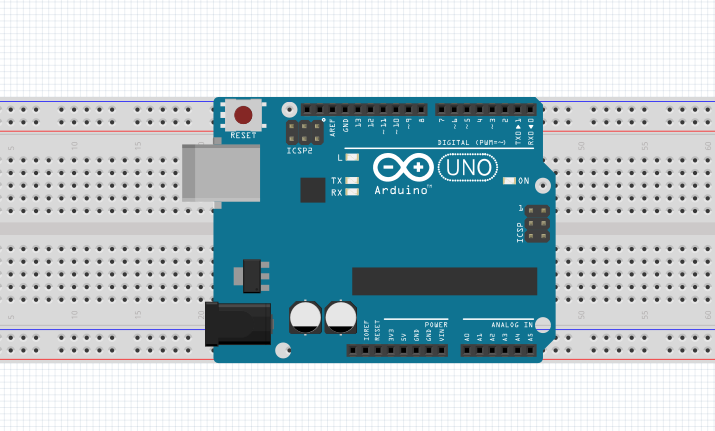
|  |  |  |  |
| --- | --- | --- | --- |
| **KELOMPOK** | C | **ACC** | |
| **Tanggal Praktikum** | 02 Juni 2018 |  |  |
|  | Nugroho Nanda Styanto |  | |
|  | Kharisma Aji Satria Tama | **Tanggal ACC :** | |
|  | Puji Hanung Winarko | **Revisi Tanggal :** | |
|  | Ahmad Sujarwo |  | |

**MODUL 3**

**Flexible and Absolute Timing On(project)**

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"

#include "basic\_io\_AVR.h"

void vContinousProcessingTask (void \*pvParameters);

void vPeriodicTask (void \*pvParameters);

const char \*pcTextForTask1 = "lampu 1 menyala\r\n";

const char \*pcTextForTask2 = "lampu 2 menyala\r\n";

const char \*pcTextForTask5 = "lampu 3 menyala\r\n";

const char \*pcTextForPeriodicTask3 = "task3 is running\r\n";

const char \*pcTextForPeriodicTask4 = "task4 is running\r\n";

void setup(void)

{

Serial.begin(9600);

xTaskCreate(vContinousProcessingTask, "Task 1", 200, (void\*)pcTextForTask1, 2, NULL);

xTaskCreate(vContinousProcessingTask, "Task 2", 200, (void\*)pcTextForTask2, 2, NULL);

xTaskCreate(vContinousProcessingTask, "Task 5", 200, (void\*)pcTextForTask5, 2, NULL);

xTaskCreate(vPeriodicTask3, "Task 3", 200, (void\*)pcTextForPeriodicTask3, 3, NULL);

xTaskCreate(vPeriodicTask4, "Task 4", 200, (void\*)pcTextForPeriodicTask4, 3, NULL);

vTaskStartScheduler();

for(;;);

}

void vContinousProcessingTask (void \*pvParameters)

{

char \*pcTaskname;

pcTaskname = (char\*)pvParameters;

for(;;)

{

vPrintString(pcTaskname);

}

vTaskDelay(500);

}

void vPeriodicTask3 (void \*pvParameters)

{

TickType\_t xLastWakeTime3;

xLastWakeTime3 = xTaskGetTickCount ();

for(;;)

{

vPrintString("sisrem 1 aktif------------------\r\n");

vTaskDelayUntil(&xLastWakeTime3, (500/portTICK\_PERIOD\_MS));

}

}

void vPeriodicTask4 (void \*pvParameters)

{

TickType\_t xLastWakeTime4;

xLastWakeTime4 = xTaskGetTickCount ();

for(;;)

{

vTaskDelayUntil(&xLastWakeTime4, (500/portTICK\_PERIOD\_MS));

vPrintString("sistem 2 aktif------------------\r\n");

vTaskDelayUntil(&xLastWakeTime4, (200/portTICK\_PERIOD\_MS));

}

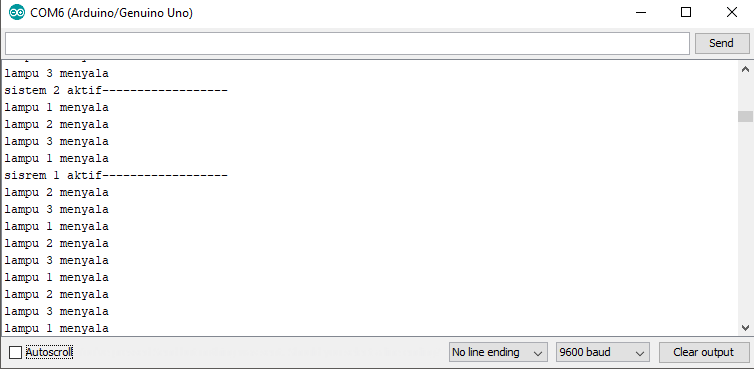
}

void loop() {

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

}

**C 1.3 Serial Monitor**



1. **ANALISA**

Analisa Percobaan

Pada praktikum task pada program. Program dibuat dapat berjalan secara terus menerus secara gantian pada masing- masing task nya. vTaskDelay() digunakan pada void vContinousProcessingTask sedangkan vTaskDelayUntil() digunakan pada void vPeriodicTask. Task 1, 2 terdapat pada void vContinousProcessingTask sedangkan Task 3 terdapat pada void vPeriodicTask3 dan Task 4 terdapat pada void vPeriodicTask4 lampu

sistem aktif baru task akan bekerja

1. **KESIMPULAN**
2. vTaskDelay di mana tugas membuka blokir relatif terhadap waktu di mana vTaskDelay() ini dipanggil.
3. vTaskDelayUntil() menentukan waktu absolut yang membuka blokir atau mengaktifkan task selanjutnya.