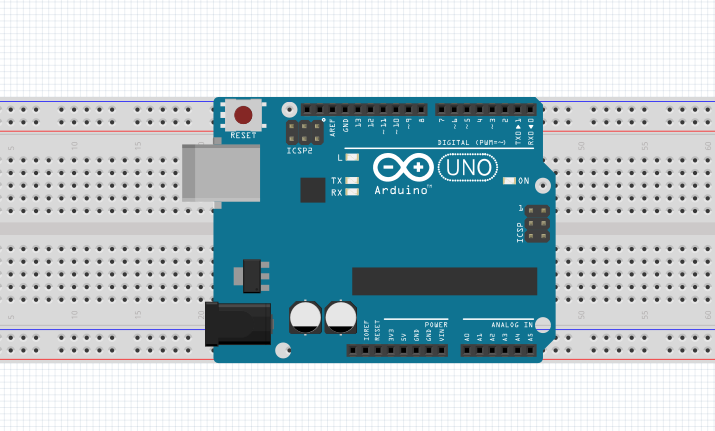
|  |  |  |  |
| --- | --- | --- | --- |
| **KELOMPOK** | C | **ACC** | |
| **Tanggal Praktikum** | 4 Mei 2018 |  |  |
|  | Nugroho Nanda Styanto |
|  | Kharisma Aji Satria Tama |
|  | Puji Hanung Winarko | **Tanggal ACC :** | |
|  | Ahmad Sujarwo | **Revisi Tanggal :** | |

**MODUL 2**

Projects Semaphore dalam FreeRTOS

1. **TUJUAN**
2. Praktikan dapat memahami dan mengetahui fungsi Semaphore dalam FreeRTOS.
3. Praktikan dapat membuat program dengan fungsi Semaphore dalam FreeRTOS.
4. Praktikan dapat membuat program dengan 3 task dengan menggunakan fungsi Semaphore dalam FreeRTOS.
5. **ALAT DAN BAHAN**
6. Laptop / PC yang telah diinstal software Arduino
7. Arduino Uno
8. **HASIL PRAKTIKUM**

**C 1.1 Skema Ragkaian**

**C 1.2 Script**

#include<FreeRTOS\_AVR.h>

SemaphoreHandle\_t sem;

static void saklar1(void\* arg){

while (1){

if (xSemaphoreTake (sem, portMAX\_DELAY)){

xSemaphoreGive(sem);

Serial.print("saklar satu ");

bus();

Serial.println("\t\tmotor 1 menyala");

}

}

}

static void saklar2(void\* arg){

while (1){

if (xSemaphoreTake (sem, portMAX\_DELAY)){

xSemaphoreGive(sem);

Serial.print("saklar dua ");

bus();

Serial.println("\t\tmoror 2 menyala");

}

}

}

static void saklar3(void\* arg){

while (1){

if (xSemaphoreTake (sem, portMAX\_DELAY)){

xSemaphoreGive(sem);

Serial.print("saklar tiga ");

bus();

Serial.println("\t\tmotor 3 menyala");

}

}

}

void bus(){

Serial.println("terhubung");

vTaskDelay(500);

}

void setup()

{

portBASE\_TYPE s1,s2,s3;

Serial.begin(9600);

sem = xSemaphoreCreateCounting(1,0);

s1= xTaskCreate(saklar1, NULL, configMINIMAL\_STACK\_SIZE, NULL, 3,NULL);

s2= xTaskCreate(saklar2, NULL, configMINIMAL\_STACK\_SIZE, NULL, 2,NULL);

s3= xTaskCreate(saklar3, NULL, configMINIMAL\_STACK\_SIZE, NULL, 1,NULL);

if (sem==NULL ||s1 !=pdPASS || s2 !=pdPASS || s3 !=pdPASS)

{

Serial.println(F("Creation Problem"));

while(1);

}

xSemaphoreGive(sem);

vTaskStartScheduler();

Serial.println(F("Insufficient RAM"));

while(1);

}

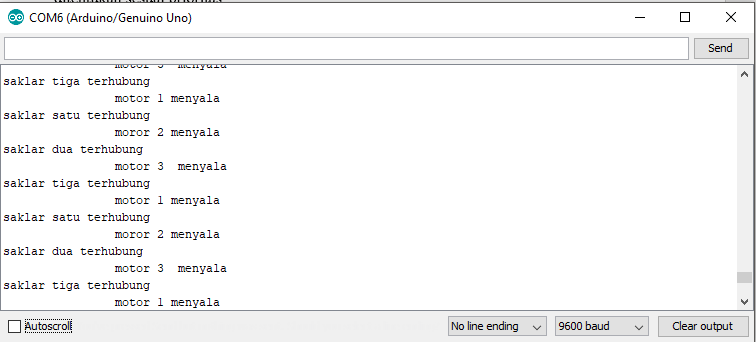
void loop()

{

// put your main code here, to run repeatedly:

}

**C 1.3 Serial Monitor**



1. **ANALISA**

Analisa Percobaan

Pada praktikum 2 ini menggunakan semaphore dan 3 task pada program. Program dibuat dapat berjalan secara terus menerus secara gantian pada masing- masing task nya. xSemaphoreGive digunakan untuk memasukkan data kemudian akan diambil oleh xSemaphoreTake dan kemudian ditampilkan pada Serial Monitor. xTaskCreate membuat set up perintah berjalan dengan prioritas yang diinginkan,semakin besar nilainya maka akan dieksekusi yang pertama. void TaskDigitalRead( void \*pvParameters ) merupakan fungsi untuk mengatur bagaimana Perintah tersebut akan berjalan. setiap task program tidak menggunakan delay pada xSemaphoreGive dan XsemaphorTake karena agar dapat berjalan saling bergantian. Pada Serial.println digunakan untuk menapilkan data keserial monitor.

1. **KESIMPULAN**
2. Semaphore adalah suatu isyarat yang digunakan untuk menentukan akses untuk berbagi sumber daya sistem.
3. xSemaphoreGive dan xSemaphoreTake dalam praktikum ini saling berhubungan dalam jalannya program
4. Dari Serial Monitor terlihat hasil dari xSemaphoreGive dan xSemaphoreTake secara acak karena pengaruh vTaskDelay