LAMPIRAN 1
(PROGRAM)

Program Lengkap

```
#define TIMER_INTERRUPT_DEBUG         0
#define ESP32_ISR_Servos_DEBUG        1

// For ESP32_C3, select ESP32 timer number (0-1)
// For ESP32 and ESP32_S2, select ESP32 timer number (0-3)

#define USE_ESP32_TIMER_NO         3
#include "ESP32_ISR_Servo.h"

#define pin_servo   19
#define pin_motor_DC   4

// penentuan nilai limit atas dan bawah untuk pembangkitan pulsa pwm oleh ESP8266 untuk mengontrol sudut motor servo daro 0 sampai 180 derajat
// nlai ini dapat diubah2 disesuaiakan dengan jenis motor servo agar diperoleh sudut yang presisi 0 dan presisi 180 derajat,
#define MIN_MICROS      490
#define MAX_MICROS      2510

int myServo   = -1;

#include "lcd16x2.h"
#include "keypad4x4.h"
#include "tcs34725.h"
#include "fingerprint.h"
#include "rtc_ds1307.h"

#include "UbidotsEsp32Mqtt.h"

const char *UBIDOTS_TOKEN = "BBFF-jNSZqAEiY6SRoqfmdgVfIKCdoyZzmg";   // Put here your Ubidots TOKEN
const char *WIFI_SSID = "ETN2";      // Put here your Wi-Fi SSID
const char *WIFI_PASS = "gaknormal";      // Put here your Wi-Fi password
Ubidots ubidots(UBIDOTS_TOKEN);
unsigned long timer;

// referensi EEPROM https://randomnerdtutorials.com/esp32-flash-memory/
// include library to read and write from flash memory
#include <EEPROM.h> // esp32 punya 512 byte address

#define address_ID_finger_terdaftar  0

// saat input nama user dibatasi hanya maksimal 10 karakter huruf karena addresnya hanya dikasih 10 byte
#define address_nama_user1    1
#define address_nama_user2    11
#define address_nama_user3    21

// nilai uang menggunakan variabel 4 byte . jadi angka dimulai dari 0 s/d 4294967295
#define address_uang_user1    101
#define address_uang_user2    111
#define address_uang_user3    121

#define address_total_uang_tabungan    201
```

```
//---------------------------- Variable tambahan -------------------------


#define max_jumlah_user   3
#define user_1            1
#define user_2            2
#define user_3            3

uint8_t   urutan_ID_finger_terdaftar;
String    nama_user_1;
uint32_t  uang_user_1;
String    nama_user_2;
uint32_t  uang_user_2;
String    nama_user_3;
uint32_t  uang_user_3;
unsigned long total_uang_tabungan;

int counter_reset = 0;
uint32_t lastMillis2 = 0;

void callback(char *topic, byte *payload, unsigned int length)
{
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i = 0; i < length; i++)
  {
    Serial.print((char)payload[i]);
  }
  Serial.println();
}

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

  pinMode(pin_motor_DC, OUTPUT); // pin 4 terhubung ke pin input dari modul relay
  digitalWrite(pin_motor_DC, HIGH);  // jika pin 4 bernilai HIGH maka relay akan OFF

  myServo   = ESP32_ISR_Servos.setupServo(pin_servo, MIN_MICROS, MAX_MICROS);
  ESP32_ISR_Servos.setPosition(myServo, 0); // pintu tertutup
  //ESP32_ISR_Servos.setPosition(myServo, 90);  // pintu terbuka

  setup_lcd16x2();
  setup_keypad4x4();
  setup_tcs34725();
  setup_fingerprint();

  Serial.println("\nInit EEPROM\n");
  if (!EEPROM.begin(1000))
  {
    Serial.println("Failed to initialise EEPROM");
    Serial.println("Restarting...");
    delay(1000);
    ESP.restart();
  }
  else // jika init eeprom sukses, maka ambil semua data
  {
    urutan_ID_finger_terdaftar = EEPROM.readUChar(address_ID_finger_terdaftar);
```

```
    nama_user_1 = EEPROM.readString(address_nama_user1);
    uang_user_1 = EEPROM.readULong(address_uang_user1);

    nama_user_2 = EEPROM.readString(address_nama_user2);
    uang_user_2 = EEPROM.readULong(address_uang_user2);

    nama_user_3 = EEPROM.readString(address_nama_user3);
    uang_user_3 = EEPROM.readULong(address_uang_user3);

    total_uang_tabungan = EEPROM.readULong(address_total_uang_tabungan);
   }

   lcd.setCursor(0, 0);
   lcd.print("Sedang Konek");
   lcd.setCursor(0, 1);
   lcd.print("ke Wifi!");

   ubidots.connectToWifi(WIFI_SSID, WIFI_PASS);
   ubidots.setCallback(callback);
   ubidots.setup();
   ubidots.reconnect();

   lcd.clear();

   lastMillis2 = millis();
}

void loop()
{
  if (!ubidots.connected())
  {
    ubidots.connect();
  }

  if (ubidots.connected())
  {
    ubidots.loop();
  }

  int uang = scan_uang_kertas();
  if (uang == uang_tidak_terdeteksi)
  {
    unsigned long t = millis() - lastMillis2;
    if(t < 1500 && urutan_ID_finger_terdaftar != 0) // artinya hanya ketika sudah ada user yang daftar
    {
      lcd.setCursor(0, 3); lcd.print("(A) Add | (D) Delete");
    }
    else if(t > 1500 && t < 3000 && urutan_ID_finger_terdaftar != 0) // artinya hanya ketika sudah ada user
yang daftar
    {
      DateTime now;
      now = rtc.now();

      lcd.setCursor(0, 3);
      lcd.print(now.day(), DEC);
      lcd.print('/');
      lcd.print(now.month(), DEC);
      lcd.print('/');
      lcd.print(now.year(), DEC);
      lcd.print("    ");
```

```
    lcd.print(now.hour(), DEC);
    lcd.print(':');
    lcd.print(now.minute() , DEC);
    lcd.print(':');
    lcd.print(now.second(), DEC);
   }
   else if(t > 3000  && urutan_ID_finger_terdaftar != 0)
   {
    lcd.setCursor(10, 3); lcd.print("          ");
    lcd.setCursor(0, 3); lcd.print("Total: Rp " + String(total_uang_tabungan));

    if(t > 5000 && urutan_ID_finger_terdaftar != 0)
    {
     lastMillis2 = millis();
    }
   }
}
else if (uang == uang100rb)
{
 lcd.setCursor(10, 3); lcd.print("Rp 100.000");
}
else if (uang == uang50rb)
{
 lcd.setCursor(10, 3); lcd.print("Rp 50.000 ");
}
else if (uang == uang20rb)
{
 lcd.setCursor(10, 0); lcd.print("Rp 20.000 ");
}
else if (uang == uang10rb)
{
 lcd.setCursor(10, 3); lcd.print("Rp 10.000 ");
}
else if (uang == uang5rb)
{
 lcd.setCursor(10, 3); lcd.print("Rp 5.000  ");
}
else if (uang == uang2rb)
{
 lcd.setCursor(10, 3); lcd.print("Rp 2.000  ");
}
else if (uang == uang1rb)
{
 lcd.setCursor(10, 3); lcd.print("Rp 1.000    ");
}

int8_t id_finger = getFingerprintID();
Serial.println("id_finger:" + String(id_finger));

if(id_finger == user_1) // 1
{
 lcd.setCursor(0, 3); lcd.print("          ");
 lcd.setCursor(0, 3); lcd.print(nama_user_1);
 if (uang > 0)
 {
   uang_user_1 = uang_user_1 + daftar_uang[uang];
   total_uang_tabungan = total_uang_tabungan + daftar_uang[uang];
   EEPROM.writeULong(address_uang_user1, uang_user_1); // nilai_awal_uang_user adalah nol rupiah
   EEPROM.writeULong(address_total_uang_tabungan, total_uang_tabungan);
   EEPROM.commit(); // setiap kali write data WAJIB diakhiri dengan commit()
```

```
//      char nama_user[20];
//      nama_user_1.toCharArray(nama_user, sizeof(nama_user));
//      ubidots.add(nama_user, uang_user_1); // Insert your variable Labels and the value to be sent
      ubidots.add("user_1", uang_user_1); // Insert your variable Labels and the value to be sent
      ubidots.add("total_uang", total_uang_tabungan); // Insert your variable Labels and the value to be sent
      ubidots.publish("Tabungan_IoT");
      delay(500);

      // aktifkan motor penggerak masukin uang
      digitalWrite(pin_motor_DC, LOW);// jika pin 7 bernilai LOW maka relay akan ON, sehingga motor akan
menarik uang kedalam
      delay(1500);  // diberi delay atau jeda sampai sekirannya uang telah masuk semua
      digitalWrite(pin_motor_DC, HIGH);// jika pin 7 bernilai HIGH maka relay akan OFF
      lcd.clear();
    }
  }
  else if(id_finger == user_2)
  {
    lcd.setCursor(0, 3); lcd.print("          ");
    lcd.setCursor(0, 3); lcd.print(nama_user_2);
    if (uang > 0)
    {
      uang_user_2 = uang_user_2 + daftar_uang[uang];
      total_uang_tabungan += daftar_uang[uang];
      EEPROM.writeULong(address_uang_user2, uang_user_2); // nilai_awal_uang_user adalah nol rupiah
      EEPROM.writeULong(address_total_uang_tabungan, total_uang_tabungan);
      EEPROM.commit(); // setiap kali write data WAJIB diakhiri dengan commit()

//      char nama_user[20];
//      nama_user_2.toCharArray(nama_user, sizeof(nama_user));
//      ubidots.add(nama_user, uang_user_2); // Insert your variable Labels and the value to be sent
      ubidots.add("user_2", uang_user_2); // Insert your variable Labels and the value to be sent
      ubidots.add("total_uang", total_uang_tabungan); // Insert your variable Labels and the value to be sent
      ubidots.publish("Tabungan_IoT");
      delay(500);

      // aktifkan motor penggerak masukin uang
      digitalWrite(pin_motor_DC, LOW);// jika pin 7 bernilai LOW maka relay akan ON, sehingga motor akan
menarik uang kedalam
      delay(1500);  // diberi delay atau jeda sampai sekirannya uang telah masuk semua
      digitalWrite(pin_motor_DC, HIGH);// jika pin 7 bernilai HIGH maka relay akan OFF
      lcd.clear();
    }
  }
  else if(id_finger == user_3)
  {
    lcd.setCursor(0, 3); lcd.print("          ");
    lcd.setCursor(0, 3); lcd.print(nama_user_3);
    if (uang > 0)
    {
      uang_user_3 = uang_user_3 + daftar_uang[uang];
      total_uang_tabungan += daftar_uang[uang];
      EEPROM.writeULong(address_uang_user3, uang_user_3); // nilai_awal_uang_user adalah nol rupiah
      EEPROM.writeULong(address_total_uang_tabungan, total_uang_tabungan);
      EEPROM.commit(); // setiap kali write data WAJIB diakhiri dengan commit()

//      char nama_user[20];
//      nama_user_3.toCharArray(nama_user, sizeof(nama_user));
//      ubidots.add(nama_user, uang_user_3); // Insert your variable Labels and the value to be sent
```

```
    ubidots.add("user_3", uang_user_3); // Insert your variable Labels and the value to be sent
    ubidots.add("total_uang", total_uang_tabungan); // Insert your variable Labels and the value to be sent
    ubidots.publish("Tabungan_IoT");
    delay(500);

    // aktifkan motor penggerak masukin uang
    digitalWrite(pin_motor_DC, LOW);// jika pin 7 bernilai LOW maka relay akan ON, sehingga motor akan
menarik uang kedalam
    delay(1500);  // diberi delay atau jeda sampai sekirannya uang telah masuk semua
    digitalWrite(pin_motor_DC, HIGH);// jika pin 7 bernilai HIGH maka relay akan OFF
    lcd.clear();
   }
  }

  // jika user tekan tombol A pada keypada maka alat akan mendeteksi sebagai ADD user baru
  if (urutan_ID_finger_terdaftar == 0) // jika alat masih kosong, belum ada user yang daftar finger
  {
   lcd.setCursor(0, 0);
   lcd.print("Belum ada Data User!");
   lcd.setCursor(0, 1);
   lcd.print("Mohon Register User!");
   lcd.setCursor(0, 3);
   lcd.print("(A) Add | (D) Delete");
   //delay(5000);
   //lcd.clear();
  }
  else // jika sudah terdapat user yang terdaftar
  {
   if (nama_user_1 != "")
   {
    lcd.setCursor(0, 0);
    lcd.print(nama_user_1);
    lcd.setCursor(9, 0);
    lcd.print(":Rp ");
    lcd.print(uang_user_1);
    //lcd.print(",-");
   }

   if (nama_user_2 != "")
   {
    lcd.setCursor(0, 1);
    lcd.print(nama_user_2);
    lcd.setCursor(9, 1);
    lcd.print(":Rp ");
    lcd.print(uang_user_2);
    //lcd.print(",-");
   }

   if (nama_user_3 != "")
   {
    lcd.setCursor(0, 2);
    lcd.print(nama_user_3);
    lcd.setCursor(9, 2);
    lcd.print(":Rp ");
    lcd.print(uang_user_3);
    //lcd.print(",-");
   }

   //delay(5000);
   //lcd.clear();
```

```
}

loop_keypad();

if (isEnterName == true && urutan_ID_finger_terdaftar < 3)
{
 lcd.clear();
 // proses pertama masukkan nama
 while (isEnterName == true)
 {
  lcd.setCursor(0, 0);
  lcd.print("Nama Panggilan:");

  loop_keypad();
 }

 // stelah input nama lanjut daftarin fingerprin
 while (isEnrollFinger == true)
 {
  uint8_t id_tujuan;

  if (urutan_ID_finger_terdaftar == 0)
  {
   id_tujuan = user_1;
  }
  else if (urutan_ID_finger_terdaftar == 1)
  {
   id_tujuan = user_2;
  }
  else if (urutan_ID_finger_terdaftar == 2)
  {
   id_tujuan = user_3;
  }

  //      Serial.println("id_tujuan:" + String(id_tujuan));
  //      Serial.println("nama_user:" + nama_user[id_tujuan]);
  //      Serial.println("uang_user:" + String(uang_user[id_tujuan]));

  int p = getFingerprintEnroll(id_tujuan);
  Serial.println(p);

  if (p == FINGERPRINT_OK) // jika pendaftarn sukses
  {
   lcd.setCursor(0, 3);
   lcd.print("Pendaftaran Sukses! ");
   delay(2000);
   lcd.setCursor(0, 3);
   lcd.print("Lepaskan Jari Anda! ");
   delay(2000);

   urutan_ID_finger_terdaftar = id_tujuan;

   if (urutan_ID_finger_terdaftar == user_1) // 1
   {
    nama_user_1 = s; // ambil nama hasil input keypad
    s = "";        // lalu kosongkan variabel
    uang_user_1 = 0; // nilai awal uang adalah nol rupiah

    EEPROM.writeUChar(address_ID_finger_terdaftar, urutan_ID_finger_terdaftar);
    EEPROM.writeString(address_nama_user1, nama_user_1);
```

```
      EEPROM.writeULong(address_uang_user1, uang_user_1); // nilai_awal_uang_user adalah nol rupiah
      EEPROM.commit(); // setiap kali write data WAJIB diakhiri dengan commit()
    }
    else if (urutan_ID_finger_terdaftar == user_2) // 2
    {
      nama_user_2 = s; // ambil nama hasil input keypad
      s = "";       // lalu kosongkan variabel
      uang_user_2 = 0; // nilai awal uang adalah nol rupiah

      EEPROM.writeUChar(address_ID_finger_terdaftar, urutan_ID_finger_terdaftar);
      EEPROM.writeString(address_nama_user2, nama_user_2);
      EEPROM.writeULong(address_uang_user2, uang_user_2); // nilai_awal_uang_user adalah nol rupiah
      EEPROM.commit(); // setiap kali write data WAJIB diakhiri dengan commit()
    }
    else if (urutan_ID_finger_terdaftar == user_3) // 3
    {
      nama_user_3 = s; // ambil nama hasil input keypad
      s = "";       // lalu kosongkan variabel
      uang_user_3 = 0; // nilai awal uang adalah nol rupiah

      EEPROM.writeUChar(address_ID_finger_terdaftar, urutan_ID_finger_terdaftar);
      EEPROM.writeString(address_nama_user3, nama_user_3);
      EEPROM.writeULong(address_uang_user3, uang_user_3); // nilai_awal_uang_user adalah nol rupiah
      EEPROM.commit(); // setiap kali write data WAJIB diakhiri dengan commit()
    }

    lcd.clear();

    isEnrollFinger = false; // dibikin false supaya keluar ddari looping while
   }
   else // jika pendaftaran gagal
   {
    lcd.setCursor(0, 3);
    lcd.print("Pendaftaran Gagal!  ");
    delay(2000);
    lcd.setCursor(0, 3);
    lcd.print("Coba Ulangi Lagi!   ");
    delay(3000);

    isEnrollFinger = true; // dibikin true karena harus mengulangi proses enroll
   }
  }
}

if (isDeleteUser == true)
{
  deleteFingerprint(1);
  delay(10);
  deleteFingerprint(2);
  delay(10);
  deleteFingerprint(3);
  delay(10);

  urutan_ID_finger_terdaftar = 0;
  nama_user_1 = "";
  uang_user_1 = 0;
  nama_user_2 = "";
  uang_user_2 = 0;
  nama_user_3 = "";
  uang_user_3 = 0;
```

```
    total_uang_tabungan = 0;

    EEPROM.writeUChar(address_ID_finger_terdaftar, 0);

    EEPROM.writeString(address_nama_user1, nama_user_1);
    EEPROM.writeULong(address_uang_user1, uang_user_1);

    EEPROM.writeString(address_nama_user2, nama_user_2);
    EEPROM.writeULong(address_uang_user2, uang_user_2);

    EEPROM.writeString(address_nama_user3, nama_user_3);
    EEPROM.writeULong(address_uang_user3, uang_user_3);

    EEPROM.commit(); // setiap kali write data WAJIB diakhiri dengan commit()

    ubidots.add("user_1", uang_user_1); // Insert your variable Labels and the value to be sent
    ubidots.add("user_2", uang_user_2); // Insert your variable Labels and the value to be sent
    ubidots.add("user_3", uang_user_3); // Insert your variable Labels and the value to be sent
    ubidots.add("total_uang", total_uang_tabungan); // Insert your variable Labels and the value to be sent
    ubidots.publish("Tabungan_IoT");

    lcd.clear();

    lcd.setCursor(0, 0);
    lcd.print("ID 1 Sukses Terhapus");
    lcd.setCursor(0, 1);
    lcd.print("ID 2 Sukses Terhapus");
    lcd.setCursor(0, 2);
    lcd.print("ID 3 Sukses Terhapus");
    delay(2000);

    lcd.clear();

    isDeleteUser = false;
  }

  if(isNeedOpenDoor == true)
  {
    if (ESP32_ISR_Servos.getPosition(myServo) == 0) // jika saat ini pintu dalam kondisi terbuka maka selanjutnya akan dikunci
    {
      ESP32_ISR_Servos.setPosition(myServo, 90);  // pintu terbuka
      Serial.println("Buka Pintu");
      delay(1000);
    }
    else if (ESP32_ISR_Servos.getPosition(myServo) == 90) // jika saat ini pintu dalam kondisi terkunci maka selanjutnya akan dibuka
    {
      ESP32_ISR_Servos.setPosition(myServo, 0); // pintu tertutup
      Serial.println("Tutup Pintu");
      delay(1000);
    }

    isNeedOpenDoor = false;
  }

  delay(50);
}
```

Fingerprint

```
#include <Adafruit_Fingerprint.h>
#include <HardwareSerial.h>

Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Serial2);

uint8_t getFingerprintEnroll(uint8_t id)  // ID #0 not allowed, try again!
{
 Serial.print("Enrolling ID #");
 Serial.println(id);
 lcd.setCursor(0, 2);
 lcd.print("Enroll Finger ID  #");
 lcd.print(id);
 delay(2000);

 int p = -1;
 Serial.print("Waiting for valid finger to enroll as #"); Serial.println(id);
 while (p != FINGERPRINT_OK)
 {
  p = finger.getImage();
  switch (p)
  {
    case FINGERPRINT_OK:
      Serial.println("Image taken"); // 2
      lcd.setCursor(0, 2);
      lcd.print("Image Diambil!      ");
      break;
    case FINGERPRINT_NOFINGER:
      Serial.println("Place Finger!"); // 1
      lcd.setCursor(0, 2);
      lcd.print("Letakkan Jari!      ");
      break;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      lcd.setCursor(0, 2);
      lcd.print("Communication error!");
      break;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println("Imaging error  ");
      lcd.setCursor(0, 2);
      lcd.print("Imaging error!      ");
      break;
    default:
      Serial.println("Unknown error");
      lcd.setCursor(0, 2);
      lcd.print("Unknown error!      ");
      break;
  }
 }

 // OK success!

 p = finger.image2Tz(1);
 switch (p)
 {
  case FINGERPRINT_OK:
    Serial.println("Image converted"); // 3
    lcd.setCursor(0, 2);
    lcd.print("Image Terkonversi!  ");
```

```
    break;
  case FINGERPRINT_IMAGEMESS:
    Serial.println("Image too messy");
    lcd.setCursor(0, 2);
    lcd.print("Image too messy!    ");
    return p;
  case FINGERPRINT_PACKETRECIEVEERR:
    Serial.println("Communication error");
    lcd.setCursor(0, 2);
    lcd.print("Communication error!");
    return p;
  case FINGERPRINT_FEATUREFAIL:
    Serial.println("Could not find fingerprint features");
    lcd.setCursor(0, 2);
    lcd.print("Could'n find finger!");
    return p;
  case FINGERPRINT_INVALIDIMAGE:
    Serial.println("Could not find fingerprint features");
    lcd.setCursor(0, 2);
    lcd.print("Could'n find finger!");
    return p;
  default:
    Serial.println("Unknown error");
    lcd.setCursor(0, 2);
    lcd.print("Imaging error!      ");
    return p;
}

Serial.println("Remove finger"); // 4
lcd.setCursor(0, 2);
lcd.print("Lepaskan Jari Anda! ");
delay(4000);

p = 0;
while (p != FINGERPRINT_NOFINGER)
{
  p = finger.getImage();
}

Serial.print("ID "); Serial.println(id); // 5
delay(200);

p = -1;
Serial.println("Place same finger again"); // 6
lcd.setCursor(0, 2);
lcd.print("Letakkan Jari Lagi! ");

while (p != FINGERPRINT_OK)
{
  p = finger.getImage();
  switch (p)
  {
    case FINGERPRINT_OK:
      Serial.println("Image taken"); // 8
      lcd.setCursor(0, 2);
      lcd.print("Image Diambil!      ");
      break;
    case FINGERPRINT_NOFINGER:
      Serial.println("Place Finger Again!"); // 7
      lcd.setCursor(0, 2);
```

```
      lcd.print("Letakkan Jari Lagi! ");
      break;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      lcd.setCursor(0, 2);
      lcd.print("Communication error!");
      break;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println("Imaging error");
      lcd.setCursor(0, 2);
      lcd.print("Imaging error!      ");
      break;
    default:
      Serial.println("Unknown error");
      lcd.setCursor(0, 2);
      lcd.print("Unknown error!      ");
      break;
  }
}

// OK success!

p = finger.image2Tz(2);
switch (p)
{
  case FINGERPRINT_OK:
    Serial.println("Image converted"); // 9
    lcd.setCursor(0, 2);
    lcd.print("Image Terkonversi!  ");
    break;
  case FINGERPRINT_IMAGEMESS:
    Serial.println("Image too messy");
    return p;
  case FINGERPRINT_PACKETRECIEVEERR:
    Serial.println("Communication error");
    return p;
  case FINGERPRINT_FEATUREFAIL:
    Serial.println("Could not find fingerprint features");
    return p;
  case FINGERPRINT_INVALIDIMAGE:
    Serial.println("Could not find fingerprint features");
    return p;
  default:
    Serial.println("Unknown error");
    return p;
}

// OK converted!
Serial.print("Creating model for #"); Serial.println(id); // 10
lcd.setCursor(0, 2);
lcd.print("Buat Model    ID  #");
lcd.print(id);

p = finger.createModel();
if (p == FINGERPRINT_OK) {
  Serial.println("Prints matched!");  // 11
  lcd.setCursor(0, 2);
  lcd.print("Model Cocok!        ");
}
else if (p == FINGERPRINT_PACKETRECIEVEERR) {
```

```
    Serial.println("Communication error");
    return p;
  }
  else if (p == FINGERPRINT_ENROLLMISMATCH) {
    Serial.println("Fingerprints did not match");
    return p;
  }
  else {
    Serial.println("Unknown error");
    return p;
  }

  Serial.print("ID "); Serial.println(id);  // 12

  p = finger.storeModel(id);
  if (p == FINGERPRINT_OK) {
    Serial.println("Stored!");  //13 / end
    lcd.setCursor(0, 2);
    lcd.print("Model ID #");
    lcd.print(id);
    lcd.print(" Disimpan");
    delay(1000);
    return p;
  }
  else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    lcd.print("Communication error!");
    return p;
  }
  else if (p == FINGERPRINT_BADLOCATION) {
    Serial.println("Could not store in that location");
    lcd.print("Could'n save model!  ");
    return p;
  }
  else if (p == FINGERPRINT_FLASHERR) {
    Serial.println("Error writing to flash");
    lcd.print("Error writing flash!");
    return p;
  }
  else {
    Serial.println("Unknown error");
    lcd.print("Unknown error!      ");
    return p;
  }
}

uint8_t deleteFingerprint(uint8_t id)
{
  uint8_t p = -1;

  p = finger.deleteModel(id);

  if (p == FINGERPRINT_OK) {
    Serial.println("Deleted!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
  } else if (p == FINGERPRINT_BADLOCATION) {
    Serial.println("Could not delete in that location");
  } else if (p == FINGERPRINT_FLASHERR) {
    Serial.println("Error writing to flash");
```

```
  } else {
    Serial.print("Unknown error: 0x"); Serial.println(p, HEX);
  }

  return p;
}

uint8_t getFingerprintID()
{
  uint8_t p = finger.getImage();
  switch (p)
  {
    case FINGERPRINT_OK:
      Serial.println("Image taken"); // s2 b2
      break;
    case FINGERPRINT_NOFINGER:
      Serial.println("No finger detected");  // s1 b1
      //return p;
      return (-1); // hasil modif.. normalnya FINGERPRINT_NOFINGER = angka 2
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      return p;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println("Imaging error");
      return p;
    default:
      Serial.println("Unknown error");
      return p;
  }
  Serial.println(p);

  // OK success!

  p = finger.image2Tz();
  switch (p)
  {
    case FINGERPRINT_OK:
      Serial.println("Image converted"); // s3 b3
      break;
    case FINGERPRINT_IMAGEMESS:
      Serial.println("Image too messy");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      return p;
    case FINGERPRINT_FEATUREFAIL:
      Serial.println("Could not find fingerprint features");
      return p;
    case FINGERPRINT_INVALIDIMAGE:
      Serial.println("Could not find fingerprint features");
      return p;
    default:
      Serial.println("Unknown error");
      return p;
  }
  Serial.println(p);

  // OK converted!
  p = finger.fingerFastSearch();
  if (p == FINGERPRINT_OK) {
```

```
    Serial.println("Found a print match!"); // b4
  }
  else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  }
  else if (p == FINGERPRINT_NOTFOUND) {
    Serial.println("Did not find a match"); // s4
    return p;
  }
  else {
    Serial.println("Unknown error");
    return p;
  }
  Serial.println(p);

  // found a match!
  Serial.print("Found ID #"); Serial.print(finger.fingerID);
  Serial.print(" with confidence of "); Serial.println(finger.confidence); // b5

  return finger.fingerID;
}

void setup_fingerprint()
{
  delay(100);

  // set the data rate for the sensor serial port
  finger.begin(57600);

  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1) {
      delay(1);
    }
  }
}
/*
void loop_fingerprint() // run over and over again
{
  Serial.println("Ready to enroll a fingerprint!");
  Serial.println("Please type in the ID # (from 1 to 127) you want to save this finger as...");
  id = readnumber();
  if (id == 0) {// ID #0 not allowed, try again!
    return;
  }
  Serial.print("Enrolling ID #");
  Serial.println(id);

  getFingerprintEnroll();
}
*/
```

KEYPAD 4X4

```
//------------------------------ KEYPAD 4x4 -------------------------------------

#include <Keypad.h>
#include <ctype.h>


const byte ROWS = 4; //four rows
const byte COLS = 4; //three columns
// Define the keymaps.  The blank spot (lower left) is the space character.
char alphaKeys[ROWS][COLS] = {
  { 'A', 'D', 'G', '@' },
  { 'J', 'M', 'P', '$' },
  { 'S', 'V', 'Y', '%' },
  { '*', ' ', '#', '&' }
};

char numberKeys[ROWS][COLS] = {
  { '1', '2', '3', 'A' },
  { '4', '5', '6', 'B' },
  { '7', '8', '9', 'C' },
  { '*', '0', '#', 'D' }
};


byte rowPins[ROWS] = {32, 33, 25, 26}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {27, 14, 12, 13}; //connect to the column pinouts of the keypad

// Create two new keypads, one is a number pad and the other is a letter pad.
Keypad numpad( makeKeymap(numberKeys), rowPins, colPins, sizeof(rowPins), sizeof(colPins) );
Keypad ltrpad( makeKeymap(alphaKeys), rowPins, colPins, sizeof(rowPins), sizeof(colPins) );


uint32_t lastMillis = 0;
String s;

char key;

static char virtKey = NO_KEY;      // Stores the last virtual key press. (Alpha keys only)
static char physKey = NO_KEY;      // Stores the last physical key press. (Alpha keys only)
static char buildStr[12];
static byte buildCount;
static byte pressCount;

static byte kpadState;

boolean alpha = true;   // Start with the numeric keypad.
bool isEnterName = false;
bool isEnrollFinger = false;
bool isDeleteUser = false;
bool isNeedOpenDoor = false;

void swOnState( char key )
{
  switch ( kpadState )
  {
    case PRESSED:
      if(isEnterName == true)
      {
```

```
      if (isalpha(key) == true)              // This is a letter key so we're using the letter keymap.
      {
       if (physKey != key)         // New key so start with the first of 3 characters.
       {
         pressCount = 0;
         virtKey = key;
         physKey = key;
         //Serial.print(virtKey);   // Used for testing.

         s += String(virtKey);

         lastMillis = millis(); // ini untuk timeout
       }
       else            // Pressed the same key again...
       {
         if(millis() - lastMillis > 1000) // timeout ini digunakan jika user ingin menginput di tombol yang sama harus menunggu jeda 700 mili detik sebelum memencept tombol yang sama
         {
           pressCount = 0;
           virtKey = key;
           s += String(virtKey);
         }
         else // jika jeda antar pencet tombol belum sampai timeout
         {
           virtKey++;                  // so select the next character on that key.
           pressCount++;               // Tracks how many times we press the same key.

           if (pressCount > 2 || virtKey > 'z')      // Last character reached atau usdah mencapai huruf 'z' so cycle back to start.
           {
             pressCount = 0;
             virtKey = key;
           }
           //Serial.print(virtKey);   // Used for testing.

           s.remove(s.length()-1);
           s += String(virtKey);
         }

         lastMillis = millis(); // ini untuk timeout
       }

       lcd.setCursor(0,1);
       lcd.print(s);
       Serial.println(s);
      }
      else if (isdigit(key) || key == ' ')
      {
       s += String(key);
       lcd.setCursor(0,1);
       lcd.print(s);
       Serial.println(s);
      }
      else if (key == '*')
      {
       s.remove(s.length()-1); // dihapus
       //s += String(char(keysAlpha[idx]+a)); // isi lagi dengan yang baru
       lcd.setCursor(0,1);
       lcd.print(s);
       lcd.setCursor(s.length(),1);
```

```
    lcd.print(' '); // untuk menghapus carakter terakhir yang didisplay di LCD
    Serial.println(s);
   }
   else if (key == '$' && s != "") //Button Save // jika nama belum diisi belum bisa lanjut sampai diisi
   {
    isEnterName = false;
    isEnrollFinger = true;

    //nama_user[id] = s;
    //s = "";
    /*
    if (idBtn == btnWO){
     WO[bat] = s;
     lcd.print("   WO Saved!!   ");
    }
    */
   }
   else if (key == '%')
   {
    isEnterName = false;
    isEnrollFinger = false;
    lcd.clear();
    s = "";
   }
   else if(key == '&') // Buttn cancel
   {
    // isDeleteUser = false;
    // alpha = true;
   }
   break;
  }

case HOLD:
  if (key == '@')
  {
   isEnterName = true;
   delay(1000);
  }
  if (key == '%') // buton 'C'
  {
   isNeedOpenDoor = true;
   delay(1000);
  }
  if(key == '&') // Button 'D'
  {
   isDeleteUser = true;
   delay(1000);
  }
  if (key == '#')                // Toggle between keymaps.
  {
   if (alpha == true)          // We are currently using a keymap with letters
   {
    alpha = false;           // Now we want a keymap with numbers.
    //digitalWrite(ledPin, LOW);
   }
   else
   {                     // We are currently using a keymap with numbers
    alpha = true;            // Now we want a keymap with letters.
   }
  }
```

```
    else if(key == '*')
    {
     lcd.clear();
     lcd.setCursor(0,0);
     lcd.print(" DATA ALL CLEAR ");
     Serial.println("\n\n\t\tDATA ALL CLEAR\n\n");

     delay(1000);
     lcd.clear();
/*
     for(bat=1; bat<=n_bat; bat++){ // 14 baterai
      WO[bat] = "";
      PN[bat] = "";
      SN[bat] = "";
     }
     */
    }
    else                    // Some key other than '#' was pressed.
    {
     buildStr[buildCount++] = (isalpha(key)) ? virtKey : key;
     buildStr[buildCount] = '\0';
     Serial.println();
     Serial.println(buildStr);
    }
    break;

   case RELEASED:
    if (buildCount >= sizeof(buildStr))  buildCount = 0;  // Our string is full. Start fresh.
    break;
  }  // end switch-case
}// end switch on state function


// Take care of some special events.

void keypadEvent_ltr(KeypadEvent key)
{
  // in here when in alpha mode
  kpadState = ltrpad.getState( );
  swOnState( key );
} // end ltrs keypad events

void keypadEvent_num( KeypadEvent key )
{
  // in here when using number keypad
  kpadState = numpad.getState( );
  swOnState( key );
} // end numbers keypad events

void setup_keypad4x4()
{
  //pinMode(ledPin, OUTPUT);
  //digitalWrite(ledPin, LOW);              // Turns the LED on.
  ltrpad.begin( makeKeymap(alphaKeys) );
  numpad.begin( makeKeymap(numberKeys) );
  ltrpad.addEventListener(keypadEvent_ltr);  // Add an event listener.
  ltrpad.setHoldTime(1000);               // Default is 1000mS
  numpad.addEventListener(keypadEvent_num);  // Add an event listener.
  numpad.setHoldTime(1000);               // Default is 1000mS
}
```

```
void loop_keypad()
{
  if ( alpha == true )
    key = ltrpad.getKey( );
  else
    key = numpad.getKey( );
}
```

LCD 20X4

```
//-------------------- LCD 20x4 ------------------------

#include <LiquidCrystal_I2C.h>
// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 20, 4);


void setup_lcd16x2()
{
  // initialize the LCD
  lcd.begin();
  // Turn on the blacklight and print a message.
  lcd.backlight();
}
```

RTC DS1307

```
#include <Wire.h>
#include "RTClib.h"

RTC_DS1307 rtc;

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

  if (! rtc.begin())
  {
    Serial.println("Couldn't find RTC");
    while (1);
  }

  if (!rtc.isrunning())
  {
//    Serial.println("RTC lost power, lets set the time!");

  // Comment out below lines once you set the date & time.
    // Following line sets the RTC to the date & time this sketch was compiled
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));

    // Following line sets the RTC with an explicit date & time
    // for example to set January 27 2017 at 12:56 you would call:
    // rtc.adjust(DateTime(2021, 6, 28, 5, 53, 26));
  }
```

```
  // seuaikan tangal dan jam sepertidi laptop. upload ketika detik 00
  // rtc.adjust(DateTime(2021, 7, 10, 20, 16, 22));
}
```

PEMBACAAN NOMINAL UANG KERTAS

```
//---------------------------- Sensor TCS DTEKSI WARNA UANG ---------------------------------------//

#include <Wire.h>
#include "Adafruit_TCS34725.h"

Adafruit_TCS34725 tcs = Adafruit_TCS34725(TCS34725_INTEGRATIONTIME_120MS, TCS34725_GAIN_16X);


// R: 70.87 G: 100.33 B: 78.65  mulus
const float R_uang100rb = 70.87;
const float G_uang100rb = 100.33;
const float B_uang100rb = 78.65;

// R: 62.07 G: 104.22 B: 83.35  mulus
// R: 62.63 G: 104.01 B: 83.04  agak lecek dan agak kusem
const float R_uang50rb = 62.07;
const float G_uang50rb = 104.22;
const float B_uang50rb = 83.35;


// R: 64.99  G: 106.86 B: 77.90 mulus
// R: 65.16  G: 106.53 B: 77.54 lecek
// R: 65.07  G: 106.78 B: 77.57 lecek banget
const float R_uang20rb = 65.07;
const float G_uang20rb = 106.78;
const float B_uang20rb = 77.57;

// R: 65.35 G: 100.86 B: 83.11  mulus
// R: 66.23 G: 101.79 B: 80.88  kusut lecek dan kusem banget
const float R_uang10rb = 65.85;
const float G_uang10rb = 101.20;
const float B_uang10rb = 82.00;

// R: 71.11 G: 103.95 B: 74.51  mulus
// R: 71.72 G: 103.46 B: 74.19  agak lecek
// R: 71.34 G: 103.15 B: 74.64  lecek banget dan kusem
const float R_uang5rb = 71.72;
const float G_uang5rb = 103.46;
const float B_uang5rb = 74.19;

// R: 65.18  G: 104.25 B: 79.78  mulus banget
// R: 65.46  G: 104.29 B: 79.15  lecek
// R: 66.48  G: 104.19 B: 78.14  agak lecek dan kusem
// R: 67.07  G: 103.99 B: 77.62  lecek banget dan kusem banget
const float R_uang2rb = 66.48;
const float G_uang2rb = 104.19;
const float B_uang2rb = 78.14;

// R: 68.63 G: 105.31 B: 75.36  agak lecek dan tidak kusem
// R: 69.06 G: 105.30 B: 74.78  lecek dan kusem
const float R_uang1rb = 69.06;
const float G_uang1rb = 105.30;
```

```
const float B_uang1rb = 74.78;

#define uang_tidak_terdeteksi  0
#define uang100rb          1
#define uang50rb           2
#define uang20rb           3
#define uang10rb           4
#define uang5rb            5
#define uang2rb            6
#define uang1rb            7

const unsigned long daftar_uang[8] = {0, 100000, 50000, 20000, 10000, 5000, 2000, 1000};


// ----------------- FILTER RUNNING MEDIAN ----------------------------

#include <RunningMedian.h>
RunningMedian samples_R = RunningMedian(5);
RunningMedian samples_G = RunningMedian(5);
RunningMedian samples_B = RunningMedian(5);


void setup_tcs34725()
{
 if (!tcs.begin())
 {
   Serial.println("No TCS34725 found ... check your connections");
   while (1); // halt!
 }
}

int scan_uang_kertas()
{
 float R, G, B;

 for (int i = 0; i < 5; i++)
 {
   tcs.getRGB(&R, &G, &B);

   samples_R.add(R);
   samples_G.add(G);
   samples_B.add(B);

   delay(10);
 }

 R = samples_R.getMedian();
 G = samples_G.getMedian();
 B = samples_B.getMedian();

 Serial.print("R: "); Serial.print(R);
 Serial.print("\tG: "); Serial.print(G);
 Serial.print("\tB: "); Serial.print(B);

 if ( R > (R_uang100rb - 2) && R < (R_uang100rb + 2) &&
      G > (G_uang100rb - 2) && G < (G_uang100rb + 2) &&
      B > (B_uang100rb - 2) && B < (B_uang100rb + 2) )
 {
   Serial.println("\t Uang 100rb");
```

```
  return uang100rb;
 }
 else if ( R > (R_uang50rb - 2) && R < (R_uang50rb + 2) &&
         G > (G_uang50rb - 2) && G < (G_uang50rb + 2) &&
         B > (B_uang50rb - 2) && B < (B_uang50rb + 2) )
 {
  Serial.println("\t Uang 50rb");
  return uang50rb;
 }
 else if ( R > (R_uang20rb - 2) && R < (R_uang20rb + 2) &&
         G > (G_uang20rb - 2) && G < (G_uang20rb + 2) &&
         B > (B_uang20rb - 2) && B < (B_uang20rb + 2) )
 {
  Serial.println("\t Uang 20rb");
  return uang20rb;
 }
 else if ( R > (R_uang10rb - 2) && R < (R_uang10rb + 2) &&
         G > (G_uang10rb - 2) && G < (G_uang10rb + 2) &&
         B > (B_uang10rb - 2) && B < (B_uang10rb + 2) )
 {
  Serial.println("\t Uang 10rb");
  return uang10rb;
 }
 else if ( R > (R_uang5rb - 2) && R < (R_uang5rb + 2) &&
         G > (G_uang5rb - 2) && G < (G_uang5rb + 2) &&
         B > (B_uang5rb - 2) && B < (B_uang5rb + 2) )
 {
  Serial.println("\t Uang 5rb");
  return uang5rb;
 }
 else if ( R > (R_uang2rb - 2) && R < (R_uang2rb + 2) &&
         G > (G_uang2rb - 2) && G < (G_uang2rb + 2) &&
         B > (B_uang2rb - 2) && B < (B_uang2rb + 2) )
 {
  Serial.println("\t Uang 2rb");
  return uang2rb;
 }
 else if ( R > (R_uang1rb - 2) && R < (R_uang1rb + 2) &&
         G > (G_uang1rb - 2) && G < (G_uang1rb + 2) &&
         B > (B_uang1rb - 2) && B < (B_uang1rb + 2) )
 {
  Serial.println("\t Uang 1rb");
  return uang1rb;
 }
 else
 {
  Serial.println("\t Uang Tidak Terdeteksi!");
  return uang_tidak_terdeteksi;
 }

}
```

LAMPIRAN 2
(DATASHEET)

1. ESP32

1. Overview

ESP32 is a single 2.4 GHz Wi-Fi-and-Bluetooth combo chip designed with the TSMC ultra-low-power 40 nm technology. It is designed to achieve the best power and RF performance, showing robustness, versatility and reliability in a wide variety of applications and power scenarios.

The ESP32 series of chips includes ESP32-D0WDQ6, ESP32-D0WD, ESP32-D2WD, and ESP32-S0WD. For details on part numbers and ordering information, please refer to Part Number and Ordering Information.

1.1 Featured Solutions

1.1.1 Ultra-Low-Power Solution

ESP32 is designed for mobile, wearable electronics, and Internet-of-Things (IoT) applications. It features all the state-of-the-art characteristics of low-power chips, including fine-grained clock gating, multiple power modes, and dynamic power scaling. For instance, in a low-power IoT sensor hub application scenario, ESP32 is woken up periodically and only when a specified condition is detected. Low-duty cycle is used to minimize the amount of energy that the chip expends. The output of the power amplifier is also adjustable, thus contributing to an optimal trade-off between communication range, data rate and power consumption.

> **Note:**
> For more information, refer to Section 3.7 RTC and Low-Power Management.

1.4.2 Clocks and Timers

- Internal 8 MHz oscillator with calibration
- Internal RC oscillator with calibration
- External 2 MHz ~ 60 MHz crystal oscillator (40 MHz only for Wi-Fi/BT functionality)
- External 32 kHz crystal oscillator for RTC with calibration
- Two timer groups, including 2 x 64-bit timers and 1 x main watchdog in each group
- One RTC timer
- RTC watchdog

1.4.3 Advanced Peripheral Interfaces

- 34 x programmable GPIOs
- 12-bit SAR ADC up to 18 channels
- 2 x 8-bit DAC
- 10 x touch sensors
- 4 x SPI
- 2 x I2S
- 2 x I2C
- 3 x UART
- 1 host (SD/eMMC/SDIO)
- 1 slave (SDIO/SPI)
- Ethernet MAC interface with dedicated DMA and IEEE 1588 support
- CAN 2.0
- IR (TX/RX)
- Motor PWM
- LED PWM up to 16 channels
- Hall sensor

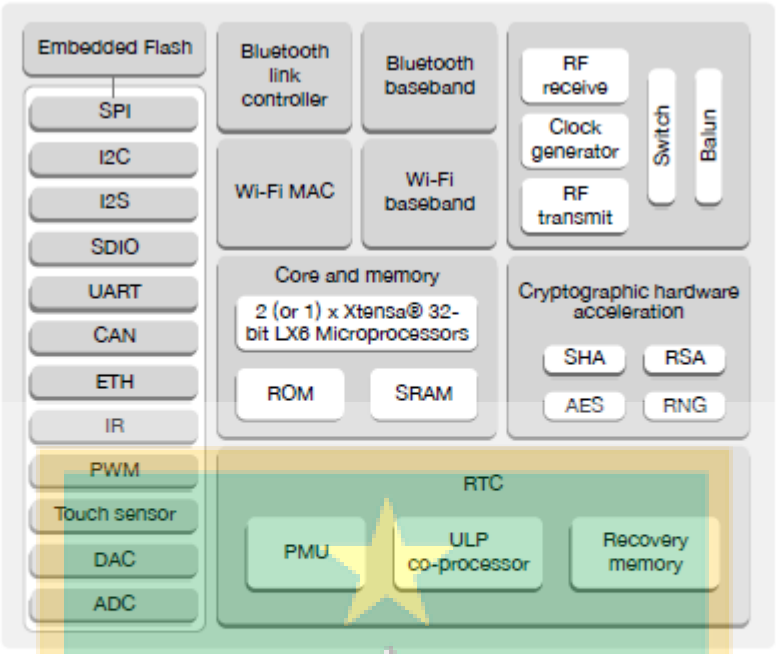


Figure 1: Functional Block Diagram

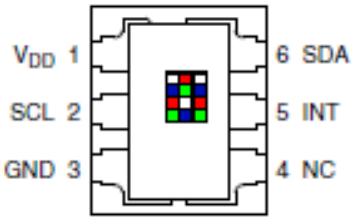
Note:
Products in the ESP32 series differ from each other in terms of their support for embedded flash and the number of CPUs they have. For details, please refer to Part Number and Ordering Information.

2. Sensor TCS34725

Features

- Red, Green, Blue (RGB), and Clear Light Sensing with IR Blocking Filter
  - Programmable Analog Gain and Integration Time
  - 3,800,000:1 Dynamic Range
  - Very High Sensitivity — Ideally Suited for Operation Behind Dark Glass
- Maskable Interrupt
  - Programmable Upper and Lower Thresholds with Persistence Filter
- Power Management
  - Low Power — 2.5-μA Sleep State
  - 65-μA Wait State with Programmable Wait State Time from 2.4 ms to > 7 Seconds
- I²C Fast Mode Compatible Interface
  - Data Rates up to 400 kbit/s
  - Input Voltage Levels Compatible with V_{DD} or 1.8 V Bus
- Register Set and Pin Compatible with the TCS3x71 Series
- Small 2 mm × 2.4 mm Dual Flat No-Lead (FN) Package

PACKAGE FN
DUAL FLAT NO-LEAD
(TOP VIEW)



Package Drawing Not to Scale

Applications

- RGB LED Backlight Control
- Light Color Temperature Measurement
- Ambient Light Sensing for Display Backlight Control
- Fluid and Gas Analysis
- Product Color Verification and Sorting

End Products and Market Segments

- TVs, Mobile Handsets, Tablets, Computers, and Monitors
- Consumer and Commercial Printing
- Medical and Health Fitness
- Solid State Lighting (SSL) and Digital Signage
- Industrial Automation

Description

The TCS3472 device provides a digital return of red, green, blue (RGB), and clear light sensing values. An IR blocking filter, integrated on-chip and localized to the color sensing photodiodes, minimizes the IR spectral component of the incoming light and allows color measurements to be made accurately. The high sensitivity, wide dynamic range, and IR blocking filter make the TCS3472 an ideal color sensor solution for use under varying lighting conditions and through attenuating materials.

The TCS3472 color sensor has a wide range of applications including RGB LED backlight control, solid-state lighting, health/fitness products, industrial process controls and medical diagnostic equipment. In addition, the IR blocking filter enables the TCS3472 to perform ambient light sensing (ALS). Ambient light sensing is widely used in display-based products such as cell phones, notebooks, and TVs to sense the lighting environment and enable automatic display brightness for optimal viewing and power savings. The TCS3472, itself, can enter a lower-power wait state between light sensing measurements to further reduce the average power consumption.

3. Sensor ZFM-20

Power	DC 3.6V-6.0V	**Interface**	UART(TTL logical level)/ USB 1.1
Working current	Typical: 100mA Peak: 150mA	**Matching Mode**	1:1 and 1:N
Baud rate	(9600*N)bps, N=1～12 (default N=6)	**Character file size**	256 bytes
Image acquiring time	<1s	**Template size**	512 bytes
Storage capacity	120/ 375/ 880	**Security level**	5 (1, 2, 3, 4, 5(highest))
FAR	<0.001%	FRR	<0.1%
Average searching time	< 1s (1:880)	**Window dimension**	14mm*18mm
Working environment	Temp: -10℃- +40℃	**Storage environment**	Temp: -40℃- +85℃
	RH: 40%-85%		RH: <85%
Outline Dimention	Split type	Module: 42*25*8.5mm (install dimension: 31.5*19mm) Sensor:56*20*21.5mm	
	Integral type	56*20*21.5mm	

1. Power supply

Item	Parameter			Unit	Note
	Min	Typ	Max		
Power Voltage (Vin)	3.6		6.0	V	Normal working value.
Maximum Voltage (Vin_{max})	−0.3		7.0	V	Exceeding the Maximum rating may cause permant harm to the Module.
Operation Current (Icc)	90	100	110	mA	
Peak Current (Ipeak)			150	mA	

2. TD（output, TTL logic level）

Item	Condition	Parameter			Unit	Note
		Min	Typ	Max		
V_{OL}	I_{OL}=−4mA			0.4	V	Logic 0
V_{OH}	I_{OH}= 4mA	2.4		3.3	V	Logic 1

3. RD（input, TTL logic level）

Item	Condition	Parameter			Unit	Note
		Min	Typ	Max		
V_{IL}				0.6	V	Loige 0
V_{IH}		2.4			V	Logic 1
I_{IH}	V_{IH} =5V		1		mA	
	V_{IH} =3.3V		30		uA	
V_{Imax}		−0.3		5.5	V	**Maximum input voltage**

3.1.2 USB communication

When it;s USB communication, definition of J1 is:

Pin Num	Name	Type	Function Description
1	Vin	in	Power supply input（refer to 3.1.1.4 for electrical parameter）
2	DP+	In/Out	USB data。
3	DP-	In/Out	USB data。
4	GND	—	Signal ground. Connected to power ground.
5	END	—	Earth Ground. Float or connect to the shield layer of cable. (doesnj t exist with Integral type)