Computer Architecture
Basic Concepts and Performance

Dr. Nausheen Shoaib

Book: Compute Organization and Architecture

Computer Architecture

ARM Achitecture

Figure 1.16 provides a block diagram of the EFM32 microcontroller from Silicon Labs with Cortex-M3 processor and core components.

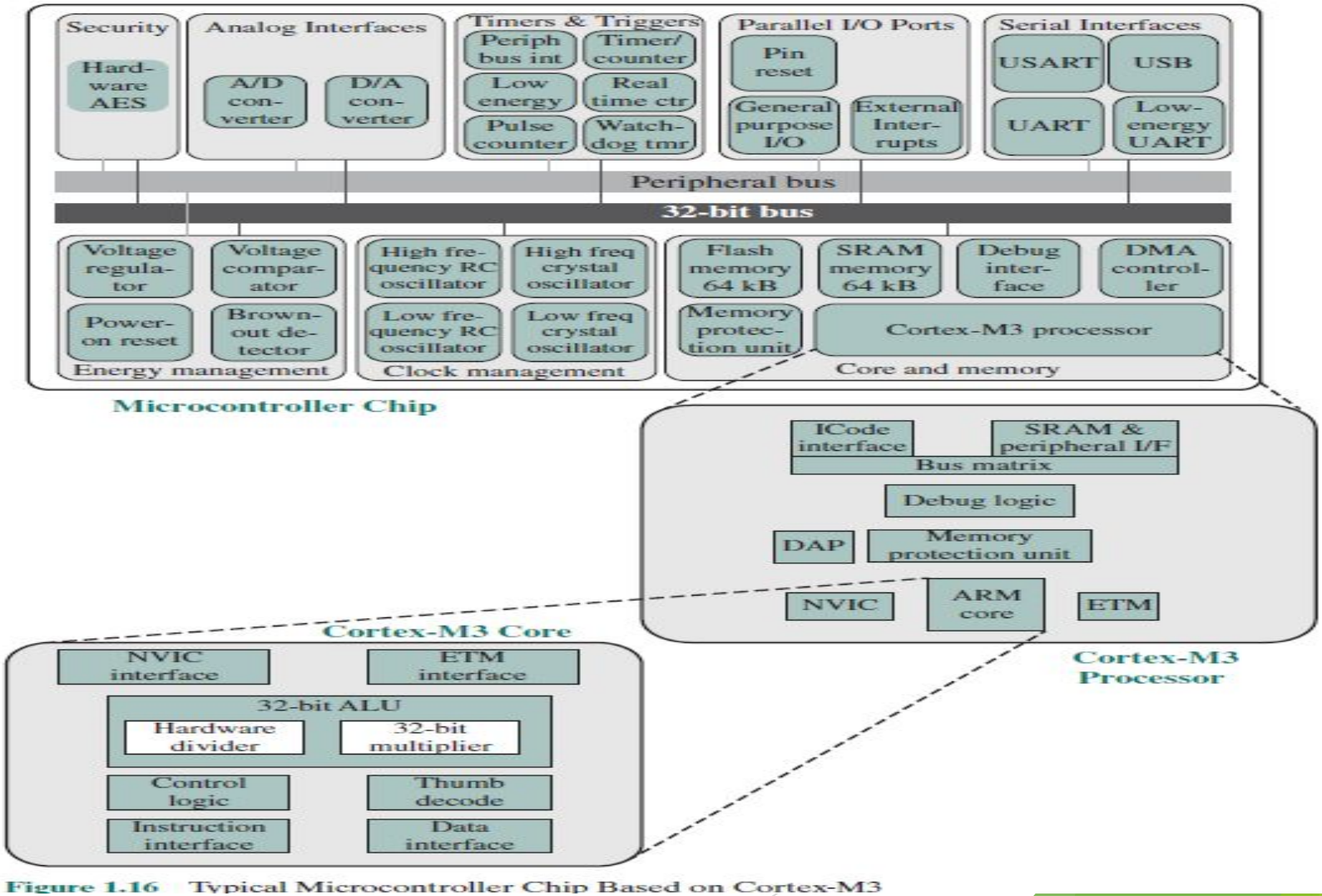

Figure 1.16 Typical Microcontroller Chip Based on Cortex-M3

ARM Achitecture

Figure 1.16 shows microcontroller built with the Cortex-M3.This microcontroller is used in a wide variety of devices, including energy, gas, and water metering; alarm and security systems; industrial automation devices; home automation devices; smart accessories; and health and fitness devices. The silicon chip consists of 10 main areas:

Core and memory: This region includes the Cortex-M3 processor, static RAM (SRAM) data memory, and flash memory for storing program instructions and nonvarying application data.

Flash memory is nonvolatile (data is not lost when power is shut off) and so is ideal for this purpose. Flash memory is a versatile form of memory used both in microcontrollers and as external memoryParallel I/O ports: Configurable for a variety of parallel I/O schemes.

SRAM stores variable data. This area also includes a debug interface, which makes it easy to reprogram and update the system in the field.Static RAM (SRAM) is a form of random-access memory used for cache memory.

Serial interfaces: Supports various serial I/O schemes.

https://www.techtarget.com/whatis/definition/SRAM-static-random-access-memory#:~:text=SRAM%20(static%20RAM)%20is%20a,performance%20and%20lower%20power%20usage.

ARM Achitecture

Analog interfaces: Analog-to-digital and digital-to-analog logic to support sensors and actuators.

Timers and triggers: Keeps track of timing and counts events, generates output waveforms, and triggers timed actions in other peripherals.

Clock management: Controls the clocks and oscillators on the chip. Multiple clocks and oscillators are used to minimize power consumption and provide short startup times.

Energy management: Manages the various low-energy modes of operation of the processor and peripherals to provide real-time management of the energy needs so as to minimize energy consumption.

Security: The chip includes a hardware implementation of the Advanced Encryption Standard (AES).

32-bit bus: Connects all of the components on the chip.

Peripheral bus: A network which lets the different peripheral modules communicate directly with each other without involving the processor. This supports timing-critical operation and reduces software overhead.

ARM Achitecture

watchdog timer (WDT) is a timer that monitors microcontroller (MCU) programs to see if they are out of control or have stopped operating.

USART (universal synchronous/asynchronous receiver/transmitter) is hardware that enables a device to communicate using serial protocols

UART (Universal Asynchronous Receiver/Transmitter) is the microchip with programming that controls a computer's interface to its attached serial devices.

Low Energy UART (LEUART) provides full UART communication running from a 32.768 kHz clock input.

Electronic oscillator is an electronic circuit that produces a periodic, oscillating or alternating current (AC) signal, usually a sine wave, square wave or a triangle wave.RC and crystal oscillator used for clock management.

Voltage comparator compares two input voltages and outputs a binary signal indicating which is larger.

ARM Achitecture

Voltage regulator is a circuit that creates and maintains a fixed output voltage, irrespective of changes to the input voltage or load conditions. Voltage regulators (VRs) keep the voltages from a power supply within a range that is compatible with the other electrical components.

Power-on reset (PoR) is a circuit that provides a predictable, regulated voltage to a microprocessor or microcontroller with the initial application of power. The PoR system ensures that the microprocessor or microcontroller will start in the same condition every time that it's powered up.

Brown Out Reset A “brown out” of a microcontroller is a temporary reduction in the power supply voltage below the level required for reliable operation.

Many microcontrollers have a protection circuit which detects when the supply voltage goes below this level and puts the device into a reset state to ensure proper startup when power returns. This action is called a “Brown Out Reset”.

A similar feature is called Low Voltage Detect (LVD) which is more complex and adds detection of multiple voltage levels and can produce an interrupt before a reset is triggered.

ARM Achitecture

External interrupts are caused by some external event or failure.

DMA Controller is a type of control unit that works as an interface for the data bus and the I/O Devices. DMA Controller has the work of transferring the data without the intervention of the processors

ARM Achitecture

Cortex-M3 core use separate buses for instructions and data,this arrangement is referred to as a **Harvard architecture.**

In comparison with the **von Neumann architecture**, which uses the same signal buses and memory for both instructions and data.

Cotext-M3 processor read both an instruction and data from memory at the same time, perform many operations in parallel, speeding application execution.

The core contains a decoder for Thumb instructions, an advanced ALU with support for hardware multiply and divide, control logic, and interfaces to the other components of the processor. The processor includes the following elements:

NVIC: Provides configurable interrupt handling abilities to the processor. It facilitates low-latency exception and interrupt handling, and controls power management.

ETM: An optional debug component that enables reconstruction of program execution. The ETM is designed to be a high-speed, low-power debug tool that only supports instruction trace.

ARM Achitecture

Debug access port (DAP): This provides an interface for external debug access to the processor.

Debug logic: Basic debug functionality includes processor halt, single-step, processor core register access, unlimited software breakpoints, and full system memory access.

Single step: allows a reverse engineer to execute a single instruction at a time before returning control to the debugger. This feature used when one needs to analyze a binary by executing a single instruction or a section of instructions

ICode interface: Fetches instructions from the code memory space.

SRAM & peripheral interface: Read/write interface to data memory and peripheral devices.

Bus matrix: Connects the core and debug interfaces to external buses on the microcontroller.

Memory protection unit: Protects critical data used by operating system from user applications, separating processing tasks by disallowing access to each other's data, disabling access to memory regions, allowing memory regions to be defined as read-only, and detecting unexpected memory accesses that could potentially break the system.

Classes of Computer

Figure 1.2 summarizes these mainstream classes of computing environments and their important characteristics.

Feature	Personal mobile device (PMD)	Desktop	Server	Clusters/warehouse-scale computer	Internet of things/ embedded
Price of system	$100–$1000	$300–$2500	$5000–$10,000,000	$100,000–$200,000,000	$10–$100,000
Price of microprocessor	$10–$100	$50–$500	$200–$2000	$50–$250	$0.01–$100
Critical system design issues	Cost, energy, media performance, responsiveness	Price-performance, energy, graphics performance	Throughput, availability, scalability, energy	Price-performance, throughput, energy proportionality	Price, energy, application-specific performance

Figure 1.2 A summary of the five mainstream computing classes and their system characteristics. Sales in 2015 included about 1.6 billion PMDs (90% cell phones), 275 million desktop PCs, and 15 million servers. The total number of embedded processors sold was nearly 19 billion. In total, 14.8 billion ARM-technology-based chips were shipped in 2015. Note the wide range in system price for servers and embedded systems, which go from USB keys to network routers. For servers, this range arises from the need for very large-scale multiprocessor systems for high-end transaction processing.

Classes of Computer

Personal Mobile Devices: Personal mobile device is the term to a collection of wireless devices with multimedia user interfaces such as cell phones, tablet computers,and so on.

Processors in a PMD are often considered embedded computers. PMDs are platforms that can run externally developed software, and share many of the characteristics of desktop computer.

For example, in playing a video on a PMD, the time to process each video frame is limited, since the processor must accept and process the next frame shortly.

Servers: Consider the servers running ATM machines for banks or airline reservation systems. Failure of such server systems is catastrophic than failure of a single desktop. Key feature of server systems are scalability, availability, efficient throughput.

Responsiveness to an individual request remains important,how many requests can be handled in a unit time.

Classes of Computer

Clusters / Warehouse Scale Computers: Clusters are collections of desktop computers or servers connected by local area networks to act as a single larger computer. Each node runs its own operating system, and nodes communicate using a networking protocol.

WSCs are the largest of the clusters with tens of thousands of servers. Scalability for a WSC is handled by the LAN connecting the computers and not by integrated computer hardware, as in the case of servers.

Supercomputers are related to WSCs,differ by emphasizing floating-point performance and by running large, communication-intensive batch programs that can run for weeks at a time.

WSCs emphasize interactive applications, large-scale storage, dependability, and high Internet bandwidth.

Classes of Parallelism and Parallel Architectures

There are basically two kinds of parallelism in applications:

Data Parallelisms	Task Parallelisms
1. Same task are performed on different subsets of same data.	1. Different task are performed on the same or different data.
2. Synchronous computation is performed.	2. Asynchronous computation is performed.
3. As there is only one execution thread operating on all sets of data, so the speedup is more.	3. As each processor will execute a different thread or process on the same or different set of data, so speedup is less.
4. Amount of parallelization is proportional to the input size.	4. Amount of parallelization is proportional to the number of independent tasks is performed.
5. It is designed for optimum load balance on multiprocessor system.	5. Here, load balancing depends upon on the e availability of the hardware and scheduling algorithms like static and dynamic scheduling.

Classes of Parallelism and Parallel Architectures

Computer hardware in turn can exploit these two kinds of application parallelism in four major ways:

1. Instruction-level parallelism: exploits data-level parallelism at modest levels with compiler help using ideas like pipelining and at medium levels.

2. Vector architectures, graphic processor units (GPUs), and multimedia instruction sets: exploit data-level parallelism by applying a single instruction to a collection of data in parallel.

3. Thread-level parallelism: exploits either data-level parallelism or task-level parallelism in a tightly coupled hardware model that allows for interaction between parallel threads.

4. Request-level parallelism: exploits parallelism among largely decoupled tasks specified by the programmer or the operating system.

Classes of Parallelism and Parallel Architecture

Flynn's Taxanomy

Single instruction stream, single data stream (SISD): This category is the uniprocessor. The programmer thinks of it as the standard sequential computer, but it can exploit ILP.

Single instruction stream, multiple data streams (SIMD): The same instruction is executed by multiple processors using different data streams. SIMD computers exploit data-level parallelism by applying the same operations to multiple items of data in parallel. Each processor has its own data memory, but there is a single instruction memory and control processor, which fetches and dispatches instructions.

Multiple instruction streams, single data stream (MISD): No commercial multiprocessor of this type has been built to date, but it rounds out this simple classification.

Multiple instruction streams, multiple data streams (MIMD): Each processor fetches its own instructions and operates on its own data, and it targets task-level parallelism. For example, MIMD computers can also exploit data-level parallelism, although the overhead is likely to be higher than would be seen in an SIMD computer. This overhead means that grain size must be sufficiently large to exploit the parallelism efficiently.

Trends in Power and Energy in Integrated Circuit

For CMOS chips, the traditional primary energy consumption has been in switching transistors, also called *dynamic energy*. The energy required per transistor is proportional to the product of the capacitive load driven by the transistor and the square of the voltage:

$$\text{Energy}_{\text{dynamic}} \propto \text{Capacitive load} \times \text{Voltage}^2$$

This equation is the energy of pulse of the logic transition of $0 \rightarrow 1 \rightarrow 0$ or $1 \rightarrow 0 \rightarrow 1$. The energy of a single transition ($0 \rightarrow 1$ or $1 \rightarrow 0$) is then:

$$\text{Energy}_{\text{dynamic}} \propto 1/2 \times \text{Capacitive load} \times \text{Voltage}^2$$

The power required per transistor is just the product of the energy of a transition multiplied by the frequency of transitions:

$$\text{Power}_{\text{dynamic}} \propto 1/2 \times \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency switched}$$

For a fixed task, slowing clock rate reduces power, but not energy.

Clearly, dynamic power and energy are greatly reduced by lowering the voltage, so voltages have dropped from 5 V to just under 1 V in 20 years. The capacitive load is a function of the number of transistors connected to an output and the technology, which determines the capacitance of the wires and the transistors.

Trends in Power and Energy in Integrated Circuit

Example#1: Some microprocessors today are designed to have adjustable voltage, so a 15% reduction in voltage may result in a 15% reduction in frequency. What would be the impact on dynamic energy and on dynamic power?

Trends in Power and Energy in Integrated Circuit

Modern microprocessors offer many techniques to improve energy efficiency :

1. Do nothing well: Most microprocessors today turn off the clock of inactive modules to save energy and dynamic power. For example, if no floating-point instructions are executing, the clock of the floating-point unit is disabled. If some cores are idle, their clocks are stopped.

2. Dynamic voltage-frequency scaling (DVFS). PMDs, laptops, and even servers have periods of low activity where there is no need to operate at the highest clock frequency and voltages. Modern microprocessors typically offer a few clock frequencies and voltages in which to operate that use lower power and energy.Figure 1.12 plots the potential power savings via DVFS for a server as the workload shrinks for three different clock rates: 2.4, 1.8, and 1 GHz. The overall server power savings is about 10%–15% for each of the two steps.

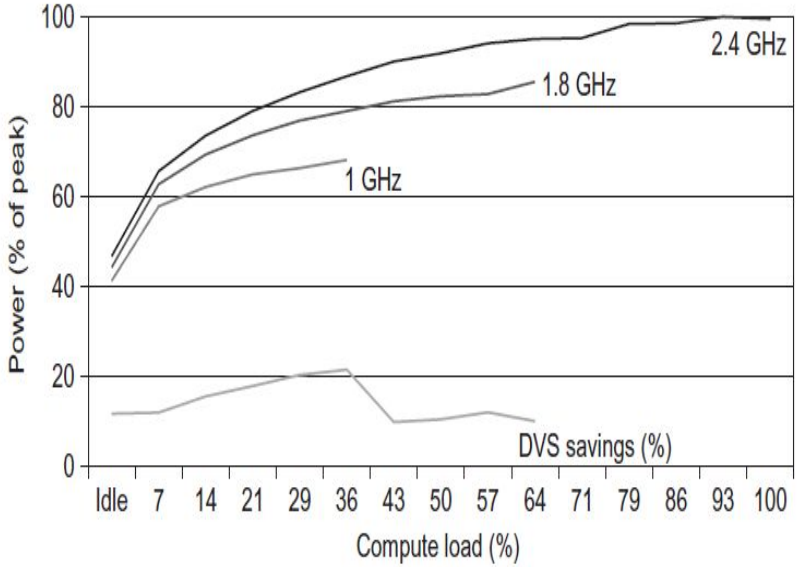

Figure 1.12 Energy savings for a server using an AMD Opteron microprocessor, 8 GB of DRAM, and one ATA disk. At 1.8 GHz, the server can handle at most up to two-thirds of the workload without causing service-level violations, and at 1 GHz, it can safely handle only one-third of the workload (Figure 5.11 in Barroso and Hölzle, 2009).

Trends in Power and Energy in Integrated Circuit

3.Design for the typical case: Given that PMDs and laptops are often idle, memory and storage offer low power modes to save energy. For example, DRAMs have a series of increasingly lower power modes to extend battery life in PMDs and laptops, and there have been proposals for disks that have a mode that spins more slowly when unused to save power.

However, you cannot access DRAMs or disks in these modes, so you must return to fully active mode to read or write, no matter how low the access rate.

Microprocessors for PCs have been designed instead for heavy use at high operating temperatures, relying on on-chip temperature sensors to detect when activity should be reduced automatically to avoid overheating.

Trends in Power and Energy in Integrated Circuit

4.Overclocking: Intel started offering Turbo mode in 2008, where the chip decides that it is safe to run at a higher clock rate for a short time, possibly on just a few cores, until temperature starts to rise.

For single-threaded code, these microprocessors can turn off all cores but one and run it faster. Although the operating system can turn off Turbo mode, there is no notification once it is enabled, so the programmers may be surprised to see their programs vary in performance.

Although dynamic power is traditionally thought of as the primary source of power dissipation in CMOS, static power is becoming an important issue because leakage current flows even when a transistor is off:

$$\text{Power}_{\text{static}} \propto \text{Current}_{\text{static}} \times \text{Voltage}$$

Trends in Cost

A wafer is tested and chopped into dies that are packaged. Therefore the cost of a packaged integrated circuit is:

$$\text{Cost of integrated circuit} = \frac{\text{Cost of die} + \text{Cost of testing die} + \text{Cost of packaging and final test}}{\text{Final test yield}}$$

The number of dies per wafer is approximately the area of the wafer divided by the area of the die. It can be more accurately estimated by:

$$\text{Cost of die} = \frac{\text{Cost of wafer}}{\text{Dies per wafer} \times \text{Die yield}}$$

The first term is the ratio of wafer area (πr2) to die area. The second compensates for the "square peg in a round hole" problem—rectangular dies near the periphery of round wafers. Dividing the circumference (πd) by the diagonal of a square die is approximately the number of dies along the edge.

$$\text{Dies per wafer} = \frac{\pi \times (\text{Wafer diameter}/2)^2}{\text{Die area}} - \frac{\pi \times \text{Wafer diameter}}{\sqrt{2 \times \text{Die area}}}$$

Trends in Cost

this formula gives only the maximum number of dies per wafer. The critical question is: What is the fraction of good dies on a wafer, or the die yield? A simple model of integrated circuit yield, which assumes that defects are randomly distributed over the wafer and that yield is inversely proportional to the complexity of the fabrication process, leads to the following:

$$\text{Die yield} = \text{Wafer yield} \times 1/(1 + \text{Defects per unit area} \times \text{Die area})^N$$

This Bose-Einstein formula is an empirical model developed by looking at the yield of many manufacturing.

Wafer yield accounts for wafers that are completely bad and so need not be tested. we'll just assume the wafer yield is 100%.

Defects per unit area is a measure of the random manufacturing defects that occur. The value was typically 0.08–0.10 defects per square inch for a 28-nm node and 0.10–0.30 for the newer 16 nm node.

Finally, N is a parameter called the process-complexity factor, a measure of manufacturing difficulty. For 28 nm processes, N is 7.5–9.5. For a 16 nm process, N ranges from 10 to 14.

Trends in Cost

Example#2: Find the number of dies per 300 mm (30 cm) wafer for a die that is 1.5 cm on a side and for a die that is 1.0 cm on a side.

Dependability

One difficult question is deciding when a system is operating properly. Infrastructure providers started offering service level agreements (SLAs) or service level objectives (SLOs) to guarantee that their networking or power service would be dependable.

For example, they would pay the customer a penalty if they did not meet an agreement of some hours per month. SLA could be used to decide whether the system was up or down.

Systems alternate between two states of service with respect to an SLA:

1. Service accomplishment, where the service is delivered as specified.

2. Service interruption, where the delivered service is different from the SLA.

Dependability

Transitions between these two states are caused by failures or restorations. Quantifying these transitions leads to the two main measures of dependability:

Module reliability: is a measure of the continuous service accomplishment (or time to failure) from a reference initial instant.

Mean time to failure (MTTF) is a reliability measure. The reciprocal of MTTF is a rate of failures, generally reported as failures per billion hours of operation, or FIT (for failures in time).

Service interruption is measured as mean time to repair (MTTR).

Mean time between failures (MTBF) is simply the sum of MTTF+MTTR.

Module availability: is a measure of the service accomplishment with respect to the alternation between the two states of accomplishment and and interruption. For nonredundant systems with repair, module availability is

$$\text{Module availability} = \frac{\text{MTTF}}{(\text{MTTF} + \text{MTTR})}$$

Dependability

Example#3: Assume a disk subsystem with the following components and MTTF:

- 10 disks, each rated at 1,000,000-hour MTTF
- 1 ATA controller, 500,000-hour MTTF
- 1 power supply, 200,000-hour MTTF
- 1 fan, 200,000-hour MTTF
- 1 ATA cable, 1,000,000-hour MTTF

Using the simplifying assumptions that the lifetimes are exponentially distributed and that failures are independent, compute the MTTF of the system as a whole.

Designing for Performnce

Desktop applications that require the great power of today's microprocessor-based systems include: Image processing, Three-dimensional rendering, Speech recognition,Video conferencing, Multimedia authoring, Voice and video annotation of files, and Simulation modeling.

Workstation systems now support highly sophisticated engineering and scientific applications and have the capacity to support image and video applications.

Businesses are relying on increasingly powerful servers to handle transaction and database processing and to support massive client/server networks that have replaced the huge mainframe computer centers.

Cloud service providers use massive high-performance banks of servers to satisfy high-volume, high-transaction-rate applications for a broad spectrum of clients.

Microprocessor Speed

Techniques built into contemporary processors are the following:

Pipelining: The execution of an instruction involves multiple stages of operation, including fetching the instruction, decoding the opcode, fetching operands, performing a calculation, and so on.

Pipelining enables a processor to work simultaneously on multiple instructions by performing a different phase for each of the multiple instructions at the same time.

The processor overlaps operations by moving data or instructions into a conceptual pipe with all stages of the pipe processing simultaneously.

For example, while one instruction is being executed, the computer is decoding the next instruction. This is the same principle as seen in an assembly line.

Branch prediction: The processor looks ahead in the instruction code fetched from memory and predicts which branches, or groups of instructions, are likely to be processed next.

If the processor guesses right most of the time, it can prefetch the correct instructions and buffer them so that the processor is kept busy.

Microprocessor Speed

Superscalar execution: This is the ability to issue more than one instruction in every processor clock cycle. In effect, multiple parallel pipelines are used.

Data flow analysis: The processor analyzes which instructions are dependent on each other's results or data, to create an optimized schedule of instructions.

Instructions are scheduled to be executed when ready, independent of the original program order. This prevents unnecessary delay.

Speculative execution: Using branch prediction and data flow analysis, some processors speculatively execute instructions ahead of their actual appearance in the program execution, holding the results in temporary locations.

This enables the processor to keep its execution engines as busy as possible by executing instructions that are likely to be needed.

Performance Balance

While processor power has raced ahead at breakneck speed, other critical components of the computer have not kept up.

The result is a need to look for performance balance: an adjustment/tuning of the organization and architecture to compensate for the mismatch among the capabilities of the various components.

The interface between processor and main memory is the most crucial pathway in the entire computer because it is responsible for carrying a constant flow of program instructions and data between memory chips and the processor.

If memory or the pathway fails to keep pace with the processor's insistent demands, the processor stalls in a wait state, and valuable processing time is lost.

Performance Balance

A system architect can attack this problem in a number of ways, all of which are reflected in contemporary computer designs. Consider the following examples:

- ✔ Increase the number of bits that are retrieved at one time by making DRAMs “wider” rather than “deeper” and by using wide bus data paths.
- ✔ Change the DRAM interface to make it more efficient by including a cache1 or other buffering scheme on the DRAM chip.
- ✔ Reduce the frequency of memory access by incorporating complex and efficient cache structures between the processor and main memory. This includes the incorporation of one or more caches on the processor chip as well as on an off-chip cache close to the processor chip.
- ✔ Increase the interconnect bandwidth between processors and memory by using higher-speed buses and a hierarchy of buses to buffer and structure data flow.

Performance Balance

Figure 2.1 gives some examples of typical peripheral devices in use on personal computers and workstations. These devices create tremendous data throughput demands.

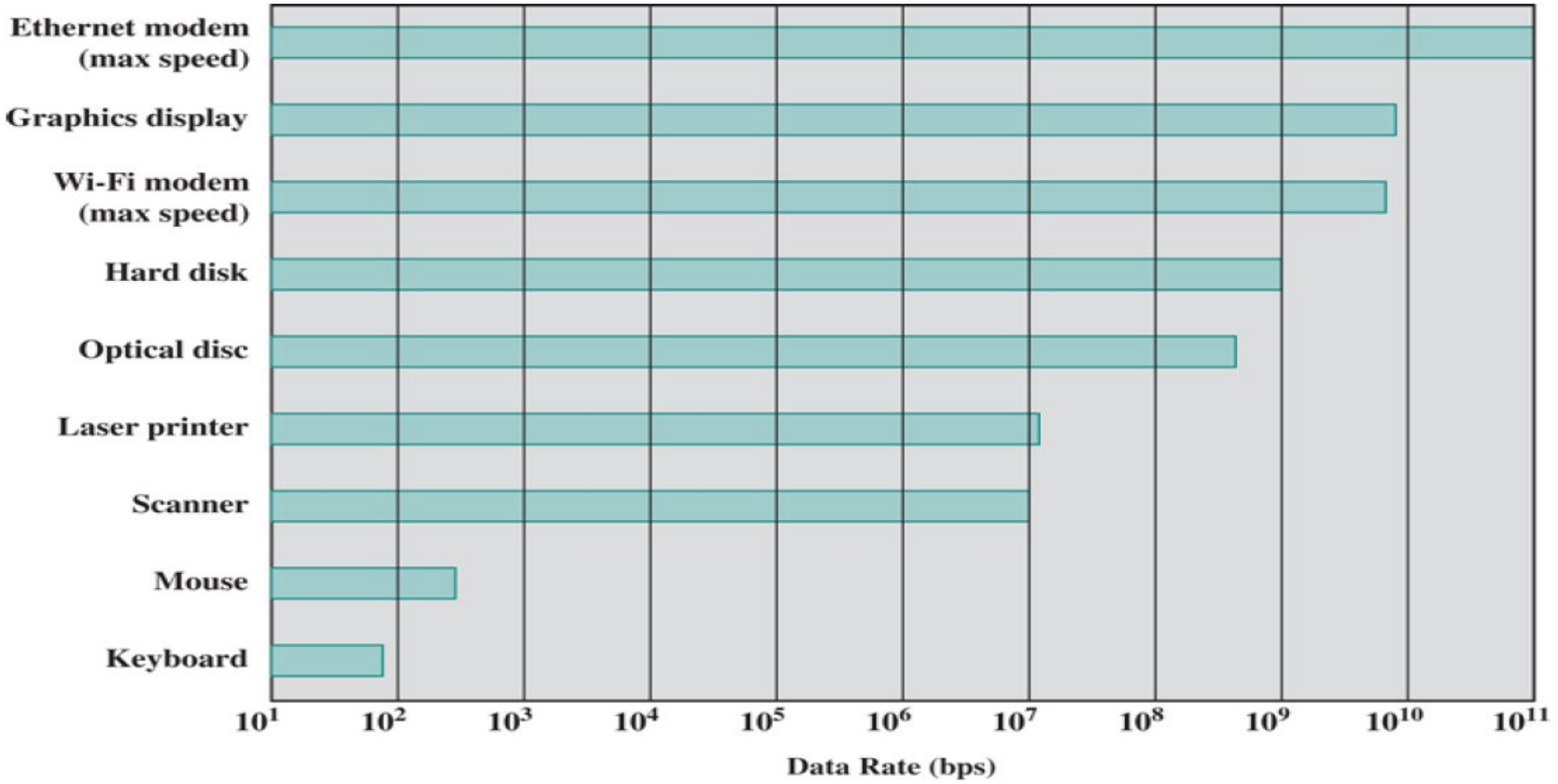

Figure 2.1 Typical I/O Device Data Rates

Performance Balance

Designers constantly strive to balance the **throughput** and **processing demands of the processor components, main memory, I/O devices, and the interconnection structures**.

This design must constantly be rethought to cope with two constantly evolving factors:

1. The rate at which performance is changing in the various technology areas (processor, buses, memory, peripherals) differs greatly from one type of element to another.

2. New applications and new peripheral devices constantly change the nature of the demand on the system in terms of typical instruction profile and the data access patterns.

Improvements in Chip Organization and Architecture

There are three approaches to achieving increased processor speed:

1. **Increase the hardware speed of the processor:** This increase is fundamentally due to shrinking the size of the logic gates on the processor chip so that more gates can be packed together more tightly and to increasing the clock rate. With gates closer together, the propagation time for signals is significantly reduced, enabling a speeding up of the processor. An increase in clock rate means that individual operations are executed more rapidly.
2. **Increase the size and speed of caches:** that are interposed between the processor and main memory. By dedicating a portion of the processor chip itself to the cache, cache access times drop significantly.
3. **Make changes to the processor organization and architecture:** that increase the effective speed of instruction execution. This involves using parallelism in one form or another.

Improvements in Chip Organization and Architecture

Dominant factor in performance gains has been increased in clock speed and logic density. As clock speed and logic density increase, a number of obstacles become more significant:

Power: As the density of logic and the clock speed on a chip increase, so does the power density . The difficulty of dissipating the heat generated on high-density, high-speed chips is becoming a serious design issue.

RC delay: The speed at which electrons can flow on a chip between transistors is limited by the resistance and capacitance of the metal wires connecting them.

Delay increases as the RC product increases. As components on the chip decrease in size, the wire interconnects become thinner, increasing resistance. The wires are closer together, increasing capacitance.

Memory latency and throughput: Memory access speed (latency) and transfer speed (throughput) slowdown processor speeds.There will be more emphasis on organization and architectural approaches to improve performance.

Improvements in Chip Organization and Architecture

Figure 2.2 illustrates: top line shows Moore's Law, the number of transistors on a single chip continues to grow exponentially. Meanwhile, the clock speed has leveled off, in order to prevent a further rise in power.

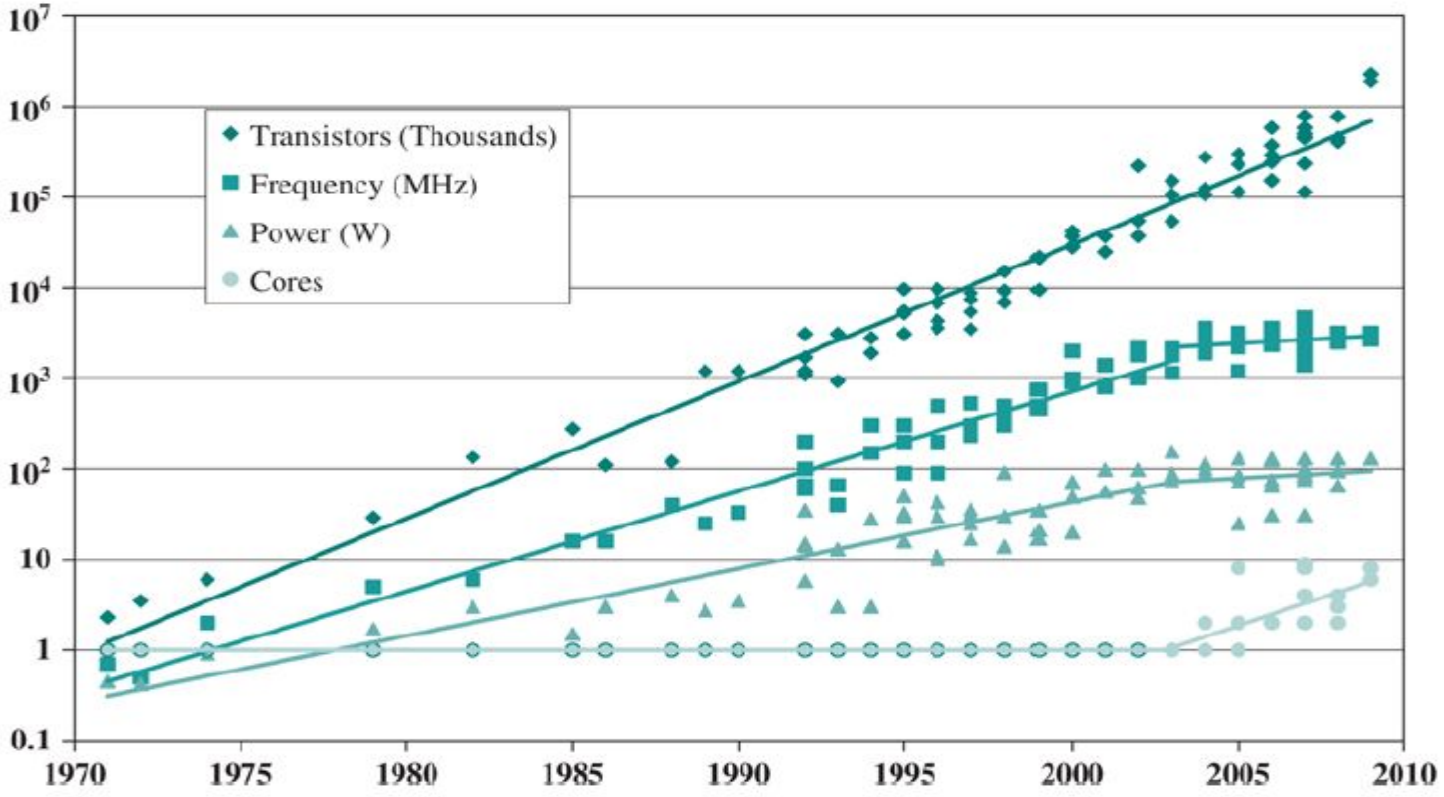

Figure 2.2 Processor Trends

Multicore, Mics, and GPGPUs

Use of multiple processors on the same chip, also referred to as multiple cores or multicore, provides the potential to increase performance without increasing the clock rate.

Within a processor, the increase in performance is roughly proportional to the square root of the increase in complexity.

If the software can support the effective use of multiple processors, then doubling the number of processors almost doubles performance.

Chip manufacturers are now in the process of increasing number of cores per chip, with more than 50 cores per chip. The performance as well as the challenges in developing software to exploit such a large number of cores has led to new term: **many integrated core (MIC).**

The multicore and MIC strategy involves a homogeneous collection of general-purpose processors on a single chip.