Petunjuk Praktikum Sistem Mikroprosesor

Modul 1 : Input Output Digital dan Interupsi Eksternal

Contents

Daftar Gambar

1 Tujuan

- Membuat aplikasi dengan Visual Code dan Arduino IDE
- Membuat aplikasi input dan output digital pada ESP32 dengan menggunakan bahasa pemrograman C
- Menggunakan interupsi eksternal untuk input digital

2 Teori

Teori sesuai dengan isi perkuliahan mengenai input dan output digital

Contoh kode untuk output digital dengan akses register ESP32

Contoh kode

```
#include "soc/gpio_reg.h"
#include "driver/gpio.h"

#define LED_PIN 2  // Built-in LED on most ESP32 boards
#define BLINK_DELAY_MS 500  // 500ms on + 500ms off = 1Hz

void setup() {
  // Configure GPIO pin as output using register access
  // Enable the GPIO pin in the output enable register
  REG_WRITE(GPIO_ENABLE_REG, REG_READ(GPIO_ENABLE_REG) | (1 << LED_PIN));

  // Set pin function to GPIO (not special function)
  PIN_FUNC_SELECT(GPIO_PIN_MUX_REG[LED_PIN], PIN_FUNC_GPIO);

  // Optional: Set pin to push-pull mode (default, but explicit)
  REG_SET_BIT(GPIO_PIN_REG[LED_PIN], GPIO_PIN_PAD_DRIVER);

  Serial.begin(115200);
  Serial.println("ESP32 Register-based LED Blink Started");
}

void loop() {
  // Turn LED ON - Set the bit high using register access
  REG_WRITE(GPIO_OUT_REG, REG_READ(GPIO_OUT_REG) | (1 << LED_PIN));
  Serial.println("LED ON");
  delay(BLINK_DELAY_MS);

  // Turn LED OFF - Clear the bit using register access
  REG_WRITE(GPIO_OUT_REG, REG_READ(GPIO_OUT_REG) & ~(1 << LED_PIN));
  Serial.println("LED OFF");
  delay(BLINK_DELAY_MS);
}
```

3 Tugas Pendahuluan

Hal-hal yang perlu dipersiapkan sebelum praktikum

- Instalasi Arduino IDE
- Instalasi PlatformIO di Visual Code Studio
- Mencoba compile contoh-contoh kode yang diberikan
- Mempersiapkan tugas-tugas kode yang diberikan.
- Menyiapkan tabel-tabel pengukuran di setiap percobaan

4 Alat dan Komponen yang digunakan

- Kit praktikum ESP32
- Osiloskop
- Frequency counter
- Signal generator

4.1 Percobaan

4.1.1 Persiapan

4.1.1.1 Catat Identitas Modul ESP32

Mencatat MAC address ESP32 sebagai identitas modul ESP32 yang dipakai. Pastikan ESP32 yang dipakai dapat diprogram dan memberikan output yang sesuai.

Berikut ini contoh kode untuk menampilkan MAC address[1]. MAC address dipakai untuk komunikasi nirkabel dengan WiFi, namun karena MAC address ini bersifat unik, maka dapat dipakai untuk identifikasi ESP32.

```
#include <WiFi.h>
#include <esp_wifi.h>

void readMacAddress(){
  uint8_t baseMac[6];
  esp_err_t ret = esp_wifi_get_mac(WIFI_IF_STA, baseMac);
  if (ret == ESP_OK) {
    Serial.printf("%02x:%02x:%02x:%02x:%02x:%02x\n",
                  baseMac[0], baseMac[1], baseMac[2],
                  baseMac[3], baseMac[4], baseMac[5]);
  } else {
    Serial.println("Failed to read MAC address");
  }
}

void setup(){
  Serial.begin(115200);

  WiFi.mode(WIFI_STA);
```

[1] https://randomnerdtutorials.com/get-change-esp32-esp8266-mac-address-arduino/

```
  WiFi.STA.begin();

  Serial.print("[DEFAULT] ESP32 Board MAC Address: ");
  readMacAddress();
}

void loop(){

}
```

4.1.1.2 Ukur tegangan VCC pada ESP32

Tegangan VCC pada ESP32 normalnya adalah 3 V sampai 3,3 V, namun dapat saja berubah tergantung pembebanan.

4.1.2 Mengukur Kecepatan Output Digital Dengan Library Arduino

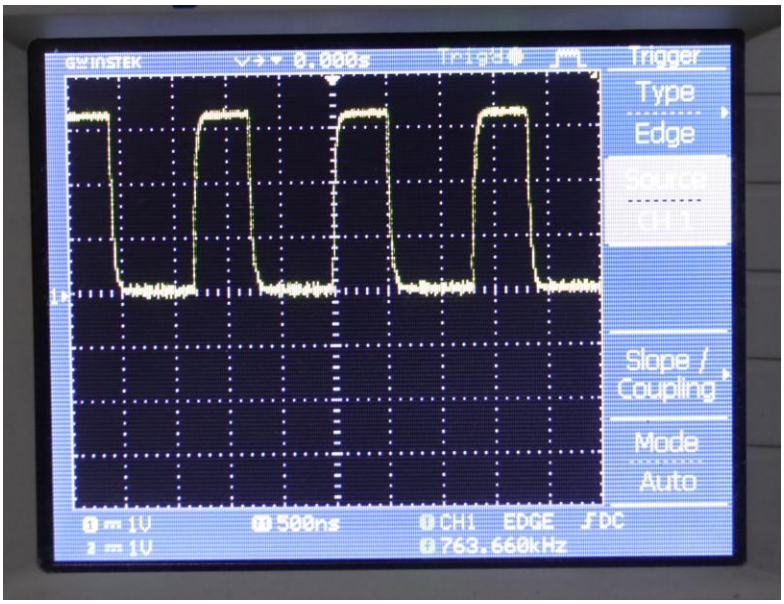
Proses input output digital memerlukan waktu. Pada percobaan ini diukur waktu yang diperlukan tersebut pada library Arduino dengan fungsi digitalWrite.

- Buatlah program untuk menghasilkan sinyal output persegi pada ESP32 tanpa delay dengan digitalwrite().
- Ukurlah frekuensi dan duty cycle sinyal tersebut.
- Ubahlah frekuensi ESP32 dan amati pengaruhnya

Contoh kode

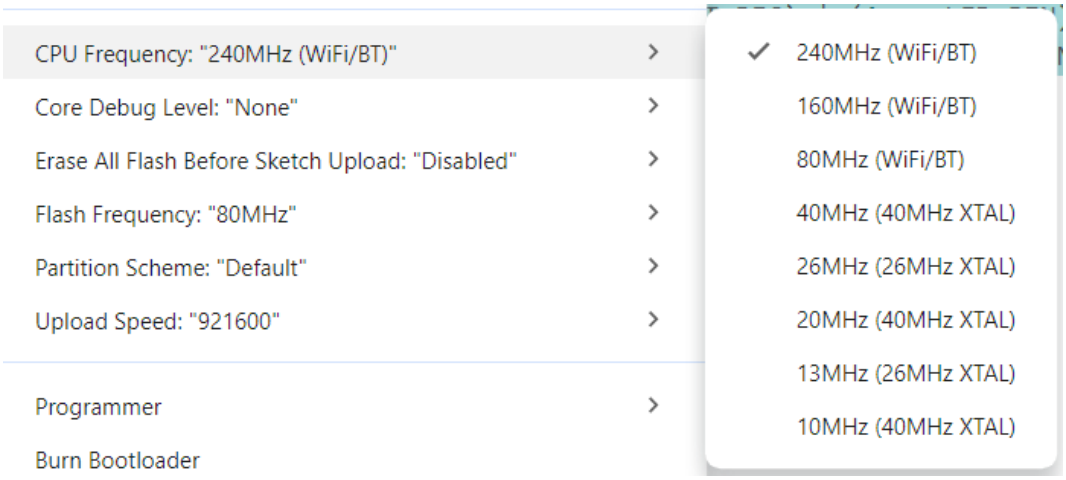
```
// LED kedip untuk Lolin32 Lite
#define LED_BUILTIN 22

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage level)
  digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the voltage LOW
}
```

Contoh output pada gambar berikut.



Gambar 1 Contoh output pengukuran frekuensi

Mengubah frekuensi dapat dilakukan dari kode, ataupun dari IDE Arduino di menu “Tools” -> “CPU Frequency”.



Gambar 2 Mengubah frekuensi clock ESP32 di Arduino IDE

Analisis:

- Apakah bentuknya simetris
- Pengaruh frekuensi clock terhadap bentuk dan frekuensi sinyal. Bisa membuat kurva korelasi dan analisis regresi

4.1.3 Mengukur Kecepatan Output Digital Dengan Akses Register

Proses input output digital memerlukan waktu. Pada percobaan ini diukur waktu yang diperlukan tersebut pada akses register. Akses register secara teoritis lebih cepat dibandingkan menggunakan library Arduino.

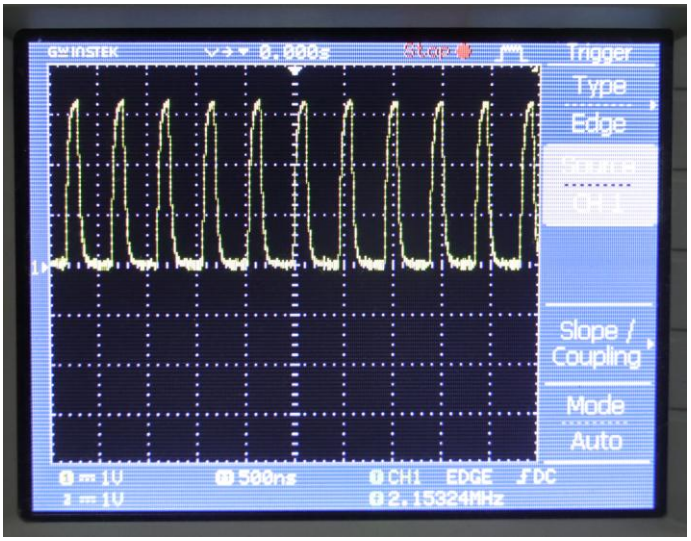
Tugas:

- Buatlah program untuk menghasilkan sinyal digital dengan frekuensi maksimum. Caranya dengan mengubah kode lampu kedip dengan menghilangkan delay.
- Buatlah pengukuran pada beberapa frekuensi clock ESP32

Contoh kode

```
#include "soc/gpio_reg.h"
#include "driver/gpio.h"

#define LED_PIN 22          // Lolin32

void setup() {
  pinMode(LED_PIN, OUTPUT);
  Serial.begin(115200);
  Serial.println("ESP32 Register-based LED Blink Started");
}

void loop() {
  REG_WRITE(GPIO_OUT_REG, REG_READ(GPIO_OUT_REG) | (1 << LED_PIN));
  REG_WRITE(GPIO_OUT_REG, REG_READ(GPIO_OUT_REG) & ~(1 << LED_PIN));
}
```

Contoh Output dapat dilihat pada gambar 4.



Gambar 3 Contoh pengukuran sinyal output

Berikut ini contoh kode untuk mengubah frekuensi clock ESP32 menjadi 240 MHz

```
#include "soc/gpio_reg.h"
```

```
#include "driver/gpio.h"

#define LED_PIN 22          // Lolin32

void setup() {
  pinMode(LED_PIN, OUTPUT);
  Serial.begin(115200);
  Serial.println("ESP32 Register-based LED Blink Started");
  setCpuFrequencyMhz(240);
}
void loop() {
  REG_WRITE(GPIO_OUT_REG, REG_READ(GPIO_OUT_REG) | (1 << LED_PIN));
  REG_WRITE(GPIO_OUT_REG, REG_READ(GPIO_OUT_REG) & ~(1 << LED_PIN));
}
```

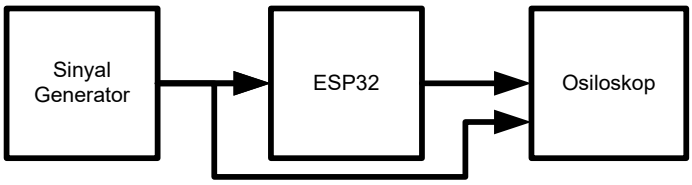
Analisis

- Lakukan analisis seperti percobaan dengan library Arduino

4.1.4 Latency Input Output Digital Dengan Library Arduino

Percobaan ini bertujuan mengukur latency pada ESP32. Latency pada konteks ini adalah jeda waktu antara sinyal masuk ke input mikrokontroler sampai dengan terjadi perubahan pada sinyal keluar. Latency tergantung salah satunya pada jenis akses IO yang dipakai. Akses GPIO menggunakan register seharusnya lebih cepat dibandingkan akses GPIO menggunakan library Arduino.

Susunan percobaan dapat dilihat pada Gambar 4.



Gambar 4 Skema pengukuran latency

Contoh kode

```
#define INPUT_1 25
#define OUTPUT_1 22

void setup() {
  pinMode(INPUT_1, INPUT_PULLUP);
  pinMode(OUTPUT_1, OUTPUT);
  Serial.begin(115200);
  Serial.println("ESP32 Started");
}

void loop() {
  int input1, input2;
  input1 = digitalRead(INPUT_1);
  digitalWrite(OUTPUT_1, input1);
}
```

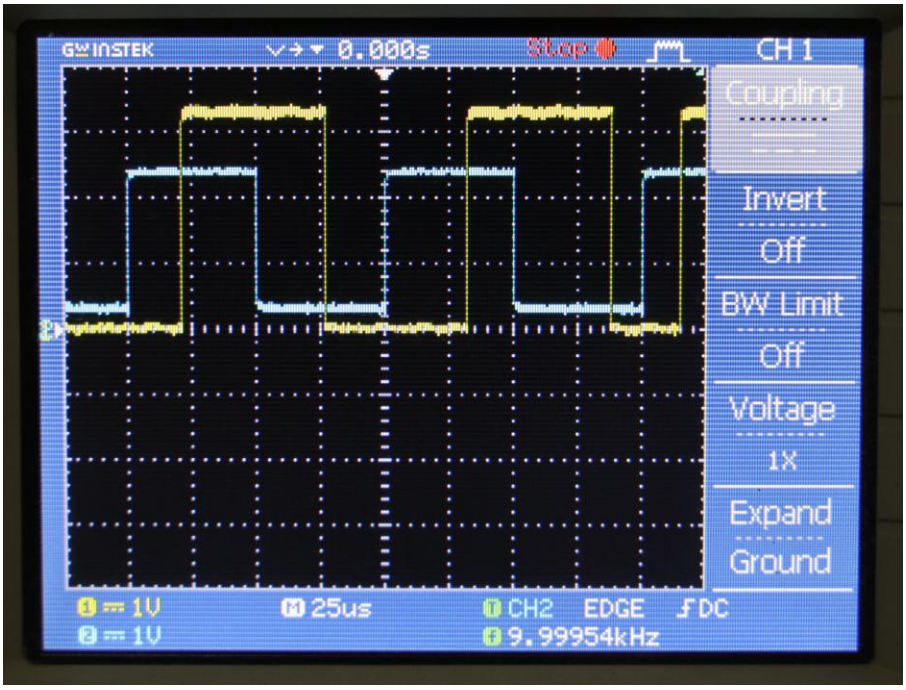
Perhatikan tegangan output dari signal generator harus sesuai dengan batasan ambang input ESP32 yang aman. Jika tegangan di luar batas ambang aman maka dapat menyebabkan kerusakan pada ESP32.

Ukurlah dulu tegangan dan frekuensi output generator sinyal dengan osiloskop. Jangan percaya 100% dengan tombol dan display di generator sinyal.

Frekuensi pada generator sinyal perlu diubah-ubah untuk dapat mengamati latency. Carilah frekuensi yang tidak terlalu tinggi dan tidak terlalu rendah. Perubahan input terlalu cepat dapat saja tidak terdeteksi oleh software (berapa limitnya?).

Berikut ini contoh pengukuran latency dengan frekuensi sinyal 10 kHz. Dapat didiskusikan apakah frekuensi ini kurang tinggi atau kurang rendah.

Perhatikan bahwa latency ini statistical, jadi perlu beberapa sampel untuk mendapatkan nilai rata-rata dan standard deviation. Jumlah sampel yang diambil disesuaikan dengan waktu yang tersedia.



Gambar 5 Contoh perbandingan sinyal input ESP32 dan output ESP32

Contoh video perekaman latency dapat dilihat di
https://www.youtube.com/watch?v=lbZnq-zKaQc

4.1.5 Latency Input Output Digital Dengan Akses Register

Seperti percobaan sebelumnya namun menggunakan akses register. Seharusnya latency lebih kecil dengan register.

4.1.6 Pengendali Relay

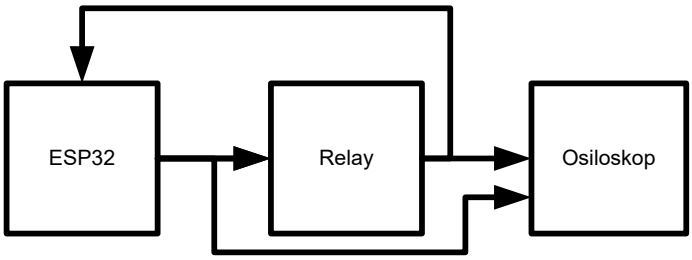
Relay dapat digunakan untuk mengatur perangkat dengan tegangan yang berbeda dengan mikrokontroler. Namun demikian relay memiliki keterbatasan kecepatan. Percobaan ini adalah mengukur waktu respon dari relay elektromekanik.

- Percobaan software untuk mengendalikan relay
- Percobaan software untuk mengukur delay relay dengan menggunakan millis(). Perhatian: statistical
- Buat software untuk mengukur delay relay dengan menggunakan osiloskop.

Tugas

- Mengukur delay pada output relay

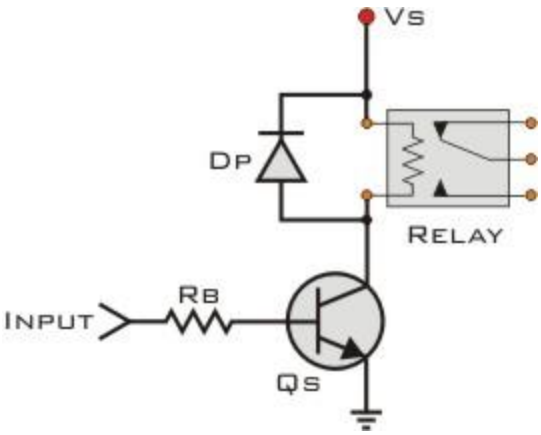
Skema sistem pengujian adalah sebagai berikut



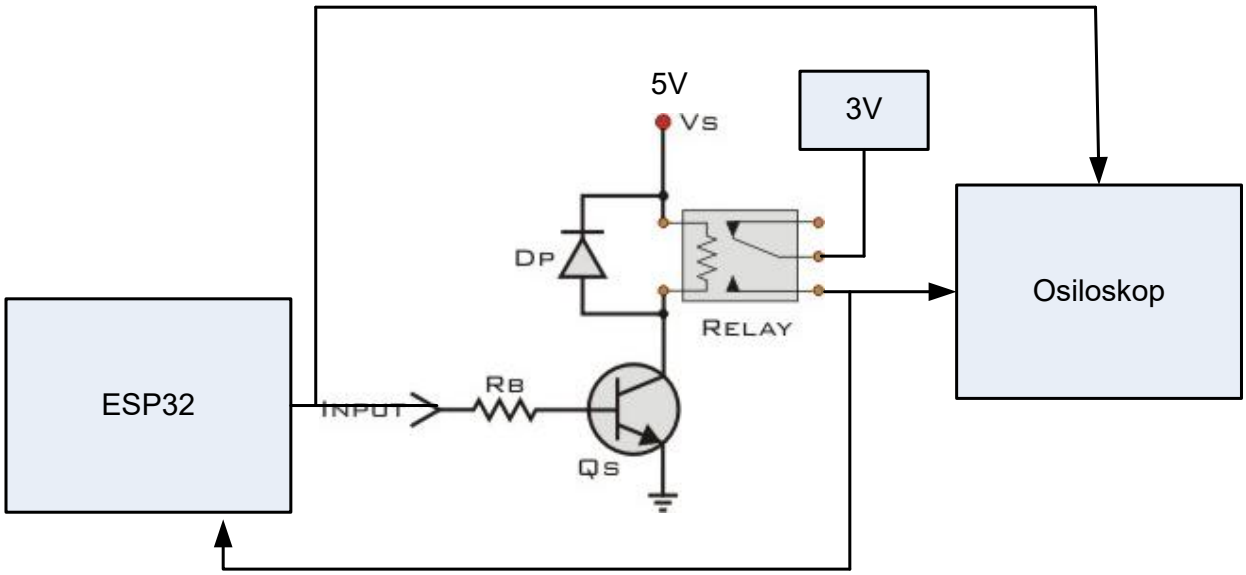
Gambar 6 Diagram blok sistem pengujian relay

Rangkaian dasar pengendali relay[2]:

[2] Dari https://forum.arduino.cc/t/transistor-with-relay-burns-out/297857



Berikut ini skema pengukuran relay



Gambar 7 Skema rangkaian pengukuran relay

Pelajari dulu cara kerja relay dan rangkaian pengendali relay. Perhatikan bahwa relay memerlukan tegangan kerja 5 volt. Input relay dapat berfungsi pada input 3V. Dalam prakteknya kontak relay dapat mengendalikan tegangan cukup tinggi (220 volt), namun dalam percobaan ini cukup menggunakan 3 volt saja sesuai dengan tegangan kerja ESP32.

Output ESP32 mengendalikan relay melalui transistor. Input ESP32 untuk mengukur tegangan yang dihasilkan pada relay. Tegangan pada kontak relay akan berubah setelah ESP32 memberikan sinyal kendali ke transistor. Perbedaan waktu ini diukur dengan perangkat lunak pada ESP32 dan dengan osiloskop.

Contoh output seperti pada percobaan sebelumnya namun latency lebih lambat (Gambar 5).

Tugas:

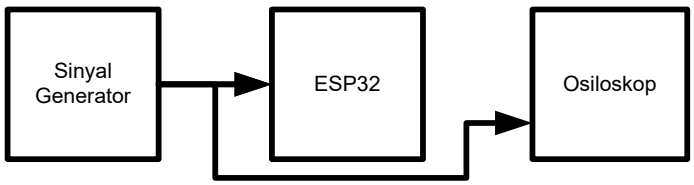
- Buat program di ESP32 untuk membangkitkan sinyal segi empat ke relay dan mengukur jeda antara sinyal kendali tersebut dan output relay. Frekuensi sinyal pada percobaan perlu disesuaikan dengan kecepatan relay.
- Buat diagram alir (flowchart) program dan kode program
- Ukur waktu respon relay dengan menggunakan fungsi millis() dari Arduino
- Ukur waktu respon relay dengan osiloskop dual kanal.

4.1.7 Pengukur Frekuensi

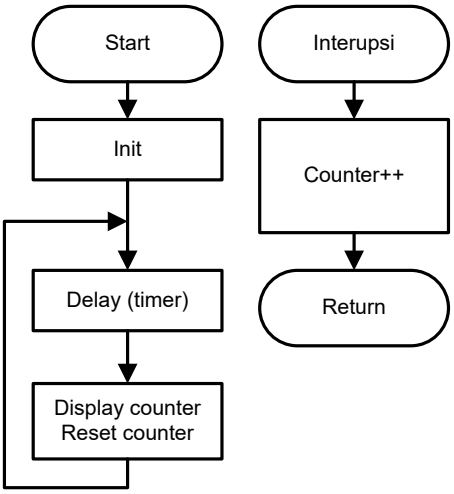
Buatlah program dengan spesifikasi:

- Menghitung kecepatan pulsa dari sinyal generator dengan interupsi eksternal dan timer. Input: generator sinyal. Output: tampilan frekuensi di port serial.
- Ukurlah sampai frekuensi berapa yang dapat diukur oleh ESP32 tanpa ada pulsa yang hilang tidak terdeteksi.

Pada Gambar 9 dapat dilihat diagram alir perangkat lunak peghitung pulsa dengan menggunakan delay sebagai sumber waktu



Gambar 8 Rangkaian pengujian interupsi eksternal



Gambar 9 Diagram alir penghitung pulsa dengan interupsi eksternal dan delay loop

5 Literatur

- Situs https://randomnerdtutorials.com/
- Situs Electronic Wings https://www.electronicwings.com/esp32
- Datasheet ESP32 di situs Espressif