| **KELOMPOK** | C | **ACC** | |
| --- | --- | --- | --- |
| **Tanggal Praktikum** | 27 Maret 2018 |  |  |
|  | Nugroho Nanda Styanto |
|  | Kharisma Aji Satria Tama | **Tanggal ACC :** | |
|  | Puji Hanung Winarko | **Revisi Tanggal :** | |
|  | Ahmad Sujarwo |  | |

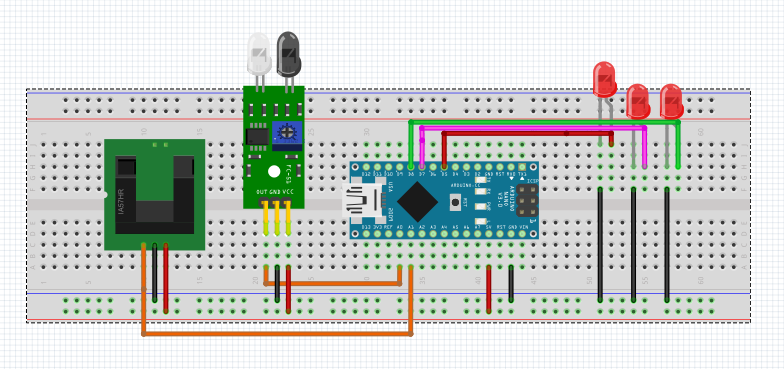
**MODUL Tugas Besar**

Pembuka pintu gerbang

1. **TUJUAN**
2. Praktikan dapat memahami salah satu sistem operasi yang ada pada Arduino yaitu FreeRTOS.
3. Praktikan dapat membuat program dari modul modul praktikum sebelumnya
4. Praktikan dapat membuat basic program dengan menggunakan sistem oprasi FreeRTOS.
5. **ALAT DAN BAHAN**
6. Laptop / PC yang telah diinstal software Arduino
7. Arduino Nano
8. LED
9. Sensor optocoupler
10. Sensor IR
11. **HASIL PRAKTIKUM**

C.1 Hasil Percobaan 1

D.1.1 Skema Rangkaian



C.1.2 Script Program

#include <Arduino\_FreeRTOS.h>

#include <semphr.h>

//SemaphoreHandle\_t xSerialSemaphore;

// define two Tasks for DigitalRead & AnalogRead

void TaskBlink( void \*pvParameters );

void TaskAnalogRead( void \*pvParameters );

void TaskAnalogRead2( void \*pvParameters );

// the setup function runs once when you press reset or power the board

void setup() {

// initialize serial communication at 9600 bits per second:

Serial.begin(9600); // perintah menampilkan nilai2 yang terbaca dengan baudrate 9600

/\*

// Semaphores are useful to stop a Task proceeding, where it should be paused to wait,

// because it is sharing a resource, such as the Serial port.

// Semaphores should only be used whilst the scheduler is running, but we can set it up here.

if ( xSerialSemaphore == NULL ) // Check to confirm that the Serial Semaphore has not already been created.

{

xSerialSemaphore = xSemaphoreCreateMutex(); // Create a mutex semaphore we will use to manage the Serial Port

if ( ( xSerialSemaphore ) != NULL )

xSemaphoreGive( ( xSerialSemaphore ) ); // Make the Serial Port available for use, by "Giving" the Semaphore.

}

\*/

// Now set up two Tasks to run independently. // pendeklarasian task yang digunakan dalam sistem dengan mengatur besar stack dan nlai

xTaskCreate( // prioritas bila besar maka akan menjadi prioritas awal...

TaskBlink

, (const portCHAR \*)"Blink" // A name just for humans

, 128 // This stack size can be checked & adjusted by reading the Stack Highwater

, NULL

, 1 // Priority, with 1 being the highest, and 4 being the lowest.

, NULL );

xTaskCreate(

TaskAnalogRead

, (const portCHAR \*) "AnalogRead"

, 128 // Stack size

, NULL

, 2 // Priority

, NULL );

xTaskCreate(

TaskAnalogRead2

, (const portCHAR \*) "AnalogRead"

, 128 // Stack size

, NULL

, 3 // Priority

, NULL );

// Now the Task scheduler, which takes over control of scheduling individual Tasks, is automatically started.

}

void loop()

{

// Empty. Things are done in Tasks.

}

/\*--------------------------------------------------\*/

/\*---------------------- Tasks ---------------------\*/

/\*--------------------------------------------------\*/

void TaskBlink(void \*pvParameters) // This is a task.

{

(void) pvParameters;

// initialize digital pin 13 as an output.

pinMode(5, OUTPUT); // deklarasi pin output 5 untuk menyalakan aktuator (led/relay/yanglainnya)

for (;;) // A Task shall never return or exit.

{

digitalWrite(5, HIGH); // eksekusi led menyala dan mati atau berkedip dengan 200 / porttick delaynya

vTaskDelay( 200 / portTICK\_PERIOD\_MS ); // wait for one second

digitalWrite(5, LOW); // turn the LED off by making the voltage LOW

vTaskDelay( 200 / portTICK\_PERIOD\_MS ); // wait for one second

}

}

void TaskAnalogRead( void \*pvParameters \_\_attribute\_\_((unused)) ) // This is a Task.

{

pinMode(7, OUTPUT);

for (;;) // perintah untuk merjalan selama program dijalankan looping

{

// read the input on analog pin 0:

int sensorValue = analogRead(A0); // pin yang digunakan untuk melihat nilai analog dari sensor obstacle / IR

//int ledPin = 7; // pin yang digunakan untuk output eksekusi aktuator bisa motor/led/relay/dll

Serial.println(sensorValue); // perintah menampilkan sensor value

/\*

// See if we can obtain or "Take" the Serial Semaphore.

// If the semaphore is not available, wait 5 ticks of the Scheduler to see if it becomes free.

if ( xSemaphoreTake( xSerialSemaphore, ( TickType\_t ) 5 ) == pdTRUE )

{

// We were able to obtain or "Take" the semaphore and can now access the shared resource.

// We want to have the Serial Port for us alone, as it takes some time to print,

// so we don't want it getting stolen during the middle of a conversion.

// print out the value you read:

Serial.println(sensorValue);

xSemaphoreGive( xSerialSemaphore ); // Now free or "Give" the Serial Port for others.

} \*/

// memberikan kemungkinan bila nilai2 analog muncul , ledhidup bila kurang dari 500 dan mati bila

if (sensorValue < 500) //lebih dari 500

{

digitalWrite(7, HIGH);

}

else (sensorValue > 500);

{

digitalWrite(7, LOW);

}

//delay(500); // one tick delay (15ms) in between reads for stability

vTaskDelay(100); // one tick delay (15ms) in between reads for stability // durasi program task ini berjalan.

}

// vTaskDelay(10); // one tick delay (15ms) in between reads for stability

}

void TaskAnalogRead2( void \*pvParameters \_\_attribute\_\_((unused)) ) // This is a Task.

{

for (;;)

{

// read the input on analog pin 0:

int sensorValue2 = analogRead(A1); // A1 sebagai pin analog read ke2 untuk membaca sensor optocoupler saat terdetek atau tidak

// int ledPin2 = 8; // pin that the LED is attached to // sebagai contoh pin aktuator

Serial.println(sensorValue2); // perintah menampilkan nilai sensor

/\*

// See if we can obtain or "Take" the Serial Semaphore.

// If the semaphore is not available, wait 5 ticks of the Scheduler to see if it becomes free.

if ( xSemaphoreTake( xSerialSemaphore, ( TickType\_t ) 5 ) == pdTRUE )

{

// We were able to obtain or "Take" the semaphore and can now access the shared resource.

// We want to have the Serial Port for us alone, as it takes some time to print,

// so we don't want it getting stolen during the middle of a conversion.

// print out the value you read:

Serial.println(sensorValue2);

xSemaphoreGive( xSerialSemaphore ); // Now free or "Give" the Serial Port for others.

}\*/

pinMode(8, OUTPUT); // pada eksekusi kali ini berbeda yaitu berkebalikan dengan seneor IR jika nilai kurang dari

if (sensorValue2 < 50) // 50 maka led 8 akan menyala, bila lebih dari 50 maka led akan menyala/ HIGH

{

digitalWrite(8, HIGH);

delay(50);

}

else (sensorValue2 > 50);

{

digitalWrite(8, LOW);

delay(50);

}

//delay(10); // one tick delay (15ms) in between reads for stability

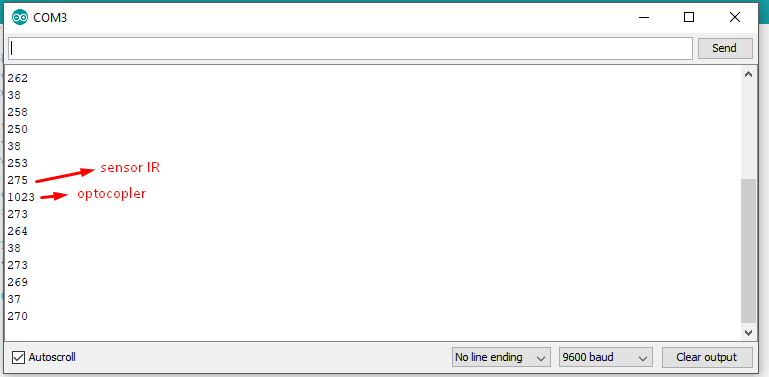
vTaskDelay(200); // one tick delay (15ms) in between reads for stability

}

vTaskDelay(1); // one tick delay (15ms) in between reads for stability // task delay yang digunakan untuk memberikan waktu pembacaan semua task selama 1 tick

}

C 1.3 Serial Monitor



1. **ANALISA**

Analisa Projek

Saat sensor IR di dekatkan objek indikator lampu yang terhubung pada sensor IR akan mati sedangkan saat sensor optocupler ada benda yang melewati sensor maka indicator akan hidup indicator ini bias di ganti dengan actuator lain semisal Relay

1. **KESIMPULAN**
2. FreeRTOS (Free Real-time operating systems) adalah sebuah *real time operating system* pada Arduino yang banyak digunakan oleh mikrokontroler untuk kebutuhan sistem operasinya.
3. Sintem bekerja degan baik meskipun sensor bekerja bersamaan karena menggunakan FreeRTOS yang bisa multitasking