|  |  |  |
| --- | --- | --- |
| **KELOMPOK** | 1 | **ACC** |
| **Tanggal Praktikum** | 27 Maret 2018 |  |
|  | Khairunnisa Sekar R. (D400150005) |  |
|  | Dian Arieska (D400150141) | **Tanggal ACC :** |
|  | Nindya Kaloka (D400170019) | **Revisi Tanggal :** |

**MODUL 1**

Analog Digital Read dengan FreeRTOS

1. **TUJUAN**
2. Praktikan dapat memahami salah satu sistem operasi yang ada pada Arduino yaitu FreeRTOS.
3. Praktikan dapat memahami program Analog Digital Read menggunakan sistem operasi FreeRTOS.
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 Uno
8. LED RGB
9. Resistor 220 ohm
10. Kabel jumper
11. **DASAR TEORI**

**FreeRTOS**

FreeRTOS (Free Real-time operating systems) adalah sebuah *real time operating system* pada Arduino. FreeRTOS ini banyak digunakan oleh mikrokontroler untuk kebutuhan sistem operasinya. FreeRTOS didistribusikan secara gratis lewat GPL dengan beberapa pengecualian untuk kalangan industri tertentu. RTOS (Real Time Operating System) adalah sistem operasi yang digunakan untuk memenuhi kebutuhan aplikasi yang bersifat real time. Real time disini berarti ia membutuhkan waktu kinerja setiap saat dimana ia dibutuhkan saat itu juga. Salah satu kunci dari keberhasilan RTOS adalah kemampuannya untuk melakukan kerja secara konsisten baik secara waktu yang ia butuhkan maupun secara task aplikasi yang mampu ia kerjakan.

Dalam FreeRTOS, itu sedikit berbeda. FreeRTOS dirancang khusus untuk dijalankan pada aplikasi dengan waktu yang sangat tepat dan tingkat keandalan yang tinggi. Jadi, RTOS digunakan ketika kendala sementara dalam suatu sistem sangat penting. Ada yang berbeda

jenis RTOS, Soft, Firm dan Hard. Sebelum masuk ke klasifiksi RTOS harus memperkenalkan terlebih dahulu mengenai jitter. Jitter adalah ukuran seberapa banyak waktu eksekusi dengan jadwal yang ditentukan atas iterasi selanjutnya. Sistem operasi real-time dioptimalkan meminimalkan jitter.

• Soft RTOS ditandai dengan jitter tinggi, Soft RTOS digunakanketika kendala waktu di mana tugas-tugas harus dijalankan kurang penting.Biasanya digunakan dalam sistem ketika risiko bahwa tugas tidak dijalankan pada waktunya tidak kritis.

• Hard RTOS yang keras memiliki lebih sedikit jitter daripada Soft RTOS, dalam jenis RTOS ini kendala waktu sangat penting dan jika tugas tidak dijalankan dalam waktu, hasilnya tidak lagi berguna dan sistem tidak berfungsi lagi.

• Firm RTOS terletak antara Soft RTOS dan Hard RTOS. Jumlah Jitternya kurang dari Soft RTOS dan lebih tinggi daripada Hard RTOS.

**LED**

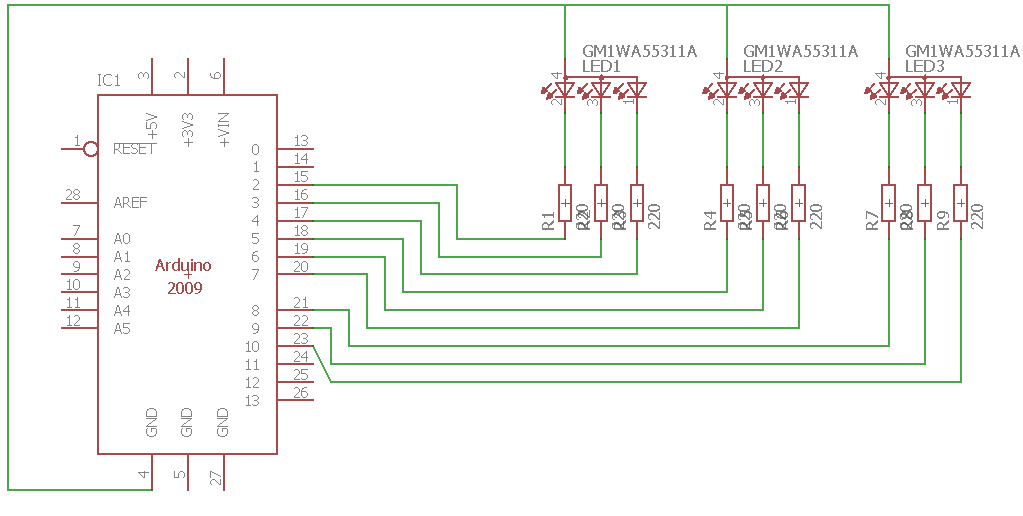
LED adalah singkatan dari “Light Emitting Diode”. Yang berarti LED adalah perangkat semi-konduktor yang menghasilkan cahaya ketika arus listrik melewati celah antara katoda dan anoda didalam sistem perangkat tsb. LED juga disebut “Solid State Lighting” karena chip LED disolder ke Printed Circuit Board (PCB) dan oleh karena itu tidak memiliki artikel-artikel yang longgar / filamen seperti bola lampu pijar, atau zat beracun seperti gas merkuri pada Lampu Hemat Energy (LHE).

LED telah beredar cukup lama, tetapi baru akhir-akhir ini produsen-produsen LED telah mulai memperluas lini produk mereka dari lampu dioda sederhana, yang digunakan terutama untuk lampu indikator, ke versi yang lebih canggih, lebih efisien dan lebih terang. Dalam dekade terakhir ini, kemajuan teknologi LED telah berhasil membuat LED yang lebih terang dan berdaya lebih tinggi untuk diaplikasikan ke dalam senter, lampu outdoor, lampu mobil, dan sebagainya. Dan hanya baru-baru ini, LED mulai digunakan secara luas untuk keperluan penerangan umum (penerangan dalam/luar ruangan, penerangan komersial, lampu dekorasi, lampu sorot, lampu panggung, dll). Ada beberapa jenis LED, tetapi yang kami gunakan pada praktikum ini adalah LED RGB. LED RGB merupakan LED yang mampu menghasilkan warna-warna dari hasil kombinasi warna Red (merah), Green (hijau), dan Blue ( biru). Yang mana warna keluaran warna dari led ini dapat kami atur dengan memberikan nilai input pada masing-masing kaki-kaki led untuk warna R-G-B. Adapun pada pembahasan kali ini yang akan Kami gunakan “LED –RGB Clear Common Cathode”.

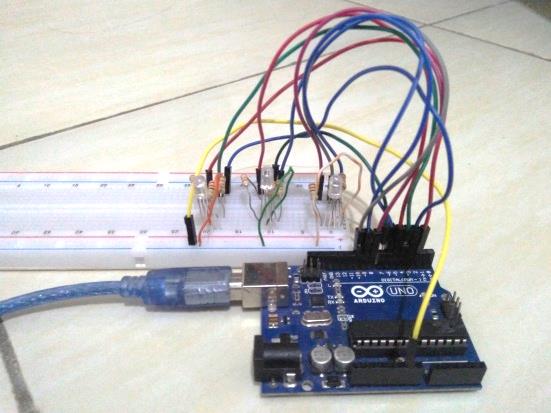
1. **HASIL PRAKTIKUM**

D.1 Hasil Percobaan 1

D.1.1 Skema Rangkaian



D.1.2 Foto Rangkaian

Sebelum Program Diupload Setelah Program Diupload

D.1.3 Script Program

#include <Arduino\_FreeRTOS.h>

//led 1

int redPin\_1 = 2;

int bluePin\_1 = 3;

int greenPin\_1 = 4;

//led 2

int redPin\_2 = 5;

int bluePin\_2 = 6;

int greenPin\_2 = 7;

//led 3

int redPin\_3 = 8;

int bluePin\_3 = 9;

int greenPin\_3 = 10;

void Task1( void \*pvParameters );

void Task2( void \*pvParameters );

void Task3( void \*pvParameters );

void setup() {

// put your setup code here, to run once:

//led 1

pinMode(redPin\_1, OUTPUT);

pinMode(bluePin\_1, OUTPUT);

pinMode(greenPin\_1, OUTPUT);

//led 2

pinMode(redPin\_2, OUTPUT);

pinMode(bluePin\_2, OUTPUT);

pinMode(greenPin\_2, OUTPUT);

//led 3

pinMode(redPin\_3, OUTPUT);

pinMode(bluePin\_3, OUTPUT);

pinMode(greenPin\_3, OUTPUT);

//create task

xTaskCreate(

Task1

, (const portCHAR \*)"Task1"

, 128

, NULL

, 1

, NULL );

xTaskCreate(

Task2

, (const portCHAR \*)"Task2"

, 128

, NULL

, 1

, NULL );

xTaskCreate(

Task3

, (const portCHAR \*)"Task3"

, 128

, NULL

, 1

, NULL );

}

void loop() {

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

}

//Task for leg 1: servo 2, servo 3, servo 4

void Task1(void \*pvParamaters) {

(void) pvParamaters;

//loop here

for (;;) {

setColor\_led1();

delay(500);

}

}

void Task2(void \*pvParamaters) {

(void) pvParamaters;

//loop here

for (;;) {

setColor\_led2();

delay(500);

}

}

void Task3(void \*pvParamaters) {

(void) pvParamaters;

//loop here

for (;;) {

setColor\_led3();

delay(500);

}

}

void setColor\_led1() {

digitalWrite(redPin\_1, HIGH);

digitalWrite(bluePin\_1, HIGH);

digitalWrite(greenPin\_1, LOW);

delay(500);

digitalWrite(redPin\_1, HIGH);

digitalWrite(bluePin\_1, LOW);

digitalWrite(greenPin\_1, HIGH);

delay(500);

digitalWrite(redPin\_1, LOW);

digitalWrite(bluePin\_1, HIGH);

digitalWrite(greenPin\_1, HIGH);

}

void setColor\_led2() {

digitalWrite(redPin\_2, HIGH);

digitalWrite(bluePin\_2, LOW);

digitalWrite(greenPin\_2, HIGH);

delay(500);

digitalWrite(redPin\_2, LOW);

digitalWrite(bluePin\_2, HIGH);

digitalWrite(greenPin\_2, HIGH);

delay(500);

digitalWrite(redPin\_2, HIGH);

digitalWrite(bluePin\_2, HIGH);

digitalWrite(greenPin\_2, LOW);

}

void setColor\_led3() {

digitalWrite(redPin\_3, LOW);

digitalWrite(bluePin\_3, HIGH);

digitalWrite(greenPin\_3, HIGH);

delay(500);

digitalWrite(redPin\_3, HIGH);

digitalWrite(bluePin\_3, HIGH);

digitalWrite(greenPin\_3, LOW);

delay(500);

digitalWrite(redPin\_3, HIGH);

digitalWrite(bluePin\_3, LOW);

digitalWrite(greenPin\_3, HIGH);

}

1. **ANALISA**

E.1. Analisa Percobaan 1

Pada percobaan ini menggunakan 1 buah Arduino Uno, 3 Buah LED RGB Common Cathode dan 9 buah resistor 200 ohm. 1 buah LED RGB dihubungkan dengan 3 resistor pada kaki Red, Green dan Blue. Kemudian kaki Resistor di jumper ke port Arduino pada digital pin 2 -10 Arduino. Sedangkan untuk kaki ground pada LED RGB dihubungkan pada port ground Arduino.

Pada program Arduino dengan menggunakan sistem operasi FreeRTOS percobaan ini menggunakan 3 Task, untuk memudahkan penyalaan warna lampu sesuai yang diinginkan. Pada perintah void Task1( void \*pvParameters ); void setup() { digunakan untuk mendaftarkan aplikasi atau task dalam sistem. Pada perintah xTaskCreate( digunakan untuk proses registrasi aplikasi dalam RTOS. Kemudian pada perintah void setColor\_led1(){ digunakan untuk membaca warna lampu yang akan menyala pada output LED RGB tersebut. Lampu akan teru menyala dengan delay 500ms tiap pergantian lampunya.

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. Pada sistem blink ini menggunakan RTOS multitasking dengan menggabungkan warna dasar LED RGB (Red, Green, Blue) dalam waktu yang sama.
4. Sistem operasi RTOS dalam program dibagi dalam beberapa task yang dapat diatur urutannya sesuai dengan kebutuhan.
5. Setiap task dapat disisipkan perintah loop ataupun tidak disisipkan sesuai dengan kebutuhan.