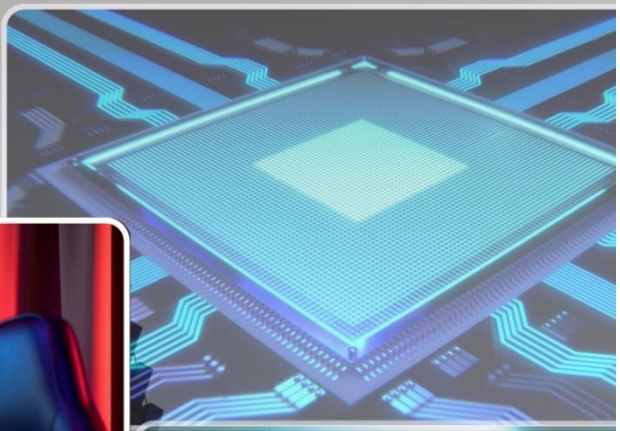
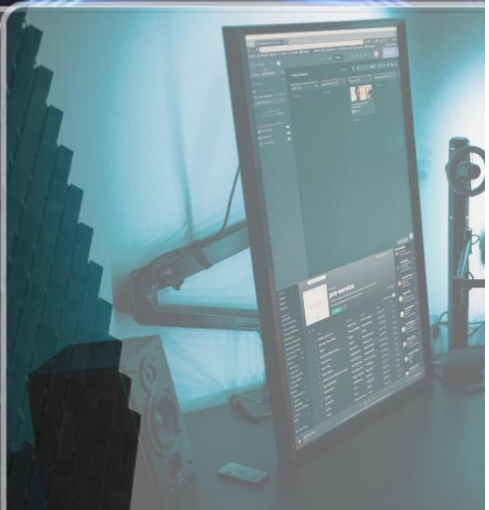
ARSITEKTUR KOMPUTER FONDASI TEKNOLOGI DIGITAL

PENULIS
RIAN TONI RAMBE, DECI IRMAYANI, VOLVO SIHOMBING





Arsitektur Komputer
Fondasi Teknologi Digital

PENERBIT
YAYASAN MMI

UU No 28 tahun 2014 tentang Hak Cipta

Fungsi dan sifat hak cipta Pasal 4

Hak Cipta sebagaimana dimaksud dalam Pasal 3 huruf a merupakan hak eksklusif yang terdiri atas hak moral dan hak ekonomi.

Pembatasan Pelindungan Pasal 26

Ketentuan sebagaimana dimaksud dalam Pasal 23, Pasal 24, dan Pasal 25 tidak berlaku terhadap:

i. penggunaan kutipan singkat Ciptaan dan/atau produk Hak Terkait untuk pelaporan peristiwa aktual yang ditujukan hanya untuk keperluan penyediaan informasi aktual;
ii. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk kepentingan penelitian ilmu pengetahuan;
iii. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk keperluan pengajaran, kecuali pertunjukan dan Fonogram yang telah dilakukan Pengumuman sebagai bahan ajar; dan
iv. penggunaan untuk kepentingan pendidikan dan pengembangan ilmu pengetahuan yang memungkinkan suatu Ciptaan dan/atau produk Hak Terkait dapat digunakan tanpa izin Pelaku Pertunjukan, Produser Fonogram, atau Lembaga Penyiaran.

Sanksi Pelanggaran Pasal 113

1. Setiap Orang yang dengan tanpa hak melakukan pelanggaran hak ekonomi sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf i untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 1 (satu) tahun dan/atau pidana denda paling banyak Rp100.000.000 (seratus juta rupiah).
2. Setiap Orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf c, huruf d, huruf f, dan/atau huruf h untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 3 (tiga) tahun dan/atau pidana denda paling banyak Rp500.000.000,00 (lima ratus juta rupiah).

Arsitektur Komputer Fondasi Teknologi Digital

Rian Toni Rambe;

Deci Irmayani;

Volvo Sihombing;

PENERBIT YAYASAN MMI

Arsitektur Komputer Fondasi Teknologi Digital

Rian Toni Rambe; Deci Irmayani; Volvo Sihombing;

Desain Cover :
Musthafa Haris Munandar

Sumber :
https://munandar.yayasanmmi.com/arsitektur-komputer

Tata Letak :
Deci Irmayani

Proofreader :
Volvo Sihombing

Ukuran :
Jml hal judul: 1, Jml hal isi naskah: 66, Uk: 15x23 cm

ISBN :
978-634-04-1797-5

Cetakan Pertama :
Juli 2025

PENERBIT YAYASAN MUNANDAR MEMBANGUN INDONESIA
Jl. Pasar Banjar Dusun XVI Desa Simpang Empat, Asahan, Sumatera Utara 21271
Telp/Wa : 082363080181
https://munandar.yayasanmmi.com/
E-mail: mmipublisher@yayasanmmi.com

PRAKATA

Segala puji dan syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa atas rahmat dan karunia-Nya sehingga buku ini, yang berjudul "Arsitektur Komputer: Fondasi Teknologi Digital", dapat diselesaikan dengan baik. Buku ini hadir sebagai upaya untuk memberikan pemahaman mendalam dan terstruktur mengenai dasar-dasar arsitektur komputer, suatu bidang yang menjadi pilar utama dalam dunia teknologi informasi dan sistem digital modern.

Dalam era transformasi digital yang begitu cepat, pemahaman terhadap bagaimana komputer bekerja tidak lagi menjadi kebutuhan eksklusif para insinyur perangkat keras. Saat ini, siapa pun yang terlibat dalam pengembangan teknologi—baik itu di bidang pemrograman, keamanan siber, kecerdasan buatan, maupun sistem embedded—perlu memiliki fondasi yang kuat tentang arsitektur komputer. Melalui buku ini, penulis mencoba menyajikan konsep-konsep teknis secara sistematis, mulai dari representasi data dan struktur CPU, hingga memori, sistem bus, serta perangkat input/output yang menjadi penghubung utama antara manusia dan mesin.

Penulis menyadari bahwa penyusunan buku ini tentu belum sempurna. Oleh karena itu, kritik dan saran dari pembaca sangat diharapkan guna perbaikan pada edisi selanjutnya. Besar harapan penulis, buku ini dapat menjadi referensi yang bermanfaat dan menjadi jembatan bagi pembaca untuk memasuki dunia teknologi digital dengan pemahaman yang lebih matang dan menyeluruh.

Hormat Kami,

Penulis

DAFTAR ISI

PRAKATA **vi**

DAFTAR ISI **vii**

BAB I. PENGANTAR ARSITEKTUR KOMPUTER 1

1.1. Definisi arsitektur komputer 1

1.2. Sejarah perkembangan komputer dan arsitekturnya 2

1.3. Perbedaan arsitektur dan organisasi computer 4

1.4 Peran arsitektur dalam teknologi digital modern 5

BAB II. SISTEM BILANGAN DAN REPRESENTASI DATA 7

2.1. Sistem bilangan: biner, oktal, desimal, heksadesimal 7

2.2. Konversi antar sistem bilangan 8

2.3. Representasi data: bilangan bulat, pecahan, karakter (ASCII, Unicode) 9

2.4. Operasi aritmatika biner 10

BAB III. FONDASI PERANGKAT KERAS KOMPUTER 13

3.1. Struktur dasar komputer (CPU, Memori, I/O) 13

3.2. Komponen utama CPU (ALU, CU, register) 14

3.3. Jenis-jenis memori: RAM, ROM, cache, virtual memory 15

3.4. Sistem input/output dan periferal 17

BAB IV. UNIT PEMROSESAN PUSAT (CPU) 19

4.1. Arsitektur dasar CPU 19

4.2. Instruksi mesin dan siklus instruksi 20
4.3. Register dan peranannya 22
4.4. Pipeline dan eksekusi instruksi 23
BAB V. SET INSTRUKSI DAN BAHASA MESIN 25
5.1. Struktur dan format instruksi 25
5.2. Addressing mode (mode pengalamatan) 27
5.3. Kategori instruksi: aritmatika, logika, kontrol, I/O 29
5.4. Contoh arsitektur: RISC vs CISC 31
BAB VI. ORGANISASI MEMORI 34
6.1. Hirarki memori 34
6.2. Mekanisme cache dan manajemen memori 35
6.3. Paging dan segmentasi 37
6.4. Penyimpanan sekunder dan sistem file dasar 38
BAB VII. BUS DAN SISTEM KOMUNIKASI 41
7.1. Arsitektur bus 41
7.2. Jenis-jenis bus: data, alamat, control 43
7.3. Interkoneksi perangkat keras 44
7.4. DMA (Direct Memory Access) 46
BAB VIII. SISTEM MASUKAN DAN KELUARAN 48
8.1. Perangkat I/O dan teknik transfer data 48
8.2. Interrupt dan polling 50
8.3. Manajemen I/O dalam sistem computer 52
8.4. Teknologi I/O modern (USB, PCIe, SATA) 52
BIOGRAFI PENULIS 53

BAB I
PENGANTAR ARSITEKTUR KOMPUTER

1.1 Definisi arsitektur komputer

Arsitektur komputer adalah sebuah disiplin ilmu yang membahas struktur, perilaku, dan rancangan dari sistem komputer. Secara umum, istilah ini mengacu pada semua aspek yang terlihat oleh seorang pemrogram saat menggunakan komputer, seperti format instruksi, tipe data, mode pengalamatan, serta mekanisme input dan output. Lebih dalam lagi, arsitektur komputer merupakan gambaran konseptual dari bagaimana komputer beroperasi, termasuk bagaimana data disimpan, diproses, dan dipindahkan antar komponen internal.

Dalam dunia komputasi modern, pemahaman terhadap arsitektur komputer menjadi sangat penting karena komputer bukan lagi sekadar alat untuk mengetik atau menghitung, melainkan mesin universal yang menjadi dasar bagi hampir semua teknologi digital, mulai dari telepon pintar, kendaraan otomatis, hingga sistem cloud. Oleh karena itu, pengetahuan tentang bagaimana komputer bekerja di tingkat dasar memberi landasan kuat bagi siapa pun yang ingin membangun, mengoptimalkan, atau memahami sistem digital masa kini.

Arsitektur komputer biasanya terbagi dalam beberapa aspek utama, yaitu arsitektur instruksi (instruction set architecture), mikroarsitektur (microarchitecture), dan organisasi sistem. Instruction set architecture (ISA) menjelaskan perintah-perintah dasar yang bisa dimengerti oleh mesin. Mikroarsitektur berkaitan dengan bagaimana ISA tersebut diimplementasikan dalam perangkat keras. Sementara organisasi sistem membahas bagaimana komponen-komponen dalam komputer saling terhubung.

Sebagai contoh, dua buah prosesor bisa menggunakan ISA yang sama, seperti arsitektur x86, tetapi dirancang dengan mikroarsitektur yang berbeda. Satu mungkin fokus pada efisiensi energi untuk laptop, sedangkan yang lain dirancang untuk performa tinggi di server. Ini menjelaskan mengapa pemahaman arsitektur penting untuk memilih atau merancang sistem sesuai kebutuhan.

Lebih lanjut, arsitektur komputer menjadi titik temu antara perangkat keras dan perangkat lunak. Seorang programmer sistem, misalnya, perlu memahami struktur arsitektur komputer agar dapat menulis kode yang optimal. Begitu pula, seorang perancang perangkat keras harus merancang prosesor yang mampu menjalankan perangkat lunak dengan efisien dan akurat. Hubungan timbal balik inilah yang menjadikan arsitektur komputer sebagai fondasi dari semua bentuk teknologi digital modern.

1.2 Sejarah perkembangan komputer dan arsitekturnya

Sejarah perkembangan arsitektur komputer tidak bisa dilepaskan dari sejarah komputer itu sendiri. Komputer generasi pertama muncul pada pertengahan abad ke-20 dengan bentuk yang sangat besar dan keterbatasan fungsional. Salah satu tonggak penting dalam sejarah ini adalah pembuatan ENIAC (Electronic Numerical Integrator and Computer) pada tahun 1946 yang dianggap sebagai komputer elektronik pertama. ENIAC dirancang untuk kebutuhan militer dan mampu melakukan perhitungan numerik dalam skala besar, meskipun belum memiliki arsitektur yang terstandarisasi.

Kemudian, pada tahun 1945, John von Neumann memperkenalkan arsitektur yang kelak menjadi dasar dari sebagian besar komputer modern. Arsitektur Von Neumann memisahkan antara unit pemrosesan (CPU), memori utama, dan unit input/output, serta memperkenalkan konsep bahwa data dan

program disimpan dalam memori yang sama. Konsep ini sangat revolusioner pada masanya dan menjadi dasar bagi hampir semua sistem komputer saat ini.

Perjalanan evolusi arsitektur komputer kemudian berlanjut melalui berbagai generasi komputer. Generasi kedua menggunakan transistor sebagai pengganti tabung vakum, menjadikan komputer lebih kecil dan hemat daya. Generasi ketiga memperkenalkan sirkuit terpadu (IC), yang memungkinkan miniaturisasi lebih lanjut. Kemudian, generasi keempat membawa mikroprosesor – satu chip yang dapat menjalankan fungsi-fungsi CPU secara penuh.

Di era modern, arsitektur komputer berkembang lebih cepat dengan munculnya berbagai jenis arsitektur seperti RISC (Reduced Instruction Set Computing), CISC (Complex Instruction Set Computing), VLIW (Very Long Instruction Word), dan arsitektur paralel. Masing-masing arsitektur ini dirancang untuk tujuan tertentu, seperti efisiensi energi, kecepatan, atau kemampuan melakukan banyak tugas secara bersamaan.

Tidak hanya itu, munculnya komputasi awan (cloud computing), internet of things (IoT), dan kecerdasan buatan (AI) juga mendorong pengembangan arsitektur baru yang lebih khusus dan terdistribusi. Misalnya, arsitektur yang mendukung pemrosesan data di tepi jaringan (edge computing) kini banyak dikembangkan untuk mengurangi latensi dan meningkatkan efisiensi komunikasi data. Semua pencapaian ini menunjukkan bahwa arsitektur komputer tidak hanya berperan sebagai pondasi, tetapi juga sebagai penggerak utama kemajuan teknologi digital global.

1.3 Perbedaan arsitektur dan organisasi computer

Dalam dunia komputasi, dua istilah yang sering terdengar namun sering pula disalahartikan adalah arsitektur komputer dan organisasi komputer. Keduanya memang berkaitan erat, tetapi memiliki ruang lingkup yang berbeda. Memahami perbedaan antara keduanya penting untuk menempatkan aspek desain dan implementasi komputer secara tepat dalam konteks pengembangan teknologi digital.

Arsitektur komputer merujuk pada atribut-atribut sistem yang terlihat oleh seorang pemrogram. Ini mencakup hal-hal seperti jenis dan jumlah instruksi yang dapat digunakan oleh program (Instruction Set Architecture/ISA), format instruksi, mode pengalamatan, jenis data yang didukung, serta mekanisme input dan output. Dengan kata lain, arsitektur menggambarkan "apa" yang dilakukan sistem komputer. Fokusnya lebih kepada rancangan logis dari sistem, bukan bagaimana ia diimplementasikan secara fisik. Seorang pengembang perangkat lunak biasanya berinteraksi langsung dengan aspek ini ketika menulis program tingkat rendah atau sistem operasi.

Sementara itu, organisasi komputer berbicara tentang bagaimana komponen-komponen dalam sistem komputer diimplementasikan secara nyata. Ini mencakup bagaimana unit pemrosesan pusat (CPU), unit aritmatika dan logika (ALU), register, cache, serta sistem bus dirancang dan dihubungkan satu sama lain. Organisasi komputer berfokus pada "bagaimana" arsitektur itu dijalankan secara fisik oleh perangkat keras. Seorang insinyur perangkat keras atau perancang chip akan lebih akrab dengan konsep organisasi ini karena mereka harus memahami spesifikasi kelistrikan, kecepatan clock, bandwidth, dan ukuran cache.

Sebagai ilustrasi, dua komputer dapat memiliki arsitektur yang sama (misalnya keduanya mendukung instruksi x86), tetapi memiliki organisasi yang berbeda. Satu komputer bisa menggunakan cache L1 dan L2 yang besar serta pipeline yang

panjang untuk meningkatkan performa, sementara yang lain menggunakan desain minimalis untuk menghemat daya. Dalam hal ini, meskipun secara arsitektur sama, organisasi perangkat kerasnya berbeda.

Dalam praktiknya, pengembangan sistem komputer yang efisien membutuhkan pemahaman menyeluruh tentang keduanya. Arsitektur memberikan panduan tentang kemampuan logis dari sistem, sedangkan organisasi menentukan seberapa cepat, hemat daya, dan handalnya sistem tersebut dijalankan. Kolaborasi antara desain arsitektur dan organisasi yang baik menjadi kunci keberhasilan dalam membangun sistem digital modern yang kompetitif dan berkelanjutan.

1.1 Peran arsitektur dalam teknologi digital modern

Di era digital seperti sekarang, peran arsitektur komputer menjadi semakin penting dan strategis. Hampir semua perangkat digital yang kita gunakan sehari-hari – mulai dari smartphone, komputer pribadi, laptop, smart TV, hingga kendaraan pintar – semuanya didasarkan pada prinsip-prinsip arsitektur komputer. Tanpa desain arsitektur yang efisien, perangkat tersebut tidak akan mampu menjalankan fungsinya secara optimal, baik dari sisi kecepatan, konsumsi daya, maupun skalabilitas.

Arsitektur komputer bukan lagi hanya soal rancangan CPU dan memori, melainkan mencakup keseluruhan ekosistem komputasi, termasuk konektivitas jaringan, keamanan data, interoperabilitas antar perangkat, hingga efisiensi energi. Dalam sistem Internet of Things (IoT), misalnya, arsitektur harus dirancang agar setiap perangkat kecil dapat berkomunikasi secara nirkabel, menggunakan daya minimal, namun tetap dapat memproses data dalam waktu nyata. Desain seperti ini tidak mungkin dilakukan tanpa pemahaman mendalam tentang prinsip arsitektur komputer.

Lebih jauh lagi, dalam teknologi kecerdasan buatan (AI) dan pembelajaran mesin (machine learning), kebutuhan akan

pemrosesan data dalam jumlah besar secara cepat dan efisien telah mendorong lahirnya arsitektur baru seperti GPU (Graphics Processing Unit) dan TPU (Tensor Processing Unit). Perangkat keras ini dirancang secara khusus untuk mempercepat komputasi paralel dan pemrosesan matriks, yang menjadi inti dari model-model AI modern. Hal ini menandakan bahwa arsitektur komputer tidak hanya mendukung kebutuhan dasar komputasi, tetapi juga menjadi faktor kunci dalam inovasi teknologi canggih.

Dalam dunia komputasi awan (cloud computing), arsitektur komputer harus mendukung virtualisasi, manajemen sumber daya dinamis, dan akses skala besar dari berbagai lokasi secara simultan. Desain arsitektur sistem cloud menuntut kombinasi antara kecepatan, keandalan, keamanan, dan efisiensi biaya. Oleh karena itu, insinyur sistem harus mempertimbangkan setiap elemen arsitektural mulai dari prosesor, penyimpanan, jaringan, hingga lapisan aplikasi.

Akhirnya, arsitektur komputer juga memainkan peran penting dalam keberlanjutan teknologi. Dengan meningkatnya kesadaran terhadap efisiensi energi dan dampak lingkungan dari teknologi informasi, desain arsitektur yang hemat energi kini menjadi prioritas. Teknologi semacam ARM-based architecture, yang hemat daya namun cukup bertenaga, kini banyak digunakan dalam perangkat mobile dan bahkan server skala besar.

Singkatnya, arsitektur komputer adalah jantung dari revolusi teknologi digital saat ini. Tanpa fondasi arsitektural yang kuat dan adaptif, inovasi dalam teknologi informasi, komunikasi, dan otomatisasi tidak akan dapat berkembang secara optimal.

BAB II
SISTEM BILANGAN DAN REPRESENTASI DATA

2.1 Sistem bilangan: biner, oktal, desimal, heksadesimal

Sistem bilangan merupakan dasar dari semua operasi dalam sistem komputer. Komputer adalah mesin digital yang hanya mengenali dua keadaan, yaitu on dan off, yang secara logika direpresentasikan dengan angka 1 dan 0. Oleh karena itu, sistem bilangan biner (basis-2) menjadi sistem bilangan utama dalam dunia komputasi. Dalam sistem ini, hanya ada dua digit yaitu 0 dan 1, dan setiap digit disebut sebagai bit (binary digit). Kombinasi dari bit-bit ini dapat digunakan untuk merepresentasikan angka, karakter, instruksi, dan berbagai jenis data lainnya.

Meskipun komputer bekerja dengan bilangan biner, manusia lebih nyaman bekerja dengan sistem desimal (basis-10), yang menggunakan sepuluh digit dari 0 hingga 9. Untuk menjembatani kebutuhan manusia dan mesin, sistem bilangan lain seperti oktal (basis-8) dan heksadesimal (basis-16) juga digunakan dalam pemrograman dan representasi data. Sistem oktal terdiri dari digit 0 hingga 7, sementara sistem heksadesimal terdiri dari digit 0 hingga 9 dan huruf A hingga F, yang mewakili angka 10 hingga 15.

Salah satu alasan utama penggunaan sistem oktal dan heksadesimal adalah karena keduanya dapat dengan mudah dikonversi dari atau ke sistem biner. Misalnya, satu digit heksadesimal mewakili empat digit biner (4 bit), sedangkan satu digit oktal mewakili tiga digit biner (3 bit). Hal ini sangat membantu dalam membaca alamat memori, kode mesin, dan data dalam sistem komputer.

Dalam praktiknya, pemrogram sering menggunakan sistem heksadesimal untuk menyederhanakan tampilan data. Misalnya, alamat memori atau kode warna dalam pemrograman web sering ditulis dalam bentuk heksadesimal seperti 0xFFEE atau #FF0000. Demikian pula, sistem oktal sering digunakan dalam pengaturan hak akses file di sistem operasi Unix/Linux.

Pemahaman yang kuat tentang berbagai sistem bilangan ini sangat penting, terutama bagi mereka yang belajar arsitektur komputer, elektronika digital, atau pengembangan perangkat lunak tingkat rendah. Tanpa pemahaman dasar ini, akan sulit memahami bagaimana data direpresentasikan, diproses, dan disimpan di dalam komputer.

2.2 Konversi antar sistem bilangan

Konversi antar sistem bilangan merupakan keterampilan dasar namun penting dalam memahami cara kerja komputer. Karena sistem komputer menggunakan bilangan biner secara internal, sering kali kita perlu mengubah nilai dari satu sistem bilangan ke sistem bilangan lainnya agar lebih mudah dibaca atau diinterpretasikan. Empat sistem bilangan yang umum digunakan adalah desimal, biner, oktal, dan heksadesimal.

Konversi dari desimal ke biner dilakukan dengan membagi angka desimal dengan 2 secara berulang hingga hasilnya nol, lalu menuliskan sisa pembagian (bit) dari bawah ke atas. Misalnya, angka desimal 13 dapat dikonversi menjadi biner sebagai 1101. Proses sebaliknya, dari biner ke desimal, dilakukan dengan menjumlahkan setiap bit yang dikalikan dengan pangkat dua sesuai posisinya.

Untuk konversi dari desimal ke oktal atau heksadesimal, prinsipnya sama, hanya saja menggunakan pembagian dengan angka 8 atau 16. Sebaliknya, konversi dari oktal/heksadesimal ke desimal dilakukan dengan mengalikan setiap digit dengan

pangkat delapan atau enam belas. Misalnya, heksadesimal 1A sama dengan desimal 26 karena $1 \times 16 + 10 = 26$.

Konversi antar sistem bilangan non-desimal seperti biner ke heksadesimal atau ke oktal bisa dilakukan secara langsung dengan metode pengelompokan. Empat digit biner dikelompokkan menjadi satu digit heksadesimal, sementara tiga digit biner menjadi satu digit oktal. Contohnya, biner 11011101 dapat dikelompokkan sebagai 1101 1101, yang jika dikonversi menjadi heksadesimal menjadi DD. Pengelompokan ini membuat interpretasi data biner menjadi lebih ringkas dan mudah dibaca.

Kemampuan untuk melakukan konversi ini tidak hanya penting secara teoretis, tetapi juga sangat berguna dalam debugging, pemrograman sistem, analisis data mentah, dan memahami representasi memori. Banyak alat bantu konversi tersedia dalam kalkulator ilmiah, perangkat lunak, atau bahkan fitur bawaan di bahasa pemrograman, namun pemahaman konsep dasarnya tetap penting agar tidak bergantung sepenuhnya pada alat.

2.3 Representasi data: bilangan bulat, pecahan, karakter (ASCII, Unicode)

Dalam sistem komputer, semua jenis data yang kompleks—seperti gambar, suara, dokumen, dan bahkan video—akhirnya diwakili oleh bilangan biner. Untuk dapat menyimpan dan memproses informasi tersebut, komputer memerlukan sistem representasi data yang terstruktur. Salah satu bentuk representasi data yang paling dasar adalah bilangan bulat (integer). Bilangan bulat dapat direpresentasikan dalam dua bentuk: bilangan bertanda (signed) dan tak bertanda (unsigned). Untuk bilangan bertanda, komputer biasanya menggunakan representasi komplemen dua (two's complement) untuk menyimpan angka negatif.

Bilangan pecahan atau angka real direpresentasikan dengan pendekatan yang lebih kompleks, yaitu menggunakan standar floating-point, khususnya IEEE 754. Dalam representasi ini, bilangan dibagi menjadi tiga bagian utama: bit tanda, eksponen, dan mantisa (significand). Floating-point memungkinkan komputer menyimpan bilangan desimal yang sangat besar atau sangat kecil dengan presisi tinggi, meskipun tetap ada keterbatasan dalam akurasi.

Selain angka, komputer juga harus mampu merepresentasikan teks dan karakter. Untuk itu, digunakan sistem pengkodean seperti ASCII (American Standard Code for Information Interchange), yang menggunakan 7 bit untuk merepresentasikan karakter-karakter umum seperti huruf, angka, dan simbol. Namun ASCII memiliki keterbatasan karena hanya mencakup 128 karakter.

Untuk mengatasi keterbatasan ini, dikembangkanlah sistem pengkodean yang lebih luas yaitu Unicode, yang mampu merepresentasikan lebih dari 100.000 karakter dari berbagai bahasa dan simbol di seluruh dunia. Unicode menggunakan berbagai format penyimpanan seperti UTF-8, UTF-16, dan UTF-32. UTF-8 adalah format paling populer karena kompatibel dengan ASCII dan efisien dalam penggunaan ruang.

Pemahaman tentang representasi data sangat penting dalam pengembangan perangkat lunak, terutama ketika bekerja dengan data mentah, pengolahan file, komunikasi data, dan penyimpanan informasi di database. Kesalahan dalam pemrosesan tipe data dapat mengakibatkan kerusakan data, bug program, atau kegagalan komunikasi antar sistem.

2.1 Operasi aritmatika biner

Operasi aritmatika dalam komputer dilakukan dalam bentuk biner. Sama seperti operasi matematika pada sistem desimal, komputer juga melakukan penjumlahan, pengurangan,

perkalian, dan pembagian, tetapi menggunakan digit 0 dan 1. Operasi-operasi ini dilakukan oleh unit aritmatika dan logika (ALU) yang berada di dalam CPU. Untuk memahami cara kerja komputer, penting untuk memahami bagaimana operasi-operasi ini dilakukan dalam bentuk biner.

Penjumlahan biner sangat mirip dengan penjumlahan desimal, hanya saja basisnya adalah 2. Aturan dasar penjumlahan biner adalah:

- 0 + 0 = 0
- 0 + 1 = 1
- 1 + 0 = 1
- 1 + 1 = 0 (dengan simpanan *carry* 1)

Pengurangan biner dilakukan dengan aturan serupa, namun sering kali komputer menggunakan metode *komplemen dua* untuk menyederhanakan operasi pengurangan menjadi penjumlahan. Ini sangat efisien dan memudahkan desain sirkuit digital.

Perkalian biner menggunakan prinsip pengulangan penjumlahan, sama seperti perkalian dalam sistem desimal. Karena hanya terdiri dari angka 0 dan 1, proses perkalian biner menjadi lebih sederhana secara logika. Pembagian biner juga dilakukan melalui pendekatan yang menyerupai pembagian panjang dalam sistem desimal, namun lebih efisien untuk diimplementasikan dalam sirkuit digital.

Selain operasi aritmatika dasar, komputer juga melakukan operasi logika biner seperti AND, OR, XOR, dan NOT. Operasi ini penting dalam pemrograman tingkat rendah dan pengolahan bit.

Pemahaman operasi aritmatika biner sangat penting dalam arsitektur komputer karena menjadi dasar bagi semua pemrosesan data. Operasi-operasi ini memungkinkan komputer

menjalankan tugas-tugas mulai dari perhitungan matematis sederhana hingga pemrosesan data kompleks seperti enkripsi, pengolahan sinyal, dan pemodelan ilmiah.

Tabel 2.4 Bilangan Biner

Desimal	Biner	Oktal	Heksadesimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

BAB III

FONDASI PERANGKAT KERAS KOMPUTER

3.1 Struktur dasar komputer (CPU, Memori, I/O)

Struktur dasar komputer merupakan kerangka utama dari sistem komputer modern. Secara umum, komputer terdiri dari tiga komponen utama: Unit Pemrosesan Pusat (CPU), memori, dan unit input/output (I/O). Ketiga bagian ini saling terhubung dan bekerja sama untuk menjalankan program, memproses data, dan menghasilkan output yang diinginkan pengguna.

CPU, sering disebut sebagai otak komputer, bertanggung jawab atas eksekusi instruksi dan pengolahan data. Ia mengendalikan semua aktivitas komputer dan mengatur urutan pelaksanaan perintah yang diterima dari perangkat lunak. Komponen utama dalam CPU meliputi unit kontrol (Control Unit), unit aritmetika dan logika (ALU), dan register. Unit kontrol bertugas mengatur aliran data dan instruksi, sedangkan ALU menangani operasi matematis dan logika. Register berfungsi sebagai tempat penyimpanan data sementara yang digunakan selama pemrosesan.

Memori komputer menyimpan data dan instruksi yang diperlukan selama proses eksekusi. Memori utama atau RAM (Random Access Memory) digunakan untuk menyimpan data sementara yang bisa diakses cepat oleh CPU. Selain RAM, komputer juga memiliki ROM (Read Only Memory), yang menyimpan instruksi tetap seperti firmware. Memori berperan penting dalam menjaga kecepatan pemrosesan, karena data yang lebih dekat ke CPU dapat diakses lebih cepat dibandingkan data di penyimpanan sekunder.

Unit I/O memungkinkan komputer berkomunikasi dengan dunia luar. Perangkat input seperti keyboard, mouse, dan scanner

digunakan untuk memasukkan data ke dalam sistem, sedangkan perangkat output seperti monitor, printer, dan speaker menampilkan hasil pengolahan data. Komponen I/O juga mencakup antarmuka jaringan dan perangkat penyimpanan seperti hard drive dan SSD.

Ketiga komponen ini dihubungkan melalui sistem bus, yaitu jalur komunikasi yang memungkinkan pertukaran data antar bagian sistem. Dengan memahami struktur dasar komputer, kita dapat melihat bahwa komputer bekerja sebagai sistem terintegrasi yang kompleks, di mana efisiensi dan performa sangat bergantung pada sinergi antar CPU, memori, dan I/O. Struktur ini menjadi pondasi dalam mempelajari arsitektur komputer lebih lanjut, terutama dalam konteks desain dan optimisasi sistem digital.

3.2 Komponen utama CPU (ALU, CU, register)

Unit Pemrosesan Pusat (CPU) merupakan inti dari sebuah sistem komputer yang bertanggung jawab atas pelaksanaan instruksi program. Di dalam CPU, terdapat tiga komponen utama yang bekerja secara sinergis: ALU (Arithmetic Logic Unit), CU (Control Unit), dan register. Masing-masing memiliki peran penting dalam menjalankan operasi dasar komputer.

ALU (Arithmetic Logic Unit) adalah bagian dari CPU yang bertugas untuk melakukan operasi aritmatika (seperti penjumlahan, pengurangan, perkalian, dan pembagian) serta operasi logika (seperti AND, OR, XOR, NOT). ALU merupakan "mesin hitung" di dalam prosesor yang mengeksekusi perintah dasar dan menentukan hasil logika dari proses komputasi. Operasi ALU biasanya dilakukan berdasarkan sinyal kontrol dari unit pengendali (CU) dan menggunakan data yang disimpan di register.

CU (Control Unit) mengatur dan mengontrol semua aktivitas di dalam komputer. Fungsi utamanya adalah mengambil

instruksi dari memori, menginterpretasikannya, lalu mengarahkan bagian lain dari CPU dan perangkat keras terkait untuk melaksanakan instruksi tersebut. CU tidak melakukan pemrosesan data secara langsung, melainkan bertindak seperti pengatur lalu lintas yang memastikan semua komponen bekerja sesuai urutan yang benar dan sinkron.

Register adalah tempat penyimpanan kecil dan sangat cepat di dalam CPU. Register menyimpan data dan instruksi yang sedang diproses. Contoh register penting termasuk Program Counter (PC) yang menyimpan alamat instruksi berikutnya, Instruction Register (IR) yang menyimpan instruksi saat ini, dan Accumulator yang digunakan untuk menyimpan hasil sementara dari operasi ALU. Register memiliki akses tercepat dibandingkan memori lainnya, karena letaknya yang sangat dekat dengan ALU.

Kolaborasi antara ketiga komponen ini memungkinkan CPU untuk melakukan fetch-decode-execute cycle, yakni mengambil instruksi dari memori, menafsirkannya, dan menjalankannya. Setiap siklus ini terjadi dalam hitungan nanodetik dan dapat diulang jutaan kali per detik, tergantung pada kecepatan clock CPU.

Dengan memahami struktur internal CPU, kita dapat mengapresiasi betapa kompleks dan cepatnya proses komputasi modern. Inilah sebabnya desain mikroarsitektur CPU menjadi salah satu bidang paling menantang dalam teknik komputer, di mana efisiensi, kecepatan, dan ukuran menjadi faktor yang harus dioptimalkan secara bersamaan.

3.3 Jenis-jenis memori: RAM, ROM, cache, virtual memory

Sistem memori dalam komputer terdiri dari berbagai jenis yang masing-masing memiliki fungsi, kecepatan, dan kapasitas yang berbeda. Memori sangat penting dalam menunjang kinerja sistem karena menyediakan ruang bagi data dan instruksi yang

akan diproses. Semakin dekat memori ke CPU, semakin cepat aksesnya, meskipun biasanya dengan kapasitas yang lebih kecil. Beberapa jenis memori utama dalam sistem komputer meliputi RAM, ROM, cache, dan virtual memory.

RAM (Random Access Memory) adalah memori utama komputer yang digunakan untuk menyimpan data dan instruksi yang sedang digunakan saat ini. RAM bersifat volatil, artinya data akan hilang ketika daya dimatikan. RAM memiliki kecepatan akses yang tinggi dan sangat mempengaruhi kecepatan sistem secara keseluruhan. Komputer dengan kapasitas RAM yang besar dapat menangani lebih banyak aplikasi secara bersamaan tanpa melambat.

ROM (Read Only Memory) merupakan memori non-volatil yang menyimpan data permanen. Data pada ROM tidak dapat diubah atau hanya dapat diubah dengan proses khusus. ROM biasanya digunakan untuk menyimpan firmware, yaitu program dasar yang diperlukan untuk memulai dan mengatur sistem komputer sebelum sistem operasi dimuat.

Cache memory adalah memori kecil dan sangat cepat yang berada di dalam atau dekat CPU. Cache menyimpan data dan instruksi yang sering digunakan, sehingga CPU tidak perlu mengakses RAM terus-menerus. Cache dibagi menjadi beberapa level (L1, L2, dan L3) berdasarkan kecepatan dan kedekatannya dengan inti CPU. Penggunaan cache sangat meningkatkan kinerja sistem karena mengurangi waktu tunggu CPU.

Virtual memory adalah teknik manajemen memori di mana sebagian penyimpanan sekunder (seperti hard disk) digunakan sebagai perpanjangan dari RAM. Ketika RAM penuh, data yang tidak aktif sementara akan dipindahkan ke disk dalam bentuk file paging atau swap. Meskipun lebih lambat dibanding RAM, virtual memory memungkinkan komputer menjalankan program yang lebih besar dari kapasitas fisik RAM.

Pemilihan dan pengaturan memori yang tepat sangat penting dalam merancang sistem komputer yang efisien. Kombinasi memori cepat (seperti cache dan RAM) dengan memori kapasitas besar (seperti virtual memory) memungkinkan komputer menyeimbangkan antara kecepatan dan fleksibilitas. Pemahaman tentang berbagai jenis memori ini menjadi dasar dalam menganalisis performa sistem dan membuat keputusan desain arsitektur komputer yang efektif.

3.4 Sistem input/output dan periferal

Sistem input/output (I/O) merupakan komponen vital dalam arsitektur komputer yang memungkinkan interaksi antara komputer dengan pengguna dan dunia luar. Tanpa sistem I/O, komputer akan menjadi mesin yang tertutup dan tidak berguna. Perangkat input digunakan untuk memasukkan data dan perintah ke dalam sistem, sementara perangkat output digunakan untuk menampilkan hasil dari proses komputasi.

Perangkat input meliputi keyboard, mouse, scanner, mikrofon, kamera, dan perangkat sentuh. Data yang dikirim melalui perangkat ini akan diproses oleh CPU sesuai instruksi program. Di sisi lain, perangkat output seperti monitor, printer, speaker, dan proyektor bertugas menyajikan hasil pengolahan kepada pengguna.

Selain perangkat input dan output konvensional, sistem I/O juga mencakup perangkat penyimpanan eksternal seperti hard disk, SSD eksternal, flashdisk, serta antarmuka jaringan seperti LAN dan Wi-Fi yang memungkinkan komputer terhubung ke internet atau jaringan lokal. Perangkat-perangkat ini disebut sebagai periferal, dan sering kali terhubung melalui port seperti USB, HDMI, Thunderbolt, dan lainnya.

Sistem I/O dalam komputer modern menggunakan dua metode utama: interrupt dan polling. Dengan polling, CPU secara rutin memeriksa status perangkat I/O apakah siap berkomunikasi.

Metode ini sederhana, tetapi tidak efisien karena membuang waktu CPU. Sementara itu, dengan sistem interrupt, perangkat I/O akan "menginterupsi" CPU ketika membutuhkan perhatian. Pendekatan ini lebih efisien karena CPU bisa fokus pada tugas utama hingga diberi sinyal.

Untuk mempercepat transfer data, komputer juga menggunakan teknik DMA (Direct Memory Access). Dengan DMA, perangkat I/O dapat mentransfer data langsung ke atau dari memori tanpa campur tangan CPU secara terus-menerus. Ini sangat penting dalam transfer data berkecepatan tinggi seperti dalam sistem multimedia atau server.

Sistem I/O adalah jembatan utama antara pengguna dan perangkat lunak yang berjalan di dalam komputer. Desain dan pengelolaan yang efisien terhadap sistem I/O sangat berpengaruh terhadap kenyamanan penggunaan, responsivitas, dan kapabilitas sistem secara keseluruhan. Oleh karena itu, arsitektur komputer modern selalu menyertakan desain I/O yang fleksibel dan mampu mendukung berbagai jenis perangkat periferal.

BAB IV
UNIT PEMROSESAN PUSAT (CPU)

4.1 Arsitektur dasar CPU

Arsitektur dasar CPU menggambarkan bagaimana unit pemrosesan pusat dirancang untuk mengeksekusi instruksi secara efisien dalam sistem komputer. Arsitektur ini mencakup susunan komponen internal dan hubungan kerja antar bagian di dalam CPU, yang bersama-sama bertugas mengolah data, mengendalikan alur instruksi, serta berinteraksi dengan memori dan perangkat input/output. Meskipun teknologi CPU telah berkembang pesat, prinsip arsitektur dasarnya masih bertumpu pada tiga elemen utama: Unit Kontrol (Control Unit), Unit Aritmetika dan Logika (ALU), dan register.

Unit Kontrol (CU) bertanggung jawab atas pengambilan dan penafsiran instruksi dari memori. CU mengatur kapan dan bagaimana bagian-bagian CPU lainnya beroperasi. Misalnya, CU akan mengirimkan sinyal ke ALU untuk melakukan operasi tertentu, atau ke register untuk menyimpan hasil sementara. CU juga mengatur komunikasi CPU dengan komponen eksternal seperti RAM dan perangkat I/O melalui sistem bus.

ALU (Arithmetic Logic Unit) merupakan bagian CPU yang melaksanakan operasi aritmetika dasar (seperti penjumlahan dan pengurangan) serta operasi logika (seperti AND, OR, NOT). ALU menerima data dari register dan bekerja atas perintah CU. Hasil dari perhitungan akan disimpan kembali ke register atau dikirim ke memori.

Register adalah memori internal berukuran kecil namun sangat cepat yang digunakan untuk menyimpan data sementara selama instruksi diproses. Setiap CPU memiliki beberapa jenis register, seperti Program Counter (PC) yang menunjukkan alamat instruksi berikutnya, Instruction Register (IR) yang menyimpan

instruksi yang sedang dieksekusi, dan Accumulator yang menyimpan hasil dari operasi ALU. Karena kecepatannya yang tinggi, register memainkan peran penting dalam mempercepat proses eksekusi instruksi.

Semua komponen ini saling terhubung melalui sistem bus internal, yang terdiri dari bus data, bus alamat, dan bus kontrol. Bus-bus ini memungkinkan transfer informasi antar bagian CPU maupun antara CPU dan komponen luar.

Arsitektur dasar CPU juga menentukan bagaimana instruksi diproses—apakah menggunakan pendekatan von Neumann (instruksi dan data berbagi memori yang sama) atau Harvard (instruksi dan data memiliki jalur dan ruang memori terpisah). Pilihan arsitektur ini berdampak besar terhadap efisiensi, kecepatan, dan kemampuan paralelisme dalam eksekusi program.

4.2 Instruksi mesin dan siklus instruksi

Instruksi mesin merupakan perintah paling dasar yang dapat dimengerti dan dijalankan oleh Unit Pemrosesan Pusat (CPU). Setiap instruksi mesin terdiri dari kode biner tertentu yang mengarahkan CPU untuk melakukan tindakan spesifik, seperti menambahkan dua angka, memindahkan data dari satu lokasi ke lokasi lain, atau melompat ke bagian program lain. Instruksi-instruksi ini tersimpan dalam memori utama dan dieksekusi oleh CPU melalui suatu rangkaian proses yang disebut siklus instruksi (instruction cycle).

Siklus instruksi adalah prosedur sistematis yang dilakukan oleh CPU untuk mengeksekusi satu per satu instruksi dari program. Proses ini terdiri dari tiga tahap utama: fetch (mengambil instruksi), decode (menafsirkan instruksi), dan execute (menjalankan instruksi). Pada prosesor modern, siklus ini dapat berlangsung miliaran kali setiap detik.

Tahap pertama adalah fetch, di mana CPU mengambil instruksi dari alamat memori yang ditunjukkan oleh Program Counter (PC). Instruksi yang diambil akan disimpan di Instruction Register (IR). Setelah itu, nilai PC diperbarui untuk menunjuk ke alamat instruksi berikutnya. Langkah ini memastikan CPU dapat melanjutkan eksekusi program secara berurutan.

Tahap kedua adalah decode, yaitu proses penafsiran isi dari instruction register oleh Control Unit (CU). CU akan menganalisis bagian-bagian instruksi, seperti opcode (kode operasi) yang menentukan tindakan yang harus dilakukan, dan operand, yaitu data atau alamat memori yang terlibat dalam instruksi tersebut.

Tahap ketiga adalah execute, di mana instruksi yang telah ditafsirkan dilaksanakan. Jika instruksi adalah operasi aritmetika atau logika, maka ALU (Arithmetic Logic Unit) akan dipanggil untuk memprosesnya. Jika instruksi melibatkan pengambilan atau penyimpanan data ke memori, maka CPU akan melakukan akses memori sesuai perintah.

Siklus ini terus berulang selama komputer menyala dan program masih memiliki instruksi untuk dijalankan. CPU juga dapat menerima interupsi dari perangkat I/O atau sistem lainnya, sehingga perlu mengatur prioritas instruksi yang akan dieksekusi.

Instruksi mesin dan siklus instruksi merupakan inti dari seluruh proses komputasi. Tanpa pemahaman terhadap siklus ini, sulit memahami bagaimana sistem komputer melaksanakan perintah, mengelola data, dan merespons input dari pengguna atau sistem lain. Oleh karena itu, konsep ini menjadi pondasi dalam studi arsitektur komputer.

4.3 Register dan peranannya

Register adalah komponen internal CPU yang berfungsi sebagai tempat penyimpanan data sementara yang digunakan selama proses eksekusi instruksi. Dibandingkan dengan memori utama (RAM), register memiliki kapasitas yang jauh lebih kecil, tetapi kecepatannya sangat tinggi. Karena letaknya sangat dekat dengan ALU (Arithmetic Logic Unit), register memungkinkan CPU mengakses dan memproses data dengan latensi minimum. Tanpa register, CPU harus berulang kali mengakses RAM untuk setiap proses, yang akan menghambat kecepatan pemrosesan secara keseluruhan.

Register memainkan peranan penting dalam mendukung efisiensi siklus instruksi CPU, khususnya dalam tahap fetch, decode, dan execute. Selama eksekusi, hasil perhitungan dari ALU disimpan terlebih dahulu ke register sebelum ditulis kembali ke memori utama atau digunakan untuk operasi selanjutnya. Dengan kata lain, register bertindak sebagai "ruang kerja" internal bagi CPU dalam menangani data dan instruksi.

Dalam praktiknya, terdapat berbagai jenis register dengan fungsi spesifik. Beberapa register penting dalam arsitektur CPU antara lain:

1. Program Counter (PC): Menyimpan alamat memori dari instruksi berikutnya yang akan diambil oleh CPU. Setelah satu instruksi dijalankan, PC akan bertambah nilainya untuk menunjuk ke instruksi selanjutnya, kecuali terjadi interupsi atau instruksi lompat (jump).

2. Instruction Register (IR): Menyimpan instruksi yang sedang dieksekusi saat ini. Instruksi ini telah diambil dari memori dan siap ditafsirkan oleh Control Unit.

3. Accumulator (ACC): Digunakan untuk menyimpan hasil sementara dari operasi yang dilakukan oleh ALU. Dalam beberapa arsitektur, accumulator juga berfungsi sebagai salah satu operand.

4. General Purpose Register (GPR): Register serbaguna yang dapat digunakan untuk menyimpan data sementara, operand, atau hasil perhitungan. Contohnya dalam arsitektur x86 adalah register seperti AX, BX, CX, dan DX.

5. Status Register (atau Flag Register): Menyimpan informasi tentang status hasil operasi, seperti tanda overflow, carry, zero, atau sign. Register ini membantu CPU dalam mengambil keputusan, misalnya untuk melompat ke bagian program tertentu berdasarkan hasil logika.

Keberadaan dan desain register yang efisien sangat memengaruhi kecepatan dan kinerja CPU. Arsitektur komputer modern bahkan menyertakan ratusan register dalam register file untuk mendukung operasi paralel dan pipelining. Selain itu, register juga menjadi bagian penting dalam pengembangan bahasa assembly dan pengoptimalan kompilasi perangkat lunak tingkat rendah.

4.1 Pipeline dan eksekusi instruksi

Pipeline merupakan teknik dalam arsitektur komputer yang digunakan untuk meningkatkan efisiensi dan kecepatan eksekusi instruksi oleh CPU. Konsep pipelining berasal dari prinsip dasar pada jalur produksi di industri manufaktur: membagi proses kompleks menjadi beberapa tahap, dan setiap tahap dikerjakan secara paralel oleh unit yang berbeda. Dalam konteks CPU, pipelining memungkinkan beberapa instruksi berada dalam berbagai tahap siklus instruksi secara bersamaan.

Secara tradisional, siklus instruksi terdiri dari tiga tahap utama: **fetch**, **decode**, dan **execute**. Tanpa pipelining, CPU harus menyelesaikan satu instruksi sepenuhnya sebelum memulai instruksi berikutnya. Akibatnya, sebagian besar waktu CPU terbuang untuk menunggu. Dengan pipelining, tahap-tahap ini

bisa dilakukan secara paralel untuk instruksi yang berbeda. Saat satu instruksi sedang dieksekusi, instruksi berikutnya sudah mulai di-decode, dan instruksi ketiga sudah mulai di-fetch.

Sebagai contoh, bayangkan sebuah CPU dengan tiga tahap pipeline. Pada siklus pertama, CPU men-*fetch* instruksi pertama. Pada siklus kedua, instruksi pertama masuk ke tahap *decode*, dan CPU men-*fetch* instruksi kedua. Pada siklus ketiga, instruksi pertama masuk ke tahap *execute*, instruksi kedua ke *decode*, dan instruksi ketiga di-*fetch*. Dengan demikian, setiap siklus clock digunakan secara maksimal untuk memproses bagian berbeda dari instruksi yang berbeda.

Namun, pipelining juga memiliki tantangan yang dikenal sebagai **hazard**, yaitu kondisi yang mengganggu kelancaran aliran instruksi dalam pipeline. Terdapat tiga jenis hazard utama:

- **Data Hazard**: Terjadi ketika satu instruksi bergantung pada hasil dari instruksi sebelumnya yang belum selesai.
- **Control Hazard**: Terjadi pada instruksi cabang (branch) yang memengaruhi jalur eksekusi berikutnya.
- **Structural Hazard**: Terjadi jika dua atau lebih instruksi membutuhkan perangkat keras yang sama pada waktu bersamaan.

Untuk mengatasi hazard, digunakan teknik seperti **forwarding**, **branch prediction**, **pipeline stalling**, dan **out-of-order execution**.

Pipelining telah menjadi komponen penting dalam CPU modern karena dapat meningkatkan throughput secara signifikan tanpa meningkatkan frekuensi clock. Ini memungkinkan CPU menyelesaikan lebih banyak instruksi dalam waktu yang sama, menghasilkan peningkatan performa secara keseluruhan. Tanpa pipelining, prosesor tidak akan mampu memenuhi tuntutan aplikasi komputasi modern yang membutuhkan kecepatan dan efisiensi tinggi.

BAB V
SET INSTRUKSI DAN BAHASA MESIN

5.1 Struktur dan format instruksi

Dalam arsitektur komputer, **instruksi** adalah perintah dasar yang dijalankan oleh CPU untuk melakukan suatu tugas tertentu, seperti memindahkan data, melakukan operasi aritmetika, atau mengendalikan alur program. Agar dapat diproses oleh CPU, setiap instruksi harus memiliki **struktur dan format** yang sesuai dengan arsitektur set instruksi yang digunakan. Struktur ini menentukan bagaimana instruksi dipecah menjadi bagian-bagian biner yang dapat diinterpretasikan dan dieksekusi oleh unit kontrol prosesor.

Secara umum, sebuah instruksi terdiri dari dua bagian utama, yaitu **Opcode (Operation Code)** dan **Operand**. **Opcode** menunjukkan jenis operasi yang harus dilakukan (misalnya tambah, pindah data, bandingkan, atau lompat), sedangkan **Operand** berisi informasi tambahan seperti data literal, alamat memori, atau referensi ke register.

Struktur instruksi biasanya ditentukan oleh panjang bit yang digunakan oleh arsitektur prosesor. Misalnya, pada arsitektur 32-bit, sebuah instruksi bisa saja terdiri dari 6 bit untuk opcode dan sisanya untuk operand atau alamat. Format umum instruksi mencakup:

- **Opcode**: bagian ini menentukan tindakan apa yang akan dilakukan.
- **Source Operand(s)**: data yang akan diproses atau lokasi sumber data.
- **Destination Operand**: lokasi penyimpanan hasil operasi.

- **Addressing Mode**: menunjukkan bagaimana operand diakses (langsung, tak langsung, melalui register, dll.).

Terdapat beberapa **format instruksi** yang umum digunakan, yaitu:

1. **Format 0-operand (zero-address instruction)** – biasanya digunakan pada arsitektur stack, di mana semua operasi dilakukan langsung di atas tumpukan (stack).
2. **Format 1-operand** – satu operand eksplisit, operand lainnya diasumsikan dari register tetap (biasanya accumulator).
3. **Format 2-operand** – dua operand eksplisit; hasil biasanya menggantikan salah satu operand.
4. **Format 3-operand** – dua operand sumber dan satu operand tujuan, yang umum pada arsitektur RISC seperti MIPS.

Pemilihan format instruksi sangat berpengaruh terhadap efisiensi ruang (bit yang dibutuhkan), kecepatan decoding oleh CPU, serta fleksibilitas pemrograman. Misalnya, arsitektur **RISC (Reduced Instruction Set Computer)** cenderung menggunakan format instruksi tetap dengan panjang yang sama agar decoding cepat dan prediktif, sedangkan **CISC (Complex Instruction Set Computer)** memiliki format variabel yang lebih fleksibel tapi lebih kompleks untuk diproses.

Dengan memahami struktur dan format instruksi, kita dapat membaca atau menulis kode mesin secara langsung, menganalisis performa program, serta merancang perangkat keras dan perangkat lunak yang efisien dan kompatibel. Ini menjadi dasar penting dalam pengembangan sistem operasi, kompiler, dan emulator arsitektur komputer.

5.2 Addressing mode (mode pengalamatan)

Addressing mode atau **mode pengalamatan** adalah cara yang digunakan oleh instruksi dalam CPU untuk mengakses operand, yaitu data yang akan diolah selama eksekusi instruksi. Operand tersebut bisa berupa data langsung, nilai yang tersimpan di memori, atau isi dari suatu register. Pemilihan addressing mode sangat menentukan fleksibilitas, efisiensi, dan kompleksitas instruksi dalam arsitektur komputer. Dengan memahami addressing mode, seorang programmer dapat mengoptimalkan pemrosesan data dan memanfaatkan sumber daya prosesor dengan lebih efisien.

Dalam proses eksekusi, CPU harus mengetahui dari mana data akan diambil atau ke mana hasil akan disimpan. Addressing mode menjelaskan bagaimana instruksi menginterpretasikan bagian operand dari sebuah instruksi untuk menentukan lokasi operand tersebut. Setiap arsitektur komputer memiliki seperangkat addressing mode yang mendukung strategi pemrograman dan desain sistem yang berbeda.

Berikut adalah beberapa jenis addressing mode yang umum digunakan:

1. **Immediate Addressing**

   Operand langsung diberikan dalam instruksi itu sendiri. Misalnya: MOV R1, #5 berarti nilai 5 langsung dimasukkan ke register R1. Mode ini cepat karena tidak memerlukan akses memori tambahan.

2. **Register Addressing**

   Operand berada di dalam register. Contoh: ADD R1, R2 berarti tambahkan isi register R2 ke R1. Akses register lebih cepat dibanding akses memori.

3. **Direct Addressing (Absolute Addressing)**

   Alamat operand secara eksplisit disebutkan dalam instruksi. Misalnya: MOV R1, [2000] berarti ambil data dari alamat memori 2000 dan simpan ke R1.

4. **Indirect Addressing**

   Operand adalah alamat yang tersimpan dalam register atau memori. Contoh: MOV R1, [R2] artinya ambil data dari alamat yang ditunjuk oleh R2.

5. **Indexed Addressing**

   Alamat efektif diperoleh dari penjumlahan antara nilai dasar (base) dengan indeks (offset). Digunakan dalam pengolahan array atau struktur data yang berulang.

6. **Relative Addressing**

   Digunakan untuk instruksi percabangan, di mana alamat tujuan dihitung berdasarkan offset relatif terhadap program counter saat ini.

7. **Base-Register Addressing**

   Alamat dihitung dengan menambahkan nilai offset ke isi register dasar. Berguna dalam pengolahan struktur data kompleks.

Setiap mode pengalamatan memiliki keunggulan dan kelemahan tergantung pada konteksnya. Mode-mode ini memungkinkan penghematan ruang instruksi, mempercepat eksekusi, dan mendukung struktur data serta kontrol alur program yang lebih kompleks.

Dalam perancangan CPU dan sistem instruksi, addressing mode memainkan peran penting untuk menyeimbangkan antara fleksibilitas pemrograman dan efisiensi perangkat keras. Oleh karena itu, pemahaman yang baik mengenai addressing mode

adalah kunci untuk menulis kode mesin atau assembly yang optimal dan efektif.

5.3 Kategori instruksi: aritmatika, logika, kontrol, I/O

Set instruksi dalam arsitektur komputer diklasifikasikan ke dalam beberapa **kategori** berdasarkan jenis operasi yang dilakukan. Klasifikasi ini bertujuan untuk memudahkan pemahaman struktur sistem komputer, menyusun program dalam bahasa mesin, dan mengoptimalkan kinerja perangkat keras dalam eksekusi tugas tertentu. Secara umum, kategori instruksi terbagi menjadi empat kelompok utama: **instruksi aritmatika**, **instruksi logika**, **instruksi kontrol**, dan **instruksi input/output (I/O)**.

1. Instruksi Aritmatika

Instruksi aritmatika digunakan untuk melakukan operasi perhitungan matematis dasar seperti penjumlahan, pengurangan, perkalian, dan pembagian. Operasi ini biasanya dilakukan oleh ALU (Arithmetic Logic Unit). Contoh instruksi aritmatika meliputi:

- ADD R1, R2, R3 → Menambahkan isi R2 dan R3, simpan hasilnya di R1.
- SUB R4, R4, R5 → Mengurangkan isi R5 dari R4.
- MUL R1, R2 → Mengalikan isi R1 dan R2.

Instruksi ini sangat penting dalam pengolahan data numerik, seperti perhitungan statistik, algoritma kriptografi, hingga simulasi ilmiah.

2. Instruksi Logika

Instruksi logika digunakan untuk memanipulasi bit secara logis. Operasi ini mencakup AND, OR, NOT, XOR, serta pergeseran bit (shift left/right). Instruksi logika berperan penting

dalam pengolahan data biner, validasi, dan manipulasi kondisi tertentu. Contoh:

- AND R1, R2, R3 → Melakukan operasi logika AND pada R2 dan R3, hasil ke R1.
- OR R4, R4, R5 → OR bit demi bit antara R4 dan R5.
- NOT R1 → Membalik semua bit di R1.

Instruksi ini sering digunakan dalam sistem operasi, kontrol perangkat keras, dan algoritma bitwise.

3. Instruksi Kontrol

Instruksi kontrol bertugas mengatur alur eksekusi program. Instruksi ini mencakup percabangan (branching), lompat (jump), dan pemanggilan prosedur (call/return). Tujuannya adalah memungkinkan program membuat keputusan atau mengulangi bagian tertentu. Contoh:

- JMP 1000 → Melompat ke alamat memori 1000.
- BEQ R1, R2, label → Lompat ke label jika R1 = R2.
- CALL Subroutine → Memanggil subprogram.

Instruksi kontrol memungkinkan struktur logika seperti if-else dan loop direalisasikan di tingkat mesin.

4. Instruksi Input/Output (I/O)

Instruksi I/O digunakan untuk berkomunikasi dengan perangkat luar seperti keyboard, mouse, printer, atau hard disk. Instruksi ini memungkinkan CPU membaca data dari perangkat input dan menulis ke perangkat output. Contoh umum:

- IN R1, PORTA → Membaca data dari PORTA ke R1.
- OUT PORTB, R2 → Menulis isi R2 ke PORTB.

Karena perangkat I/O seringkali bekerja lebih lambat dari CPU, instruksi I/O juga berkaitan erat dengan manajemen interupsi dan buffer.

Dengan memahami klasifikasi instruksi ini, kita dapat melihat bahwa setiap program komputer pada dasarnya adalah rangkaian dari operasi-operasi dasar ini. Desain set instruksi yang baik akan mencerminkan keseimbangan antara kompleksitas perangkat keras dan kemudahan pemrograman, menjadikan sistem lebih efisien dan fleksibel dalam menangani berbagai jenis aplikasi.

5.4 Contoh arsitektur: RISC vs CISC

Dalam dunia arsitektur komputer, terdapat dua pendekatan utama dalam merancang set instruksi dan struktur internal CPU, yaitu **RISC (Reduced Instruction Set Computer)** dan **CISC (Complex Instruction Set Computer)**. Kedua arsitektur ini memiliki filosofi desain yang berbeda dan mencerminkan dua strategi untuk meningkatkan efisiensi dan kinerja prosesor. Pemahaman terhadap perbedaan antara RISC dan CISC sangat penting karena banyak prosesor modern—baik dalam perangkat desktop, mobile, maupun embedded—dibangun berdasarkan salah satu dari dua pendekatan ini.

1. Arsitektur RISC (Reduced Instruction Set Computer)

RISC dirancang dengan prinsip menyederhanakan set instruksi sehingga setiap instruksi dapat dieksekusi dalam satu siklus mesin. Karakteristik utama arsitektur RISC antara lain:

- Menggunakan jumlah instruksi yang sedikit dan sederhana.
- Instruksi memiliki format tetap dan panjang yang seragam.

- Mengandalkan register-register internal dalam jumlah besar untuk meminimalkan akses memori.
- Eksekusi instruksi lebih cepat dan efisien.
- Mudah di-*pipeline*, sehingga cocok untuk pemrosesan paralel.

Contoh arsitektur RISC meliputi **MIPS**, **SPARC**, **ARM**, dan **RISC-V**. Arsitektur ARM, misalnya, sangat dominan di perangkat mobile karena hemat daya dan efisien. Kompiler memiliki peran besar dalam menghasilkan urutan instruksi optimal karena CPU tidak menyediakan instruksi kompleks.

2. Arsitektur CISC (Complex Instruction Set Computer)

Sebaliknya, CISC menggunakan banyak instruksi dengan tingkat kompleksitas yang lebih tinggi. Karakteristik utama CISC:

- Set instruksinya luas dan kompleks, bisa mencakup ratusan instruksi.
- Satu instruksi dapat melakukan beberapa operasi sekaligus (misalnya, mengambil data dari memori, memprosesnya, dan menyimpannya kembali).
- Format instruksi variatif (panjang dan strukturnya tidak seragam).
- Jumlah register lebih sedikit dibanding RISC.
- Pendekatan ini bertujuan mengurangi jumlah instruksi dalam program.

Contoh paling terkenal dari arsitektur CISC adalah **Intel x86** dan **AMD64**, yang mendominasi pasar PC dan server. Meskipun kompleks, prosesor CISC modern telah dioptimalkan secara internal untuk men-decode instruksi kompleks menjadi mikro-operasi RISC agar dapat dieksekusi secara efisien.

Perbandingan dan Perkembangan

Perkembangan teknologi membuat batas antara RISC dan CISC semakin kabur. Banyak prosesor CISC modern mengadopsi teknik internal RISC (misalnya, pipelining, register renaming), sementara prosesor RISC menambahkan instruksi kompleks untuk efisiensi tertentu. Pilihan antara RISC dan CISC kini lebih bergantung pada konteks penggunaan: efisiensi energi dan ukuran (mobile) cenderung memilih RISC, sedangkan performa tinggi dan kompatibilitas (PC/server) lebih memilih CISC.

Dengan memahami kedua arsitektur ini, kita dapat menilai kelebihan dan kekurangan masing-masing pendekatan serta bagaimana mereka diimplementasikan dalam prosesor modern yang kita gunakan sehari-hari.

BAB VI
ORGANISASI MEMORI

6.1 Hirarki memori

Dalam arsitektur komputer modern, **hirarki memori** merupakan konsep fundamental yang menggambarkan susunan sistem memori berdasarkan kecepatan akses, kapasitas, dan biaya per bit. Tujuan utama dari hirarki memori adalah untuk menciptakan sistem penyimpanan yang seimbang antara kinerja dan efisiensi biaya, dengan memanfaatkan berbagai jenis memori yang memiliki karakteristik berbeda. Karena teknologi penyimpanan cepat seperti register dan cache sangat mahal dan terbatas kapasitasnya, sementara memori besar seperti hard disk atau penyimpanan awan jauh lebih lambat, maka dibutuhkan struktur hierarkis yang mengatur bagaimana data disimpan dan diakses.

Hirarki memori secara umum terdiri dari beberapa tingkatan, dimulai dari yang paling cepat namun paling kecil dan mahal di puncak, hingga yang paling lambat namun paling besar dan murah di dasar. Tingkatan dalam hirarki memori tersebut meliputi:

1. **Register**
   Merupakan bagian dari CPU itu sendiri. Memori ini sangat cepat tetapi sangat terbatas jumlahnya. Register digunakan untuk menyimpan data sementara selama eksekusi instruksi.

2. **Cache Memory (L1, L2, L3)**

   Cache adalah memori berkecepatan tinggi yang berfungsi sebagai penyangga antara CPU dan RAM. Cache L1 adalah yang tercepat dan paling kecil, berada sangat dekat dengan inti CPU. L2 dan L3 lebih besar namun sedikit

lebih lambat. Cache menyimpan data yang sering diakses agar CPU tidak perlu mengambilnya langsung dari RAM.

3. **Main Memory (RAM)**

   RAM adalah memori utama komputer yang digunakan untuk menyimpan data dan instruksi selama program dijalankan. Meskipun lebih lambat dari cache, RAM memiliki kapasitas lebih besar dan bersifat volatile, artinya data akan hilang saat daya dimatikan.

4. **Secondary Storage**

   Termasuk hard disk drive (HDD), solid-state drive (SSD), dan media penyimpanan lainnya. Kecepatan aksesnya jauh lebih lambat dibanding RAM, tetapi kapasitasnya sangat besar dan bersifat non-volatile.

5. **Tertiary Storage dan Cloud Storage**

   Digunakan untuk backup atau arsip data jangka panjang. Contohnya tape drive atau layanan penyimpanan daring. Akses data sangat lambat tetapi biaya per bit sangat murah.

Prinsip kerja dari hirarki memori adalah memanfaatkan **locality of reference**—yaitu kemungkinan besar data yang baru diakses akan diakses kembali dalam waktu dekat (temporal locality) atau data yang berdekatan dalam alamat memori juga akan digunakan (spatial locality). Dengan menempatkan data yang paling sering digunakan pada memori paling cepat, sistem dapat mencapai efisiensi performa yang tinggi tanpa harus menggunakan memori cepat dalam kapasitas besar.

6.2 Mekanisme cache dan manajemen memori

Dalam sistem komputer modern, kecepatan akses antara CPU dan memori utama (RAM) sering kali tidak seimbang. CPU

mampu memproses data jauh lebih cepat daripada kecepatan RAM dalam menyediakannya. Untuk mengatasi kesenjangan ini, digunakanlah **memori cache**, sebuah jenis memori berkecepatan tinggi yang bertindak sebagai perantara antara CPU dan RAM. **Mekanisme cache** bekerja dengan menyimpan data atau instruksi yang sering digunakan (disebut *temporal locality*) atau data yang letaknya berdekatan (*spatial locality*) agar CPU dapat mengaksesnya dengan cepat tanpa harus menunggu dari RAM.

Cache biasanya terbagi menjadi beberapa tingkat: **L1 (Level 1)**, **L2**, dan **L3**. Cache L1 berada paling dekat dengan inti CPU dan memiliki kecepatan paling tinggi namun kapasitas terkecil. L2 dan L3 memiliki kapasitas lebih besar, tapi kecepatan aksesnya sedikit lebih rendah. Ketika CPU membutuhkan data, ia pertama-tama akan mencari di L1, kemudian L2, lalu L3, dan jika tidak ditemukan (disebut *cache miss*), baru diambil dari RAM.

Untuk mengatur data dalam cache, digunakan beberapa **strategi manajemen**, seperti:

- **Mapping Techniques**: menentukan bagaimana data dari RAM dipetakan ke dalam cache. Contohnya *direct-mapped cache*, *associative cache*, dan *set-associative cache*.
- **Replacement Policy**: jika cache sudah penuh, kebijakan ini menentukan data mana yang harus diganti. Contoh: *Least Recently Used (LRU)* atau *First In First Out (FIFO)*.
- **Write Policy**: menentukan bagaimana perubahan data di cache ditulis ke memori utama, seperti *write-through* (langsung ditulis ke RAM) dan *write-back* (baru ditulis saat data diganti).

Sementara itu, **manajemen memori** adalah sistem yang mengatur bagaimana ruang memori dialokasikan dan digunakan

oleh proses-proses dalam komputer. Sistem operasi memiliki peran penting di sini, termasuk dalam:

- **Alokasi Memori Dinamis** (malloc/free),
- **Segmentasi dan Paging**, untuk mengatur pemetaan memori secara efisien,
- serta **virtual memory**, yang memungkinkan program seolah-olah memiliki ruang memori besar meski fisik RAM terbatas, dengan menggunakan ruang di penyimpanan sekunder (seperti HDD/SSD).

Kombinasi mekanisme cache dan manajemen memori memungkinkan komputer bekerja secara efisien dengan kecepatan tinggi meskipun menghadapi keterbatasan memori fisik. Tanpa sistem ini, performa CPU akan sangat terhambat akibat latensi tinggi saat mengambil data dari memori utama atau penyimpanan sekunder.

6.3 Paging dan segmentasi

Dalam pengelolaan memori modern, sistem operasi dan perangkat keras bekerja sama untuk menyediakan mekanisme yang efisien dalam mengalokasikan dan melindungi ruang memori. Dua teknik utama yang digunakan dalam **manajemen memori** adalah **paging** dan **segmentasi**. Keduanya bertujuan untuk mengatur pemetaan antara **alamat virtual** yang digunakan oleh program dan **alamat fisik** yang tersedia di RAM. Dengan teknik ini, komputer dapat menjalankan banyak program secara bersamaan, mencegah konflik memori antarproses, dan memaksimalkan pemanfaatan ruang memori fisik.

Paging

Paging adalah teknik manajemen memori di mana ruang alamat virtual dan fisik dibagi menjadi blok-blok berukuran tetap. Blok pada alamat virtual disebut **page**, sementara blok pada

memori fisik disebut **frame**. Ketika sebuah program dijalankan, hanya sebagian kecil dari keseluruhan halaman (page) yang perlu dimuat ke RAM, dan sisanya dapat disimpan di penyimpanan sekunder.

Saat program mengakses alamat virtual tertentu, sistem operasi dan unit manajemen memori (MMU – Memory Management Unit) akan menerjemahkannya ke alamat fisik melalui **page table**. Jika page yang diminta tidak ada di memori (terjadi **page fault**), maka sistem akan memuatnya dari disk ke RAM.

Paging memiliki keunggulan karena:

- Menghilangkan fragmentasi eksternal.
- Memungkinkan penggunaan **virtual memory**, sehingga program bisa lebih besar dari RAM yang tersedia.
- Menyediakan isolasi antarproses yang lebih aman.

Segmentasi

Berbeda dengan paging, **segmentasi** membagi memori menjadi bagian-bagian logis yang disebut **segmen**, seperti segmen kode, segmen data, dan segmen stack. Setiap segmen memiliki panjang yang bervariasi tergantung kebutuhan program. Segmentasi lebih sesuai dengan struktur logika program dan mendukung perlindungan memori yang lebih fleksibel.

Setiap alamat virtual dalam segmentasi terdiri dari dua bagian: **nomor segmen** dan **offset**. MMU akan menggunakan **segment table** untuk memetakan segmen ke lokasi memori fisik. Segmentasi memudahkan pembagian tugas, pengamanan akses memori, dan pengelompokan data dalam struktur modular.

6.4 Penyimpanan sekunder dan sistem file dasar

Dalam sistem komputer, **penyimpanan sekunder (secondary storage)** merupakan komponen penting yang

berfungsi untuk menyimpan data dan program secara permanen. Berbeda dengan memori utama (RAM) yang bersifat volatil—data akan hilang saat komputer dimatikan—penyimpanan sekunder bersifat non-volatil dan dirancang untuk menyimpan data dalam jangka panjang. Beberapa contoh media penyimpanan sekunder adalah **hard disk drive (HDD)**, **solid-state drive (SSD)**, **optical disk (CD/DVD)**, dan **flash drive**.

Penyimpanan sekunder berperan sebagai tempat utama untuk menyimpan sistem operasi, aplikasi, serta file pengguna. Karena kapasitasnya jauh lebih besar dan harganya lebih murah per gigabyte dibanding memori utama, penyimpanan sekunder menjadi solusi ideal untuk kebutuhan data dalam jumlah besar. Namun, kelemahan utama dari media ini adalah kecepatan akses yang lebih lambat dibandingkan RAM dan cache, sehingga digunakan terutama untuk menyimpan data yang tidak sedang diproses secara aktif.

Agar komputer dapat mengelola dan mengakses data dalam penyimpanan sekunder secara terstruktur, diperlukan sebuah sistem yang disebut **sistem file (file system)**. Sistem file bertanggung jawab atas organisasi data, penamaan file, pengelompokan dalam folder atau direktori, serta pengaturan hak akses dan metadata file (seperti waktu pembuatan dan ukuran file).

Beberapa sistem file yang umum digunakan meliputi:

- **FAT32 (File Allocation Table)**: sistem file sederhana dan kompatibel dengan berbagai perangkat, tetapi memiliki keterbatasan seperti ukuran maksimal file 4GB.
- **NTFS (New Technology File System)**: sistem file standar pada Windows modern yang mendukung enkripsi, kompresi, dan kontrol akses.

- **ext4 (Fourth Extended Filesystem)**: digunakan oleh banyak sistem Linux, mendukung volume besar, journaling, dan efisiensi tinggi.
- **exFAT**: dirancang untuk flash storage seperti USB dan SD card, dengan kompatibilitas luas dan dukungan untuk file besar.

Selain pengorganisasian, sistem file juga berperan dalam mengelola ruang penyimpanan fisik pada media, termasuk proses **alokasi blok**, **fragmentasi**, dan **defragmentasi**. Sistem file juga memungkinkan **mounting** dan **partitioning**, yakni membagi satu disk fisik menjadi beberapa bagian logis untuk keperluan berbeda.

BAB VII

BUS DAN SISTEM KOMUNIKASI

7.1 Arsitektur bus

Dalam sistem komputer, **bus** adalah jalur komunikasi yang memungkinkan pertukaran data antara berbagai komponen internal, seperti CPU, memori, perangkat input/output, dan penyimpanan sekunder. Bus bertindak sebagai "jalan raya" digital yang menghubungkan berbagai bagian sistem komputer agar dapat bekerja secara terpadu. Konsep **arsitektur bus** mencakup struktur, fungsi, dan mekanisme pengendalian dari jalur-jalur ini dalam mengoordinasikan komunikasi data.

Secara umum, bus terdiri dari tiga bagian utama, yaitu:

1. **Data Bus**

   Digunakan untuk mentransfer data aktual antar komponen. Lebar data bus (misalnya 8, 16, 32, atau 64 bit) menentukan jumlah data yang dapat dikirim dalam satu siklus. Semakin lebar data bus, semakin cepat sistem dapat mentransfer data.

2. **Address Bus**

   Menyampaikan alamat lokasi memori atau perangkat I/O yang menjadi tujuan atau sumber data. Address bus bersifat unidirectional (satu arah), dari CPU ke komponen lain. Lebar address bus menentukan kapasitas maksimum alamat memori yang dapat diakses. Misalnya, address bus 32-bit dapat menjangkau hingga 4 GB memori.

3. **Control Bus**

   Membawa sinyal kendali dan sinyal status, seperti *read*, *write*, *interrupt*, dan *clock*. Control bus mengatur bagaimana dan kapan data dikirim melalui data bus, serta siapa yang memiliki kendali pada suatu waktu.

Arsitektur bus dapat diklasifikasikan menjadi dua kategori utama:

- **Bus Tunggal (Single Bus Architecture)** Semua komponen terhubung ke satu jalur bus. Sederhana dan murah, namun memiliki kelemahan berupa bottleneck—ketika dua atau lebih perangkat ingin menggunakan bus secara bersamaan, maka terjadi antrian.
- **Bus Ganda (Multiple Bus Architecture)** Menggunakan dua atau lebih bus, seperti bus sistem dan bus I/O. Pendekatan ini meningkatkan kinerja karena perangkat tidak harus berbagi bus tunggal. Misalnya, sistem modern dapat memiliki **front-side bus** (menghubungkan CPU dan RAM) serta **back-side bus** (untuk cache), atau menggunakan **bus serial berkecepatan tinggi** seperti PCI Express.

Selain arsitektur fisik, manajemen bus juga penting. Beberapa teknik pengendalian bus antara lain **bus arbitration** (penentuan prioritas perangkat untuk mengakses bus) dan **synchronous vs. asynchronous transfer** (berdasarkan sinyal clock atau tidak).

Dengan memahami arsitektur bus, kita dapat melihat bagaimana elemen-elemen dalam komputer bekerja secara sinkron untuk melakukan tugas-tugas komputasi secara cepat, efisien, dan terkoordinasi.

7.2 Jenis-jenis bus: data, alamat, control

Dalam sistem komputer, **bus** merupakan sekumpulan jalur komunikasi (biasanya berupa kawat atau sirkuit dalam papan sirkuit) yang digunakan untuk mentransfer informasi antar komponen perangkat keras. Untuk memastikan komunikasi yang efisien dan terorganisir, bus dalam komputer dibagi menjadi tiga jenis utama berdasarkan fungsinya, yaitu **data bus**, **address bus**, dan **control bus**. Masing-masing memiliki peran penting dalam proses eksekusi instruksi dan pengolahan data di dalam sistem komputer.

1. Data Bus

Data bus berfungsi untuk mentransfer data aktual antara komponen, misalnya dari memori ke CPU atau dari CPU ke perangkat input/output. Data yang dikirim melalui jalur ini bisa berupa nilai numerik, instruksi, maupun hasil perhitungan.

Lebar data bus (jumlah bit yang dapat ditransmisikan secara simultan) sangat menentukan kapasitas pemrosesan sistem. Sebagai contoh, prosesor 32-bit memiliki data bus selebar 32 bit, artinya dapat memindahkan 4 byte data dalam satu siklus clock. Sistem dengan data bus yang lebih lebar (misalnya 64 bit) mampu mentransfer lebih banyak data sekaligus, yang berarti kinerja sistem lebih tinggi dalam hal throughput data.

2. Address Bus

Address bus bertugas mengirimkan alamat memori atau alamat perangkat I/O yang menjadi tujuan atau sumber data. Address bus bersifat **unidirectional** (satu arah), yaitu hanya dari CPU ke memori atau perangkat lain.

Lebar address bus menentukan **kapasitas maksimum memori** yang dapat diakses oleh sistem. Misalnya, address bus 16-bit memungkinkan akses ke 2^{16} (65.536) lokasi memori, sedangkan bus 32-bit mampu mengakses hingga 4 gigabyte (2^{32}

alamat). Ini menjelaskan mengapa sistem 64-bit modern memiliki kemampuan memori jauh lebih besar dibanding sistem 32-bit.

3. Control Bus

Control bus membawa sinyal kontrol yang digunakan untuk mengatur operasi komputer dan mengkoordinasikan aktivitas antar komponen. Sinyal-sinyal ini meliputi:

- **Read (RD)** dan **Write (WR)**: menunjukkan apakah CPU sedang membaca atau menulis data.
- **Clock**: sinkronisasi semua proses dalam sistem komputer.
- **Interrupt**: notifikasi dari perangkat I/O bahwa suatu peristiwa telah terjadi.
- **Bus Request dan Bus Grant**: digunakan dalam sistem dengan banyak perangkat yang berbagi bus.

Control bus bersifat **bidirectional**, karena CPU dan perangkat lain dapat saling mengirim sinyal kontrol.

Ketiga jenis bus ini bekerja secara terpadu dalam sistem komputer. Ketika CPU ingin membaca data dari memori, ia akan mengirimkan alamat melalui address bus, mengaktifkan sinyal kontrol melalui control bus, dan menerima data melalui data bus. Proses ini terjadi sangat cepat dan berulang terus-menerus dalam siklus eksekusi instruksi.

Pemahaman tentang jenis-jenis bus ini penting dalam mendalami bagaimana komunikasi terjadi antar komponen komputer secara terstruktur dan sinkron.

7.3 Interkoneksi perangkat keras

Dalam sebuah sistem komputer, berbagai komponen seperti CPU, memori, perangkat input/output, dan penyimpanan sekunder harus dapat berkomunikasi satu sama lain dengan

lancar. Komunikasi antar komponen ini dimungkinkan melalui suatu sistem yang disebut **interkoneksi perangkat keras**. Interkoneksi ini tidak hanya menghubungkan perangkat secara fisik, tetapi juga mengatur cara data dan sinyal dikirim serta dikendalikan agar dapat berjalan secara sinkron dan efisien.

Interkoneksi perangkat keras pada dasarnya melibatkan tiga elemen utama: **komponen**, **bus komunikasi**, dan **pengendali (controller)**. Komponen-komponen seperti prosesor, RAM, dan perangkat I/O terhubung melalui bus sistem yang terdiri dari data bus, address bus, dan control bus. Masing-masing komponen memiliki pengendali tersendiri yang memungkinkan mereka untuk menginterpretasikan sinyal yang datang dan mengatur perilaku mereka saat berinteraksi dengan komponen lain.

Salah satu tantangan utama dalam interkoneksi perangkat keras adalah perbedaan kecepatan dan protokol komunikasi antar komponen. Misalnya, CPU bekerja dengan kecepatan sangat tinggi, sementara perangkat I/O seperti printer atau hard disk mungkin bekerja jauh lebih lambat. Untuk menjembatani perbedaan ini, sistem komputer menggunakan **pengendali I/O (I/O controller)** dan **buffer**, serta menerapkan metode sinkronisasi agar data tidak hilang atau rusak selama transmisi.

Terdapat berbagai arsitektur interkoneksi yang digunakan dalam sistem komputer, di antaranya:

- **Arsitektur Bus Tunggal (Single-Bus Architecture)**, di mana semua komponen terhubung ke satu jalur bus. Ini sederhana namun mudah menjadi bottleneck karena hanya satu komponen yang dapat berkomunikasi pada satu waktu.
- **Arsitektur Bus Ganda**, yang memisahkan jalur data untuk memori dan I/O, memungkinkan lebih dari satu komunikasi terjadi secara bersamaan.

- **Point-to-Point Interconnect**, seperti yang digunakan dalam **PCI Express (PCIe)**, di mana setiap perangkat memiliki jalur khusus yang langsung terhubung ke prosesor atau chipset. Hal ini mengurangi konflik bus dan meningkatkan kecepatan.

Selain struktur fisiknya, interkoneksi perangkat keras juga melibatkan protokol komunikasi yang mengatur kapan dan bagaimana data dikirim. Protokol ini mencakup aturan handshaking, waktu jeda, sinyal validasi, dan sebagainya.

7.1 DMA (Direct Memory Access)

Dalam sistem komputer, pertukaran data antara perangkat input/output (I/O) dan memori utama biasanya melibatkan CPU sebagai perantara. Namun, pendekatan ini bisa menjadi tidak efisien, terutama ketika volume data yang ditransfer sangat besar, karena CPU harus terlibat dalam setiap langkah transfer. Untuk mengatasi masalah ini, digunakanlah mekanisme yang disebut **DMA (Direct Memory Access)** atau **Akses Memori Langsung**.

DMA adalah metode transfer data di mana perangkat I/O dapat mengakses memori utama secara langsung, tanpa harus melibatkan CPU untuk setiap byte atau word data. Dengan DMA, CPU hanya perlu menginisialisasi proses transfer (misalnya, menentukan alamat sumber dan tujuan serta jumlah data), lalu transfer data dikendalikan oleh sebuah **DMA controller (pengendali DMA)**. Setelah transfer selesai, DMA controller akan memberi sinyal kepada CPU bahwa proses telah rampung melalui **interrupt**.

Cara Kerja DMA

Proses DMA melibatkan beberapa tahap berikut:

1. **Inisialisasi oleh CPU**: CPU mengatur DMA controller dengan informasi seperti alamat memori, jumlah data yang akan ditransfer, dan arah transfer (baca atau tulis).
2. **Pelepasan kendali bus oleh CPU**: Saat DMA aktif, DMA controller mengambil alih bus sistem dari CPU.
3. **Transfer data**: DMA controller mentransfer data secara langsung antara perangkat I/O dan memori utama.
4. **Pemberitahuan selesai**: Setelah selesai, DMA mengirim interrupt ke CPU untuk melanjutkan aktivitas normal.

Keunggulan DMA

- **Efisiensi tinggi**: CPU tidak dibebani oleh proses transfer data, sehingga dapat menjalankan instruksi lain secara paralel.
- **Cepat**: DMA lebih cepat dibandingkan metode transfer biasa karena tidak terganggu oleh siklus instruksi CPU.
- **Ideal untuk transfer besar**: Cocok digunakan untuk perangkat seperti hard disk, kartu suara, kartu grafis, dan jaringan, yang sering mentransfer data dalam jumlah besar.

Jenis DMA

Beberapa mode operasi DMA antara lain:

- **Burst Mode**: DMA mentransfer seluruh blok data sekaligus sebelum mengembalikan kendali bus ke CPU.
- **Cycle Stealing**: DMA mencuri satu siklus bus pada satu waktu, membagi waktu akses dengan CPU.

- **Transparent Mode**: DMA bekerja hanya saat CPU tidak menggunakan bus, sehingga tidak mengganggu eksekusi CPU.

DMA adalah komponen vital dalam arsitektur komputer modern karena meningkatkan efisiensi komunikasi antar perangkat keras, terutama dalam sistem yang memerlukan pemrosesan data real-time atau throughput tinggi.

BAB VIII

SISTEM MASUKAN DAN KELUARAN

8.1 Perangkat I/O dan teknik transfer data

Dalam sistem komputer, **perangkat input/output (I/O)** merupakan komponen penting yang memungkinkan komunikasi antara pengguna, perangkat eksternal, dan sistem komputer itu sendiri. Perangkat **input** berfungsi untuk memasukkan data ke dalam sistem (misalnya keyboard, mouse, scanner, mikrofon), sedangkan perangkat **output** menyajikan hasil pengolahan data dari komputer kepada pengguna (seperti monitor, printer, dan speaker). Beberapa perangkat, seperti touchscreen atau modem, bahkan berfungsi sebagai **perangkat I/O gabungan**.

Interaksi antara perangkat I/O dan sistem komputer tidak sesederhana yang terlihat. Setiap perangkat I/O memiliki karakteristik dan kecepatan kerja yang berbeda-beda, sehingga dibutuhkan **mekanisme transfer data** yang efisien agar komunikasi antar perangkat dapat berlangsung dengan optimal dan tidak menghambat kinerja CPU.

Terdapat beberapa teknik utama dalam proses **transfer data antara perangkat I/O dan memori/CPU**, yaitu:

1. Programmed I/O

Pada metode ini, CPU secara aktif mengontrol proses input/output. CPU akan memeriksa status perangkat dan mentransfer data satu per satu secara langsung. Karena CPU terus-menerus memantau status perangkat (dikenal sebagai **polling**), teknik ini kurang efisien dan membebani prosesor, terutama jika perangkat bekerja lambat.

2. Interrupt-Driven I/O

Teknik ini lebih efisien daripada programmed I/O. CPU tidak perlu terus-menerus memeriksa status perangkat.

Sebaliknya, ketika perangkat I/O siap mentransfer data, ia akan mengirimkan **interrupt** ke CPU. CPU kemudian menjeda tugasnya untuk menangani proses I/O dan melanjutkan tugas utama setelahnya. Teknik ini banyak digunakan karena lebih hemat sumber daya CPU.

3. Direct Memory Access (DMA)

DMA digunakan untuk mentransfer data dalam jumlah besar tanpa melibatkan CPU secara langsung. DMA controller mengelola transfer data langsung antara memori dan perangkat I/O. CPU hanya menginisialisasi dan kemudian bisa melanjutkan proses lain tanpa terganggu. Teknik ini sangat ideal untuk perangkat berkecepatan tinggi seperti disk drive, kartu jaringan, atau kartu grafis.

Pemilihan teknik transfer data sangat bergantung pada jenis perangkat, volume data, dan kebutuhan performa sistem. Dalam banyak kasus, kombinasi dari beberapa metode digunakan agar sistem dapat menyeimbangkan antara efisiensi dan kecepatan.

8.2 Interrupt dan polling

Dalam sistem komputer, **komunikasi antara CPU dan perangkat input/output (I/O)** merupakan bagian penting dari operasi keseluruhan sistem. Karena perangkat I/O memiliki kecepatan kerja yang lebih lambat dibanding CPU, dibutuhkan mekanisme pengaturan yang efisien agar CPU tidak terbuang percuma hanya untuk menunggu data dari perangkat I/O. Dua metode utama yang digunakan dalam pengelolaan komunikasi ini adalah **polling** dan **interrupt**.

Polling

Polling adalah metode di mana **CPU secara aktif dan terus-menerus memeriksa status perangkat I/O** untuk

mengetahui apakah perangkat siap untuk mengirimkan atau menerima data. Proses ini berlangsung dalam bentuk pengecekan loop, biasanya dengan membaca bit status pada port perangkat.

Contoh sederhana polling: CPU akan memeriksa printer setiap beberapa milidetik untuk memastikan apakah kertas telah ditarik dan siap mencetak. Jika belum, CPU akan terus memeriksa sampai perangkat menyatakan siap.

Meskipun polling mudah diimplementasikan dan cocok untuk perangkat sederhana atau sistem real-time dengan kontrol ketat, metode ini **tidak efisien**, karena CPU terpaksa menghabiskan banyak waktu hanya untuk mengecek status perangkat. Ini mengakibatkan pemborosan siklus prosesor, terutama bila terdapat banyak perangkat yang harus diperiksa secara bergiliran.

Interrupt

Interrupt adalah metode yang jauh lebih efisien dibanding polling. Dalam sistem interrupt, **perangkat I/O akan mengirimkan sinyal kepada CPU** hanya saat perangkat membutuhkan perhatian atau siap mentransfer data. Saat sinyal interrupt diterima, CPU akan menghentikan sementara tugas yang sedang dijalankan, menyimpan status eksekusi (konteks), lalu memanggil **Interrupt Service Routine (ISR)**—sebuah subprogram yang menangani peristiwa tersebut.

Setelah proses interrupt selesai, CPU akan melanjutkan tugas sebelumnya seperti semula. Sistem interrupt membuat CPU **bekerja secara reaktif**, bukan aktif, sehingga lebih hemat daya dan efisien. Interrupt juga mendukung **prioritas**, memungkinkan sistem mengatur respons terhadap perangkat berdasarkan urgensi. Misalnya, sinyal dari mouse atau keyboard dapat didahulukan daripada sinyal dari printer.

Polling dan interrupt memiliki kelebihan masing-masing dan dapat digunakan secara selektif tergantung pada jenis

perangkat dan kebutuhan sistem. Sistem modern biasanya menggabungkan keduanya: polling untuk perangkat yang sederhana dan kritikal real-time, serta interrupt untuk perangkat yang jarang aktif namun memerlukan perhatian cepat saat digunakan.

8.3 Manajemen I/O dalam sistem computer

Manajemen I/O (Input/Output) dalam sistem komputer adalah proses pengaturan dan pengendalian interaksi antara CPU, memori, dan berbagai perangkat input/output. Fungsi ini sangat penting karena perangkat I/O memiliki karakteristik, kecepatan, dan kebutuhan yang beragam, sehingga diperlukan sistem yang mampu mengelola semua komunikasi data dengan cara yang efisien, terorganisasi, dan aman.

Tanggung jawab utama dalam manajemen I/O dipegang oleh **sistem operasi (OS)**, yang berperan sebagai penghubung antara perangkat keras I/O dan program aplikasi. Sistem operasi menyediakan **abstraksi perangkat**, sehingga program tidak perlu mengetahui detail teknis dari masing-masing perangkat. Sebaliknya, mereka hanya perlu berinteraksi dengan **driver** atau **API** standar yang disediakan oleh OS.

Tugas Sistem Operasi dalam Manajemen I/O

Beberapa tugas utama sistem operasi dalam manajemen I/O meliputi:

1. **Mengatur alokasi perangkat I/O**
   Sistem operasi memastikan bahwa perangkat I/O digunakan secara bergantian oleh proses-proses dalam sistem. Misalnya, satu printer tidak bisa digunakan oleh dua proses sekaligus.
2. **Menangani komunikasi antar perangkat dan CPU**
   OS mengelola mekanisme seperti interrupt, DMA, dan

polling, serta memilih teknik transfer yang paling sesuai berdasarkan beban sistem dan karakteristik perangkat.

3. **Menyediakan buffering dan spooling**
   Buffering digunakan untuk menyimpan data sementara agar perbedaan kecepatan antara CPU dan perangkat I/O tidak menyebabkan kehilangan data. Spooling (Simultaneous Peripheral Operation On-Line) memungkinkan beberapa permintaan output ditangani dalam antrean, seperti dalam pencetakan dokumen secara bergantian.

4. **Mengatur penjadwalan I/O (I/O scheduling)**
   Sama seperti CPU scheduling, OS dapat menerapkan strategi untuk menentukan urutan penanganan permintaan I/O, seperti **FCFS (First Come, First Served)**, **SSTF (Shortest Seek Time First)**, dan lain-lain, untuk mengoptimalkan waktu respon dan throughput.

5. **Menangani error dan proteksi perangkat**
   Sistem operasi juga bertanggung jawab dalam mendeteksi dan menangani kesalahan pada proses I/O, serta memastikan tidak ada proses yang menyalahgunakan perangkat tertentu.

Dengan sistem manajemen I/O yang baik, komputer mampu menjalankan banyak proses dengan perangkat I/O secara bersamaan (konkurensi), meminimalkan waktu tunggu, dan meningkatkan kinerja sistem secara keseluruhan. Tanpa manajemen I/O yang efisien, sistem akan mudah mengalami bottleneck, konflik perangkat, dan penurunan performa.

8.4 Teknologi I/O modern (USB, PCIe, SATA)

Seiring berkembangnya kebutuhan komputasi yang semakin kompleks dan cepat, teknologi input/output (I/O) juga terus mengalami evolusi. Tujuannya adalah untuk mengatasi

keterbatasan bandwidth, meningkatkan kecepatan transfer data, serta menyederhanakan konektivitas antar perangkat. Di antara berbagai teknologi I/O modern yang digunakan saat ini, tiga yang paling dominan dan penting adalah **USB (Universal Serial Bus)**, **PCIe (Peripheral Component Interconnect Express)**, dan **SATA (Serial ATA)**.

1. USB (Universal Serial Bus)

USB adalah teknologi I/O paling umum yang digunakan untuk menghubungkan berbagai perangkat seperti keyboard, mouse, flash drive, printer, kamera, hingga smartphone. Keunggulan USB terletak pada **kemudahan penggunaan (plug and play)**, **kompatibilitas luas**, dan **kemampuan pengisian daya** sekaligus transfer data.

Sejak diperkenalkan pertama kali, USB telah berevolusi dari **USB 1.0** hingga **USB4**. Versi terbaru mendukung kecepatan hingga **40 Gbps**, mendekati performa bus berkecepatan tinggi lainnya. USB juga mendukung **hot swapping**, artinya perangkat bisa dilepas-pasang tanpa perlu mematikan komputer.

2. PCIe (Peripheral Component Interconnect Express)

PCIe merupakan standar bus internal berkecepatan tinggi yang digunakan untuk menghubungkan **komponen-komponen penting dalam sistem**, seperti kartu grafis (GPU), SSD NVMe, kartu suara, atau kartu jaringan. PCIe menggantikan teknologi PCI lama dengan pendekatan **point-to-point** dan **jalur paralel yang dapat diskalakan** (x1, x4, x8, x16, dst.), sehingga memungkinkan kecepatan transfer data yang jauh lebih besar dan latensi rendah.

Dengan PCIe generasi 5 dan 6, kecepatan data bisa mencapai **64 GT/s (giga transfer per detik)** per jalur. Ini sangat krusial dalam dunia komputasi berat seperti gaming, AI, dan big data.

3. SATA (Serial Advanced Technology Attachment)

SATA adalah antarmuka yang dirancang khusus untuk menghubungkan **perangkat penyimpanan seperti hard disk drive (HDD) dan solid-state drive (SSD)** dengan motherboard. SATA menggantikan antarmuka PATA yang lebih lambat, dengan kecepatan transfer hingga **6 Gbps** (pada SATA III).

SATA terkenal dengan konektivitas yang andal, bentuk fisik yang ringkas, dan kompatibilitas yang baik di lingkungan PC desktop maupun server. Meskipun saat ini SSD NVMe melalui PCIe mulai menggantikan SATA dalam hal kecepatan, SATA tetap populer karena harganya yang terjangkau dan stabilitasnya.

BIOGRAFI PENULIS

Rian Toni Rambe

Lahir di pasar simundol pada tanggal 28 januari 2004. Ia merupakan salah satu mahasiswi aktif di Universitas Labuhanbatu, yang saat ini sedang menempuh pendidikan pada Program Studi Manajemen Informatika, Fakultas Sains dan Teknologi. Ketertarikannya terhadap dunia teknologi informasi dan sistem komputer telah mendorongnya untuk terus mengembangkan pengetahuan dan keterampilan di bidang tersebut, baik melalui pembelajaran akademik maupun kegiatan di luar perkuliahan. Sejak awal menjalani pendidikan tinggi, Rian dikenal sebagai pribadi yang tekun, bertanggung jawab, dan memiliki semangat belajar yang tinggi. Ia memiliki cita-cita besar sebagai programer, sebuah impian yang dilandasi oleh keinginannya untuk berkontribusi dalam dunia teknologi. Oleh karena itu, ia terus mempersiapkan diri dengan memperkuat kemampuan teknis, memperluas wawasan, serta membangun sikap profesional yang dibutuhkan di dunia digital. Dengan komitmen dan tekad yang kuat, Rian toni rambe berusaha menjadikan masa kuliahnya sebagai fondasi yang kokoh untuk mencapai cita-citanya di masa depan.

Deci Irmayani, S.Kom, M.Kom.

Lahir di Rantauprapat pada tanggal 27 Mei 1986. telah menyelesaikan Pendidikan Sarjana (S1) di STMIK Potensi Utama Medan dan melanjutkan Pendidikan Magister (S2) di UPI YPTK Padang. Saat ini, bekerja sebagai Dosen di Universitas Labuhan Batu, Fakultas Sains dan Teknologi, dengan spesialisasi di bidang komputer. berkomitmen untuk memberikan kontribusi dalam bidang pendidikan dan teknologi, serta membagikan pengetahuan kepada generasi mendatang.

Volvo Sihombing, S.Kom, M.Kom.

Lahir di Desa Durian Asahan Sumut, 15 Mei 1985. Telah menyelesaikan Pendidikan Sarjana (S1) di STMIK AKAKOM Yogyakarta 2008 dan melanjutkan Pendidikan Magister (S2) di UPI YPTK Padang. Saat ini, bekerja sebagai Dosen di Universitas Labuhan Batu, Fakultas Sains dan Teknologi, dengan spesialisasi di bidang komputer. berkomitmen untuk memberikan kontribusi dalam bidang pendidikan dan teknologi, serta membagikan pengetahuan kepada generasi mendatang.

SINOPSIS

Arsitektur Komputer: Fondasi Teknologi Digital adalah buku yang dirancang untuk memberikan pemahaman menyeluruh tentang bagaimana sistem komputer bekerja dari dalam. Buku ini membahas elemen-elemen fundamental dalam arsitektur komputer yang menjadi dasar dari seluruh perangkat digital modern, mulai dari struktur CPU, sistem memori, hingga teknologi input/output dan sistem komunikasi data.

Disusun secara sistematis dan mudah dipahami, buku ini mengupas konsep-konsep seperti representasi data, set instruksi, hirarki memori, organisasi bus, hingga mekanisme kerja perangkat I/O secara teknis namun aplikatif. Setiap bab dilengkapi dengan penjelasan rinci, ilustrasi konsep, serta contoh penerapan dalam sistem komputer nyata, termasuk pembahasan teknologi terkini seperti USB, PCIe, DMA, dan manajemen memori virtual.

Buku ini cocok untuk mahasiswa teknik informatika, ilmu komputer, teknik elektro, serta siapa saja yang ingin memahami bagaimana perangkat keras komputer bekerja secara logis dan terstruktur. Dengan pendekatan konseptual yang kuat dan narasi yang mengalir, buku ini tidak hanya membekali pembaca dengan pengetahuan teknis, tetapi juga dengan wawasan untuk memahami fondasi di balik inovasi teknologi digital masa kini.

PENERBIT
YAYASAN MMI

ISBN 978-634-04-1797-5