

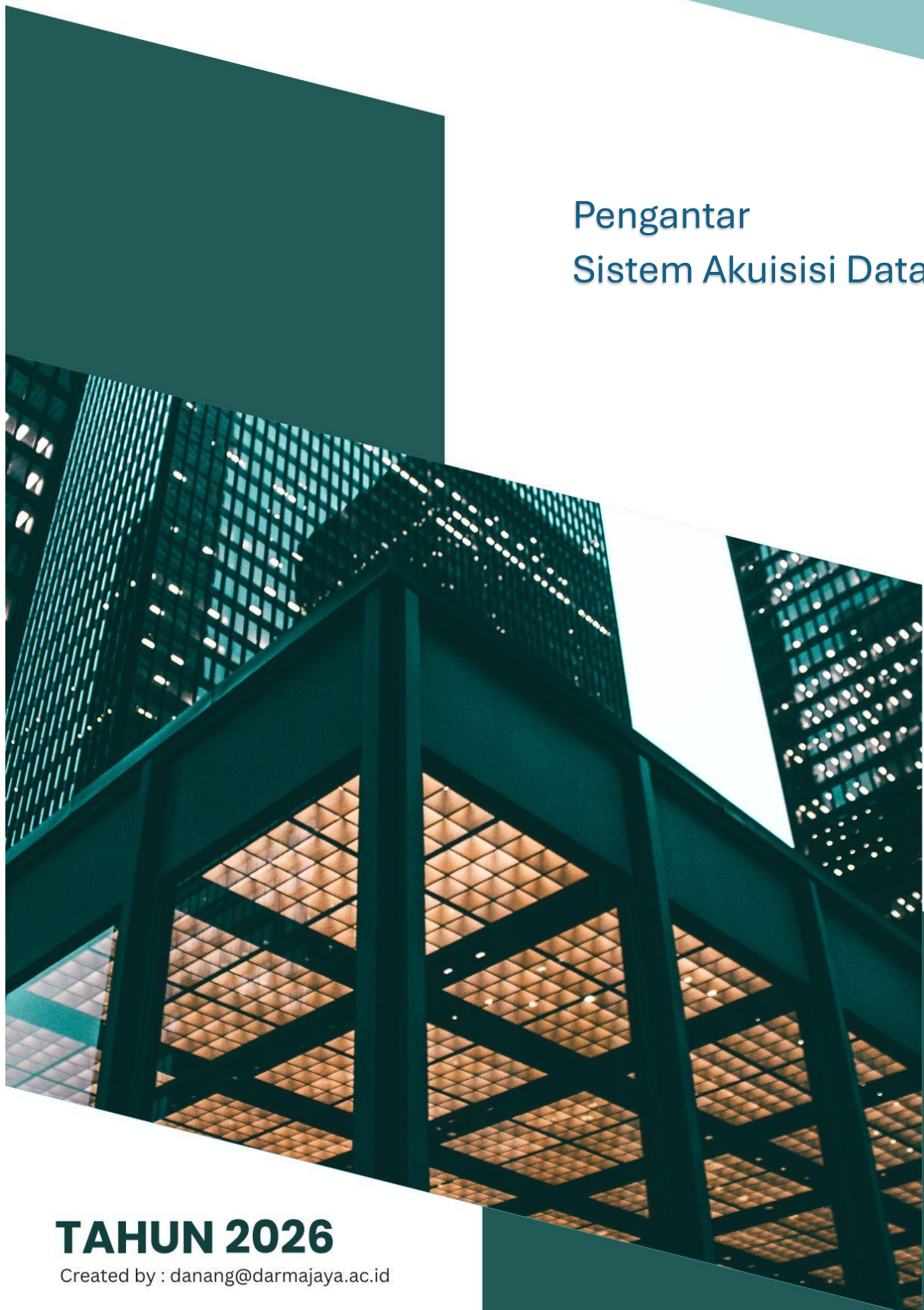
SISTEM AKUISISI DATA



SISTEM KOMPUTER
DARMAJAYA



Pengantar
Sistem Akuisisi Data



TAHUN 2026

Created by : danang@darmajaya.ac.id

DAFTAR ISI

DAFTAR ISI	2
MODUL 2: KONSEP DASAR SENSOR, TRANSDUSER, DAN PRAKTIKUM AKUISISI SINYAL ANALOG	Error! Bookmark not defined.
A. INFORMASI UMUM.....	Error! Bookmark not defined.
B. Capaian Pembelajaran (CPMK)	Error! Bookmark not defined.
C. BAGIAN 1: MATERI TEORI (PERTEMUAN 3).....	Error! Bookmark not defined.
1. Pengertian Sensor dan Transduser	Error! Bookmark not defined.
2. Klasifikasi Sensor	Error! Bookmark not defined.
3. Karakteristik Kinerja Sensor (Parameter Statis) ..	Error! Bookmark not defined.
4. Konsep Pengkondisian Sinyal Dasar: Pembagi Tegangan (Voltage Divider)	Error! Bookmark not defined.
Bookmark not defined.	
BAGIAN 2: MATERI PRAKTIKUM (PERTEMUAN 4)	Error! Bookmark not defined.
1. Tujuan Instruksional Praktikum	Error! Bookmark not defined.
2. Alat dan Bahan Laboratorium	Error! Bookmark not defined.
3. Topologi dan Skema Rangkaian (Wiring Diagram)	Error! Bookmark not defined.
4. Eksperimen Kode Program 1: Membaca Resolusi ADC (Potensiometer)...	Error! Bookmark not defined.
Bookmark not defined.	
5. Eksperimen Kode Program 2: Pemrosesan Konversi Tegangan (LDR)	Error! Bookmark not defined.
Bookmark not defined.	
D. EVALUASI DAN TUGAS LAPORAN PRAKTIKUM 2	Error! Bookmark not defined.

C. BAGIAN 2: MATERI PRAKTIKUM (PERTEMUAN 12)

2.1. Tujuan Instruksional Praktikum

1. Mahasiswa menguasai pemodelan *firmware* C/C++ non-blokir (*non-blocking*) menggunakan fasilitas *Timer Interrupt* ESP32.
2. Mahasiswa mampu melakukan kalkulasi matematis *Prescaler* dan *Alarm Value* untuk mencapai frekuensi sampling (f_s) yang presisi hingga resolusi mikrodetik.
3. Mahasiswa mampu merancang arsitektur perangkat lunak modular yang memisahkan tugas kritis (ISR) dengan tugas berat (*Main Loop*).

2.2. Kebutuhan Alat dan Bahan

Tabel 2. Rincian Komponen Praktikum

No	Nama Komponen	Deskripsi dan Spesifikasi	Jumlah
1	Workstation	PC/Laptop dengan OS Windows/Linux, Arduino IDE 2.x terinstal.	1 Unit
2	Development Board	Mikrokontroler ESP32 DOIT DevKit V1 (30/38 pin)	1 Buah
3	Kabel Data	Kabel USB Type-C atau Micro-USB yang terbukti mendukung transmisi data.	1 Buah
4	Breadboard	Papan sirkuit rakitan (Project Board) 400 atau 830 lubang.	1 Buah
5	Potensiometer	Resistor variabel mekanik 10k Ω atau 50k Ω sebagai simulator.	1 Buah
6	Kabel Jumper	Konfigurasi <i>Male-to-Male</i> (M-M) dan <i>Male-to-Female</i> (M-F).	1 Set

2.3. Topologi Perangkat Keras dan Diagram Pengkabelan (Wiring)

Kita akan mempertahankan instrumen sensor analog simulasi (Potensiometer) untuk menguji validitas *sampling* data. Potensiometer dipilih karena sifat tegangannya yang linier murni, memudahkan kita menganalisis keteraturan log waktu (*timestamp*).

[TEMPATKAN GAMBAR 3 DI SINI: SKEMATIK WIRING FRITZING]

Deskripsi Gambar untuk Dosen: Sisipkan gambar rakitan grafis dari aplikasi Fritzing.

- Kabel Merah: Menghubungkan pin 3.3V ESP32 ke terminal kiri potensiometer.
- Kabel Hitam: Menghubungkan pin GND ESP32 ke terminal kanan potensiometer.
- Kabel Kuning/Hijau: Menghubungkan terminal tengah (*wiper*) potensiometer ke **Pin 34** (ADC1_CH6) pada ESP32.

2.4. Eksperimen 1: Dasar Pemrograman Timer Interrupt (Blink Tanpa Delay)

Sebagai pemanasan arsitektural, kita akan mengedipkan lampu LED Biru bawaan ESP32 secara persis setiap 1 detik (1000 milidetik), tetapi sama sekali tidak menggunakan perintah `delay()` di fungsi utama.

Tulis ulang sintaks *firmware* berikut di kanvas Arduino IDE secara presisi:

```
C++
```

```
/*
```

```
* MODUL 6: EKSPERIMEN 1 - DASAR HARDWARE TIMER INTERRUPT
```

```
* Deskripsi: Mengedipkan LED internal dengan presisi absolut menggunakan Timer 0.
```

```
*      Membuktikan bahwa fungsi loop() sama sekali tidak terbebani.
```

```
*/
```

```
// Mendefinisikan pin LED bawaan ESP32 (Umumnya Pin 2)
```

```
const int ledPin = 2;
```

```
// Deklarasi pointer untuk objek Timer perangkat keras ESP32
```

```
hw_timer_t * timer = NULL;
```

```
// Variabel status LED. Wajib menggunakan 'volatile' karena diubah dalam ISR
```

```
volatile bool statusLED = false;
```

```
// -----
```

```
// FUNGSI INTERRUPT SERVICE ROUTINE (ISR)
```

```

// Atribut IRAM_ATTR memaksa fungsi ini disimpan di RAM (bukan Flash)
// agar eksekusinya jauh lebih cepat.
// -----
void IRAM_ATTR onTimer() {
    // Baris logika pembalik (Toggle)
    statusLED = !statusLED;
    // Mengeksekusi perubahan tegangan fisis ke komponen LED
    digitalWrite(ledPin, statusLED);
}

void setup() {
    Serial.begin(115200);
    pinMode(ledPin, OUTPUT);
    Serial.println("Memulai Inisialisasi Hardware Timer 0...");

    // LANGKAH 1: Inisialisasi Timer
    // Parameter: (Nomor Timer [0-3], Nilai Prescaler, Menghitung Naik [true/false])
    // Base Clock 80 MHz dibagi prescaler 80 menghasilkan kecepatan 1 MHz (1 tick = 1
mikrodetik).
    timer = timerBegin(0, 80, true);

    // LANGKAH 2: Mengaitkan Timer dengan Fungsi ISR
    // Parameter: (Objek Timer, Alamat Fungsi ISR, Tipe Interrupt [Edge/Level])
    timerAttachInterrupt(timer, &onTimer, true);

    // LANGKAH 3: Mengatur Batas Waktu (Alarm)
    // Nilai 1.000.000 mikrodetik = 1 Detik.
    // Parameter akhir 'true' berarti Auto-Reload (akan terus berulang otomatis).
    timerAlarmWrite(timer, 1000000, true);

    // LANGKAH 4: Menyalakan Mesin Alarm
    timerAlarmEnable(timer);

```

```

Serial.println("Sistem Timer Aktif. CPU Bebas Tugas di Loop.");
}

void loop() {
    // FUNGSI UTAMA KOSONG MUTLAK.
    // Eksekusi kedipan lampu berjalan secara mandiri (Asynchronous)
    // dipicu oleh detak komponen keras Quartz Crystal di dalam ESP32.
}

```

Analisis Hasil Eksperimen 1:

1. Klik *Upload* dan perhatikan papan mikrokontroler Anda. LED akan berkedip sangat stabil.
2. Karena blok `void loop()` sama sekali kosong, ini membuktikan secara empiris bahwa interupsi berjalan independen di latar belakang tanpa campur tangan urutan eksekusi perangkat lunak baris-demi-baris konvensional.

2.5. Eksperimen 2: Akuisisi Data ADC Real-Time dengan Arsitektur Penguncian Memori (Atomic Block)

Pada eksperimen ini, kita memadukan ilmu instrumentasi (membaca sensor analog) dengan ilmu sistem operasi tertanam (sinkronisasi memori).

Skenario: Kita membutuhkan data tekanan (disimulasikan oleh potensiometer) tepat setiap 20 milidetik (Frekuensi Sampling 50 Hz). Data dibaca secara kilat oleh ISR, disimpan ke memori *buffer*, lalu di fungsi `loop()`, CPU akan memindahkan data tersebut ke *Serial Monitor* secara aman tanpa merusak integritas memori.

Buat lembar proyek (Sketch) baru dan tulis kode industri berikut:

```

C++

/*
 * MODUL 6: EKSPERIMEN 2 - AKUISISI SENSOR SINKRON & MANAJEMEN MEMORI
 * Deskripsi: Sampling periodik 50 Hz dengan pola arsitektur Atomic Block.
 */

const int pinAnalog = 34;

```

```

// Alokasi memori untuk objek Hardware Timer (Menggunakan Timer 1)
hw_timer_t * timerADC = NULL;

// -----
// VARIABEL GLOBAL BERBAGI (SHARED VARIABLES)
// Digunakan oleh ISR dan Loop. Wajib "volatile".
// -----

volatile int bufferSensor = 0;
volatile unsigned long bufferWaktu = 0;
volatile bool flagDataBaru = false; // Bendera status penanda ketersediaan data

// -----
// FUNGSI ISR: AKUISISI KILAT
// Dilarang keras memakai delay() atau Serial.print() di sini!
// -----

void IRAM_ATTR isrAkuisisiSensor() {
    // 1. Ambil cap waktu (Timestamp)
    bufferWaktu = millis();

    // 2. Baca tegangan analog (Kuantisasi ADC)
    bufferSensor = analogRead(pinAnalog);

    // 3. Kibarkan bendera agar sistem utama memproses data
    flagDataBaru = true;
}

void setup() {
    Serial.begin(115200);
    delay(1000);
    Serial.println("=== SISTEM AKUISISI DAQ 50 HZ DIAKTIFKAN ===");

    // Setup Hardware Timer (Menggunakan Timer 1 agar tidak konflik dengan Timer 0)

```

```

timerADC = timerBegin(1, 80, true);
timerAttachInterrupt(timerADC, &isrAkuisisiSensor, true);

// Kalkulasi Waktu: 20 milidetik = 20.000 mikrodetik
timerAlarmWrite(timerADC, 20000, true);
timerAlarmEnable(timerADC);
}

void loop() {
    // Pengecekan status bendera secara kontinu (Polling Status)
    if (flagDataBaru == true) {

        // -----
        // BLOK ATOMIK (ATOMIC BLOCK) - MANAJEMEN MEMORI AMAN
        // Membekukan interupsi sesaat untuk memindahkan data volatil
        // ke variabel memori lokal agar tidak tertimpa oleh ISR berikutnya.
        // -----
        noInterrupts(); // Matikan sensor interupsi

        // Salin (Copy) data
        int dataAman = bufferSensor;
        unsigned long waktuAman = bufferWaktu;

        // Turunkan bendera
        flagDataBaru = false;

        interrupts(); // Hidupkan interupsi kembali secepat mungkin!
        // Akhir Blok Atomik

        // -----
        // EKSEKUSI TUGAS BERAT (HEAVY TASK)
        // Area ini bebas blokir, aman untuk mengirim data ke Jaringan WiFi,

```

```

// Database Server, atau menyimpan ke Modul MicroSD Card.
// -----
Serial.print("Timestamp: ");
Serial.print(waktuAman);
Serial.print(" ms \t| Data Resolusi: ");
Serial.println(dataAman);
}

// Sisa waktu di fungsi loop() sangat leluasa dan hemat komputasi (Low-Power).
}

```

Instruksi Analisis Data:

1. Lakukan *Upload* dan buka layar *Serial Monitor*.
2. Anda akan melihat parameter Timestamp meningkat dengan selisih absolut kelipatan **20** (Misal: 1020 ms, 1040 ms, 1060 ms, dst).
3. Anda telah membuktikan kemampuan menciptakan sistem DAQ yang sangat presisi tanpa satu pun baris *delay* yang mematikan kinerja sistem, memenuhi kaidah perancangan perangkat keras yang beretika, andal, dan teruji.

D. EVALUASI FORMATIF DAN TUGAS LAPORAN PRAKTIKUM 6

Untuk mengevaluasi kinerja desain *embedded system* yang terintegrasi, mahasiswa wajib menganalisis eksperimen di atas dalam bentuk Laporan Praktikum. Format wajib mengacu pada standar kelulusan CPL-Sistem Komputer Institut Informatika dan Bisnis Darmajaya.

Soal Analisis Analitik dan Algoritmik (Individu)

1. **Investigasi Kritis Keterbatasan Waktu ISR (Bobot 25%):** Dalam kurikulum *Best Practice* industri DAQ, mahasiswa dilarang menaruh fungsi `Serial.print()` di dalam blok `void IRAM_ATTR isrAkuisisiSensor()`. Lakukan kalkulasi matematis sederhana: Jika kecepatan Serial adalah 9600 baud (sekitar 1 milidetik untuk mengirim 1 byte/karakter), berapa lama waktu (dalam milidetik) yang dibutuhkan CPU untuk mengirim string *"Data Tersimpan: 4095"* (total 21 karakter)? Apa dampaknya jika *Alarm Timer* disetel pada interval 10 milidetik (100 Hz) namun fungsi di dalam ISR membutuhkan waktu komputasi yang lebih lama dari 10 milidetik? (Jelaskan fenomena *ISR Starvation/Crash*).

2. Kalkulasi Desain Hardware Timer (Bobot 25%):

Sebuah *vendor* perkebunan cerdas meminta Anda merancang akuisisi *sensor node* (menggunakan ESP32 Base Clock 80 MHz). Karena fenomena kelembaban tanah sangat lambat, mereka hanya butuh *sampling* setiap 0.5 detik (setengah detik). Jika Anda menetapkan parameter prescaler sebesar **8000**, hitunglah berapa kecepatan 1 tick *timer* tersebut (dalam satuan milidetik)! Dan berapakah nilai *Alarm* yang harus disisipkan pada argumen fungsi `timerAlarmWrite(timer, NILAI, true)` agar sistem memicu interupsi tepat setiap 0.5 detik?

3. Dekonstruksi Fenomena Race Condition (Bobot 30%):

Pada Eksperimen 2, terdapat mekanisme sintaks `noInterrupts()` dan `interrupts()`. Jelaskan dengan skenario analogi yang detail, apa itu fenomena **Race Condition** pada memori mikrokontroler. Apa yang akan terjadi pada struktur keutuhan data (misal variabel tipe data *long* berukuran 4 byte) apabila CPU sedang menyalin byte ke-1 dan byte ke-2 di memori lokal, namun tiba-tiba interupsi perangkat keras menyerobot masuk dan memperbarui keseluruhan isi 4 byte tersebut?

4. Pengembangan Skema Multi-Sensor Asinkron (Bobot 20%):

Sistem ESP32 memiliki 4 buah penanda waktu perangkat keras (*Timer 0, 1, 2, 3*). Jelaskan secara konseptual (tanpa perlu menuliskan kode utuh), bagaimana strategi arsitektur perangkat keras dan perangkat lunak jika Anda diwajibkan membaca 2 sensor yang memiliki karakteristik frekuensi berbeda:

- Sensor A (Getaran, butuh *sampling* sangat cepat: setiap 5 ms)
- Sensor B (Suhu ambien, butuh *sampling* sangat lambat: setiap 2000 ms)
- Keduanya harus dikerjakan dalam satu papan ESP32 tanpa *blocking* satu sama lain!

Standar Rubrik Penilaian Portofolio Laporan

Tabel 3. Matriks Penilaian Ketercapaian CPMK104 & CPMK146

Kriteria Komponen Penilaian	Skala Penilaian Sempurna (Nilai Maksimum)
Keterampilan Faktual & Reproduksi Kode	(20 Poin) Kode dieksekusi dengan sempurna, menyertakan <i>screenshot</i> terminal Serial Monitor yang memperlihatkan laju pewaktuan (Timestamp) secara linier dan konsisten.
Akurasi Perhitungan Arsitektur Timer	(30 Poin) Kalkulasi matematis pembagi waktu (<i>Prescaler</i> dan konversi Mikrodetik/Milidetik) diturunkan secara akurat, logika <i>Tick</i> digambarkan terstruktur sesuai satuan SI.
Ketajaman Pemecahan Masalah (<i>Debugging</i>)	(30 Poin) Kemampuan menjelaskan keandalan sistem terkait fenomena korupsi data (<i>Race Condition</i>) diulas mendalam, analogi mudah dipahami secara logika keilmuan komputer.
Argumentasi Desain Sistem Fleksibel	(20 Poin) Usulan pemisahan objek <i>Hardware Timer</i> (Timer 0 dan Timer 1) untuk penjadwalan multi-sensor yang asinkron direpresentasikan dengan konsep yang logis dan solid.