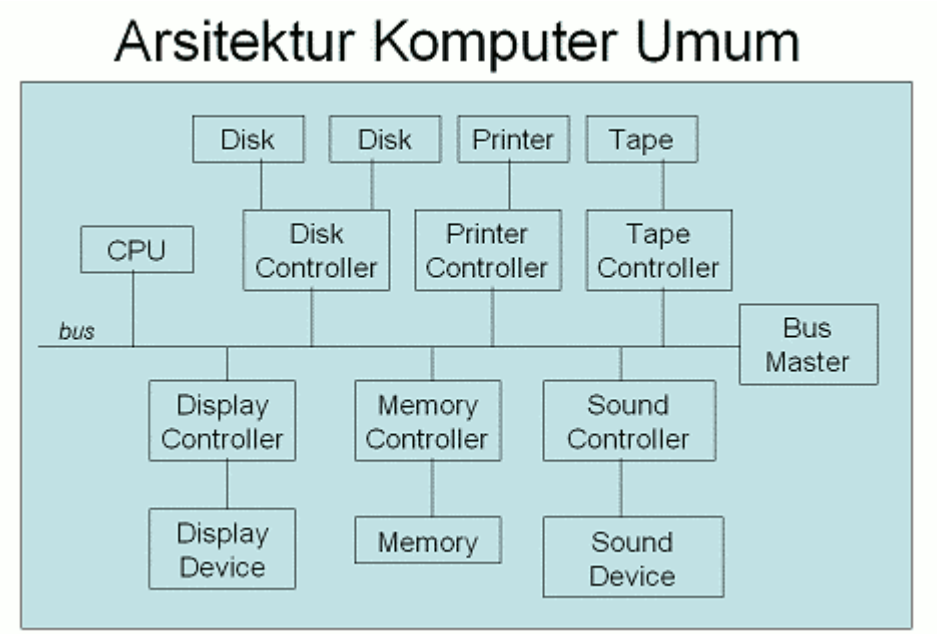
Struktur Sistem Komputer

Tidak ada suatu ketentuan khusus tentang bagaimana seharusnya struktur sistem sebuah komputer. Setiap ahli dan desainer arsitektur komputer memiliki pandangannya masing-masing. Akan tetapi, untuk mempermudah kita memahami detail dari sistem operasi di bab-bab berikutnya, kita perlu memiliki pengetahuan umum tentang struktur sistem komputer.

Operasi Sistem Komputer

Secara umum, sistem komputer terdiri atas CPU dan sejumlah *device controller* yang terhubung melalui sebuah *bus* yang menyediakan akses ke memori. Umumnya, setiap *device controller* bertanggung jawab atas sebuah hardware spesisfik. Setiap *device* dan CPU dapat beroperasi secara konkuren untuk mendapatkan akses ke memori. Adanya beberapa *hardware* ini dapat menyebabkan masalah sinkronisasi. Karena itu untuk mencegahnya sebuah *memory controller* ditambahkan untuk sinkronisasi akses memori.

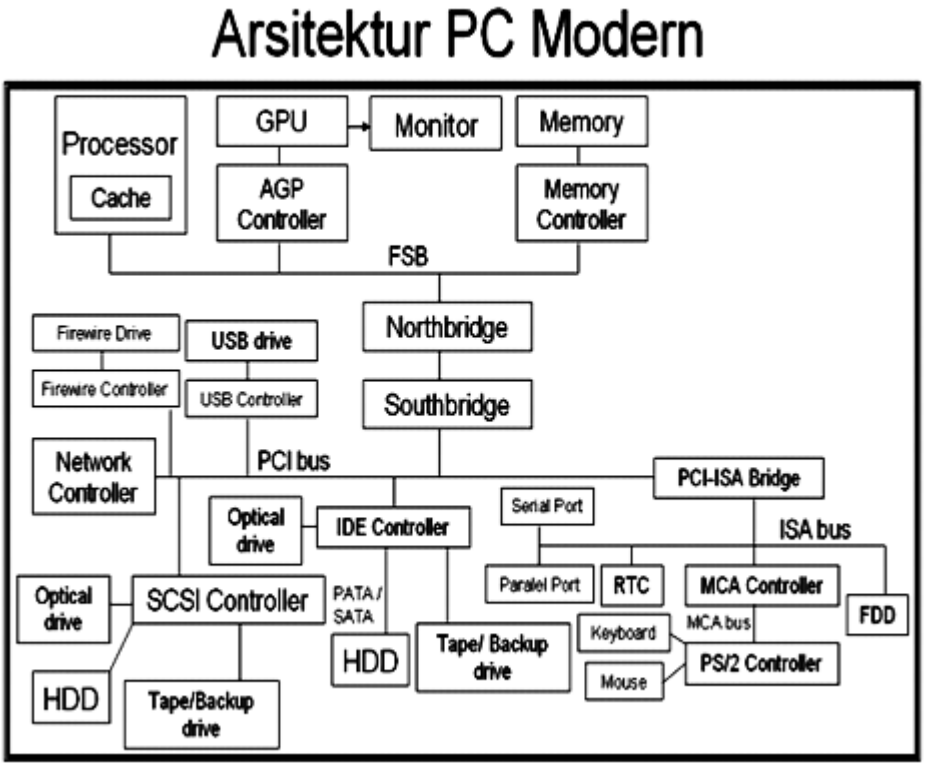


Gambar Arsitektur Umum Komputer

Pada sistem komputer yang lebih maju, arsitekturnya lebih kompleks. Untuk meningkatkan performa, digunakan beberapa buah *bus* . Tiap *bus* merupakan jalur data antara beberapa *device* yang berbeda. Dengan cara ini *RAM, Prosesor, GPU (VGA AGP)* dihubungkan oleh *bus* utama berkecepatan tinggi yang lebih dikenal dengan nama FSB *(Front Side Bus)*. Sementara perangkat lain yang lebih lambat dihubungkan oleh *bus* yang berkecepatan lebih rendah yang terhubung dengan *bus* lain yang lebih cepat sampai ke bus utama. Untuk komunikasi antar bus ini digunakan sebuah *bridge* .

Tanggung jawab sinkronisasi *bus* yang secara tak langsung juga mempengaruhi sinkronisasi memori dilakukan oleh sebuah *bus controller* atau

dikenal sebagai *bus master*. *Bus master* akan mengendalikan aliran data hingga pada satu waktu, bus hanya berisi data dari satu buah *device*. Pada prakteknya *bridge* dan *bus master* ini disatukan dalam sebuah *chipset* .



Gambar Arsitektur PC Modern

Keterangan: *GPU* = Graphics Processing Unit; *AGP* = Accelerated Graphics Port; *HDD* = Hard Disk Drive; *FDD* = Floppy Disk Drive; *FSB* = Front Side Bus; *USB* = Universal Serial Bus; *PCI* = Peripheral Component Interconnect; *RTC* = Real Time Clock; *PATA* = Pararel Advanced Technology Attachment; *SATA* = Serial Advanced Technology Attachment; *ISA* = Industry Standard Architecture; *IDE* = Intelligent Drive Electronics/Integrated Drive Electronics; *MCA* = Micro Channel Architecture; *PS/2* =Sebuah *port* yang dibangun IBM untuk menghubungkan mouse ke *PC*;

Jika komputer dinyalakan, yang dikenal dengan nama *booting,* komputer akan menjalankan *bootstrap program* yaitu sebuah program sederhana yang disimpan dalam ROM yang berbentuk chip *CMOS (Complementary Metal Oxide Semiconductor)*. Chip CMOS modern biasanya bertipe *EEPROM (Electrically Erasable Programmable Read Only Memory),* yaitu memori *non-volatile* (tak terhapus jika power dimatikan) yang dapat ditulis dan dihapus dengan pulsa elektronik. Lalu *bootsrap program* ini lebih dikenal sebagai *BIOS (Basic Input Output System)*.

Bootstrap program utama, yang biasanya terletak di *Motherboard* akan memeriksa *hardware-hardware* utama dan melakukan inisialisasi terhadap program dalam *hardware* yang dikenal dengan nama *firmware*.

Bootstrap program utama kemudian akan mencari dan meload *kernel* sistem operasi ke memori lalu dilanjutkan dengan inisialisasi sistem operasi.Dari sini program sistem operasi akan menunggu kejadian tertentu. Kejadian ini akan menentukan apa yang akan dilakukan sistem operasi berikutnya (*event-driven*).

Kejadian ini pada komputer modern biasanya ditandai dengan munculnya *interrupt* dari software atau hardware, sehingga Sistem Operasi ini disebut *Interrupt-driven. Interrupt* dari *hardware* biasanya dikirimkan melalui suatu signal tertentu, sedangkan *software* mengirim *interrupt* dengan cara menjalankan *system call* atau juga dikenal dengan istilah *monitor call . System/Monitor call* ini akan menyebabkan *trap* yaitu *interrupt* khusus yang dihasilkan oleh software karena adanya masalah atau permintaan terhadap layanan sistem operasi. *Trap* ini juga sering disebut sebagai *exception* .

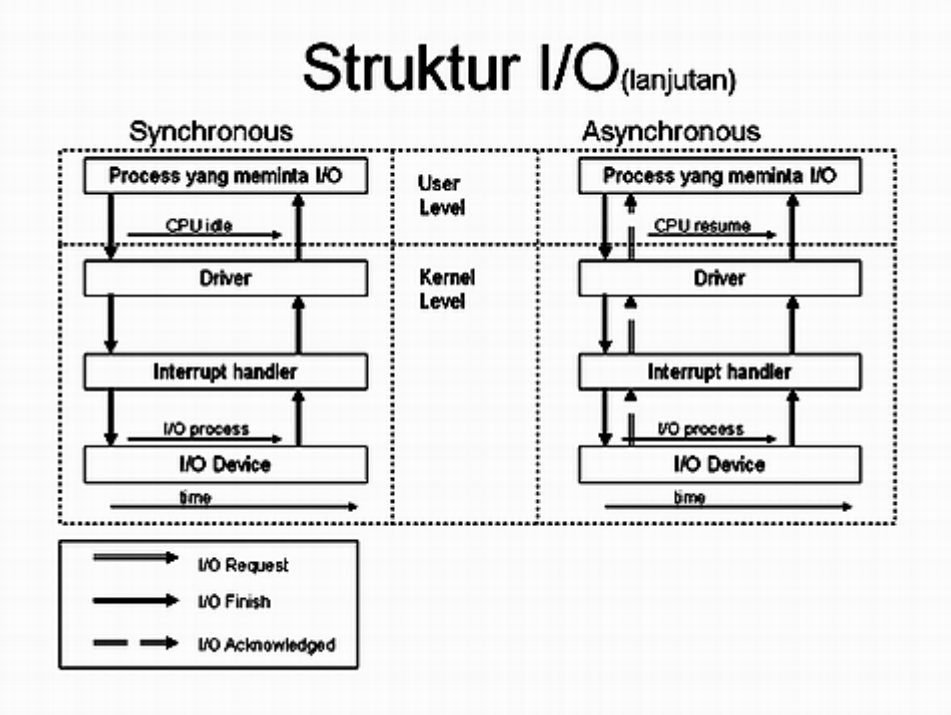
Setiap *interrupt* terjadi, sekumpulan kode yang dikenal sebagai *ISR (Interrupt Service Routine)* akan menentukan tindakan yang akan diambil. Untuk menentukan tindakan yang harus dilakukan, dapat dilakukan dengan dua cara yaitu *polling* yang membuat komputer memeriksa satu demi satu perangkat yang ada untuk menyelidiki sumber *interrupt* dan dengan cara menggunakan alamat-alamat *ISR* yang disimpan dalam array yang dikenal sebagai *interrupt vector* di mana sistem akan memeriksa *Interrupt Vector* setiap kali *interrupt* terjadi.

Arsitektur *interrupt* harus mampu untuk menyimpan alamat instruksi yang di- *interrupt*. Pada komputer lama, alamat ini disimpan di tempat tertentu yang tetap, sedangkan padakomputer baru, alamat itu disimpan di *stack* bersama-sama dengan informasi state saat itu.

Struktur I/O

Ada dua macam tindakan jika ada operasi *I/O* . Kedua macam tindakan itu adalah:

- Setelah proses I/O dimulai, kendali akan kembali ke user program saat proses I/O selesai *(Synchronous).* Instruksi wait menyebabkan CPU idle sampai *interrupt* berikutnya. Akan terjadi *Wait loop* (untuk menunggu akses berikutnya). Paling banyak satu proses I/O yang berjalan dalam satu waktu.
- Kemudian kendali akan kembali ke user program tanpa menunggu proses I/O selesai (Asynchronous). System call permintaan pada sistem operasi untuk mengizinkan user menunggu sampai I/O selesai.Device-status table mengandung data masukkan untuk tiap I/O device yang menjelaskan tipe, alamat, dan keadaannya. Sistem operasi memeriksa I/O device untuk mengetahui keadaan device dan mengubah tabel untuk memasukkan *interrupt.* Jika I/O device mengirim/mengambil data ke/dari memory hal ini dikenal dengan nama (Direct Memory Access) DMA.



Gambar Struktur I/O

Direct Memory Access

Digunakan untuk *I/O device* yang dapat memindahkan data dengan kecepatan tinggi (mendekati frekuensi bus memori). *Device controller* memindahkan data dalam blok-blok dari buffer langsung ke memory utama atau sebaliknya tanpa campur tangan prosesor. *Interrupt* hanya terjadi tiap blok bukan tiap word atau byte data. Seluruh proses DMA dikendalikan oleh sebuah controller bernama *DMA Controller (DMAC)*. *DMA Controller* mengirimkan atau menerima signal dari memori dan *I/O device.* Prosesor hanya mengirimkan alamat awal data, tujuan data, panjang data ke *DMA Controller*. *Interrupt* pada prosesor hanya terjadi saat proses transfer selesai. Hak terhadap penggunaan *bus memory* yang diperlukan *DMA controller* didapatkan dengan bantuan *bus arbiter* yang dalam PC sekarang berupa *chipset Northbridge* .

Bus

Suatu jalur transfer data yang menghubungkan setiap *device* pada komputer. Hanya ada satu buah *device* yang boleh mengirimkan data melewati sebuah bus, akan tetapi boleh lebih dari satu *device* yang membaca data bus tersebut. Terdiri dari dua buah model: *Synchronous bus* di mana digunakan dengan bantuan clock tetapi berkecepatan tinggi, tapi hanya untuk device berkecepatan tinggi juga; *Asynchronous bus* digunakan dengan sistem *handshake* tetapi berkecepatan rendah, dapat digunakan untuk berbagai macam *device* .

Struktur Storage

Hal penting yang perlu diingat adalah program adalah bagian dari data.

1. Register

Tempat penyimpanan beberapa buah data *volatile* yang akan diolah langsung di prosesor yang berkecepatan sangat tinggi. Register ini berada di dalam prosesor dengan jumlah yang sangat terbatas karena fungsinya sebagai tempat perhitungan/komputasi data

2. Cache Memory

Tempat penyimpanan sementara (*volatile*) sejumlah kecil data untuk meningkatkan kecepatan pengambilan atau penyimpanan data di memori oleh prosesor yang berkecepatan tinggi. Dahulu *cache* disimpan di luar prosesor dan dapat ditambahkan. Misalnya *pipeline burst* cache yang biasa ada di komputer awal tahun 90-an. Akan tetapi seiring menurunnya biaya produksi *die* atau *wafer* dan untuk meningkatkan kinerja, *cache* ditanamkan di prosesor. Memori ini biasanya dibuat berdasarkan desain *static memory.*

3. Random Access Memory (RAM) - Main Memory

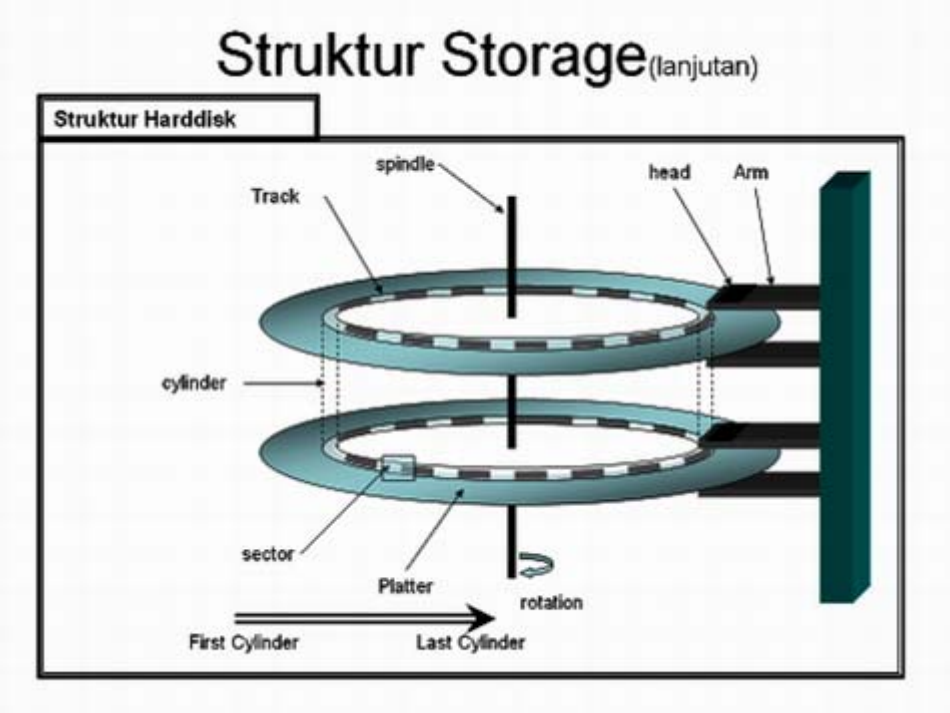
Tempat penyimpanan sementara sejumlah data *volatile* yang dapat diakses langsung oleh prosesor. Pengertian langsung di sini berarti prosesor dapat mengetahui alamat data yang ada di memori secara langsung. Sekarang, *RAM* dapat diperoleh dengan harga yang cukup murah dangan kinerja yang bahkan dapat melewati *cache* pada komputer yang lebih lama.

4. Extension Memory

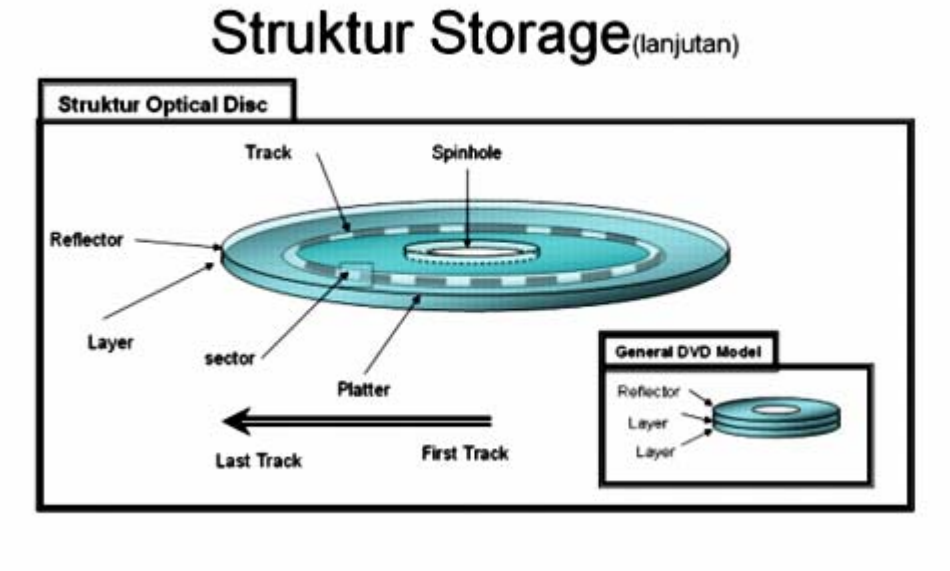
Tambahan memory yang digunakan untuk membantu proses-proses dalam komputer, biasanya berupa buffer. Peranan tambahan memori ini sering dilupakan akan tetapi sangat penting artinya untuk efisiensi. Biasanya tambahan memori ini memberi gambaran kasar kemampuan dari perangkat tersebut, sebagai contoh misalnya jumlah *VGA memory, soundcard memory.*

Secondary Storage

Media penyimpanan data yang non-volatile yang dapat berupa *Flash Drive, Optical Disc, Magnetic Disk, Magnetic Tape.* Media ini biasanya daya tampungnya cukup besar dengan harga yang relatif murah. *Portability-nya* juga relatif lebih tinggi.



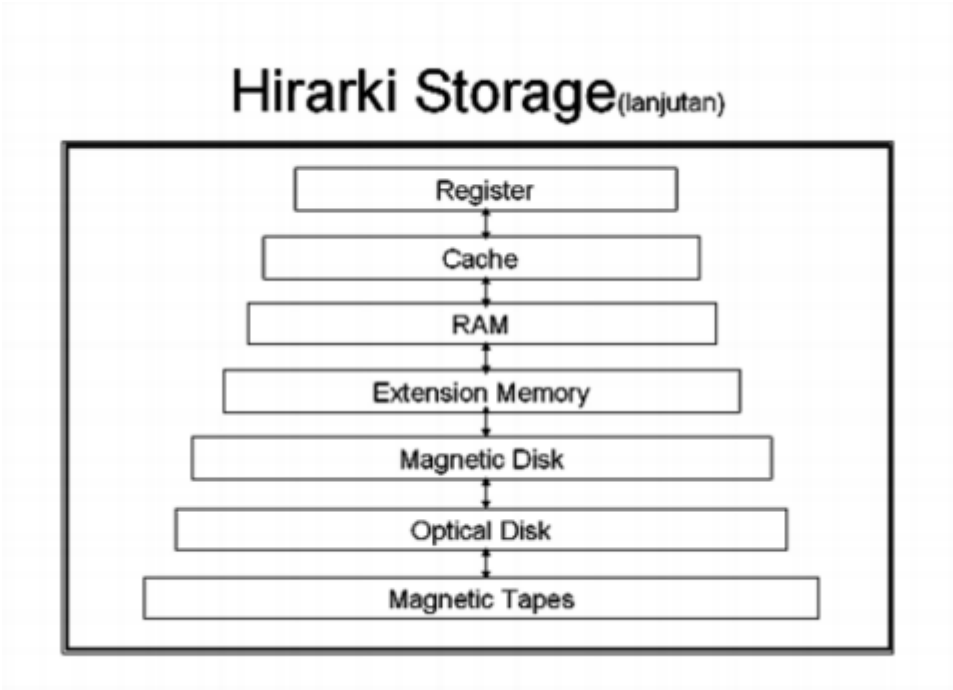
Gambar Struktur Harddisk



Gambar Struktur Optical Drive

Hirarki Storage

Dasar susunan sistem storage adalah kecepatan, biaya, sifat volatilitas. *Caching* menyalin informasi ke *storage media* yang lebih cepat; Main memory dapat dilihat sebagai cache terakhir untuk *secondary storage* . Menggunakan memory berkecepatan tinggi untuk memegang data yang diakses terakhir. Dibutuhkan *cache management policy. Cache* juga memperkenalkan tingkat lain di hirarki storage. Hal ini memerlukan data untuk disimpan bersama-sama di lebih dari satu level agar tetap konsisten.



Gambar Hirarki storage

Referensi

1. Harry Helms, “**Computer Handbook”**, McGraw-Hill, 1983.
2. Peter Denning, "**Computer Science: the Discipline**", In *Encyclopedia of Computer Science* (A. Ralston and D. Hemmendinger, Eds), 1999.
3. V. Carl Hamacher, Zvonko G. Vranesic, Safwat G. Zaky, **Computer Organization** (5th Edition), McGraw-Hill, 2001.
4. IEEE Computer Society (IEEE CS), http://computer.org