Trends in Technology

To plan for the evolution of a computer, the designer must be aware of rapid changes in implementation technology. Five implementation technologies, which change at a dramatic pace, are critical to modern implementations:

- *Integrated circuit logic technology*—Transistor density increases by about 35% per year, quadrupling somewhat over four years. Increases in die size are less predictable and slower, ranging from 10% to 20% per year. The combined effect is a growth rate in transistor count on a chip of about 40% to 55% per year, or doubling every 18 to 24 months. This trend is popularly known as Moore's law. Device speed scales more slowly, as we discuss below.
- *Semiconductor DRAM* (dynamic random-access memory)—Now that most DRAM chips are primarily shipped in DIMM modules, it is harder to track chip capacity, as DRAM manufacturers typically offer several capacity products at the same time to match DIMM capacity. Capacity per DRAM chip has increased by about 25% to 40% per year recently, doubling roughly every two to three years. This technology is the foundation of main memory, and we discuss it in Chapter 2. Note that the rate of improvement has continued to slow over the editions of this book, as Figure 1.8 shows. There is even concern as whether the growth rate will stop in the middle of this decade due to the increasing difficulty of efficiently manufacturing even smaller DRAM cells [Kim 2005]. Chapter 2 mentions several other technologies that may replace DRAM if it hits a capacity wall.
- *Semiconductor Flash* (electrically erasable programmable read-only memory)—This nonvolatile semiconductor memory is the standard storage device in PMDs, and its rapidly increasing popularity has fueled its rapid growth rate in capacity. Capacity per Flash chip has increased by about 50% to 60% per year recently, doubling roughly every two years. In 2011, Flash memory is 15 to 20 times cheaper per bit than DRAM. Chapter 2 describes Flash memory.
- *Magnetic disk technology*—Prior to 1990, density increased by about 30% per year, doubling in three years. It rose to 60% per year thereafter, and increased to 100% per year in 1996. Since 2004, it has dropped back to about 40% per year, or doubled every three years. Disks are 15 to 25 times cheaper per bit than Flash. Given the slowed growth rate of DRAM, disks are now 300 to 500 times cheaper per bit than DRAM. This technology is central to server and warehouse scale storage, and we discuss the trends in detail in Appendix D.
- *Network technology*—Network performance depends both on the performance of switches and on the performance of the transmission system. We discuss the trends in networking in Appendix F.

Energy and Power within a Microprocessor

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→1→0 or 1→0→1. The energy of a single transition (0→1 or 1→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 5V to just under 1V 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.

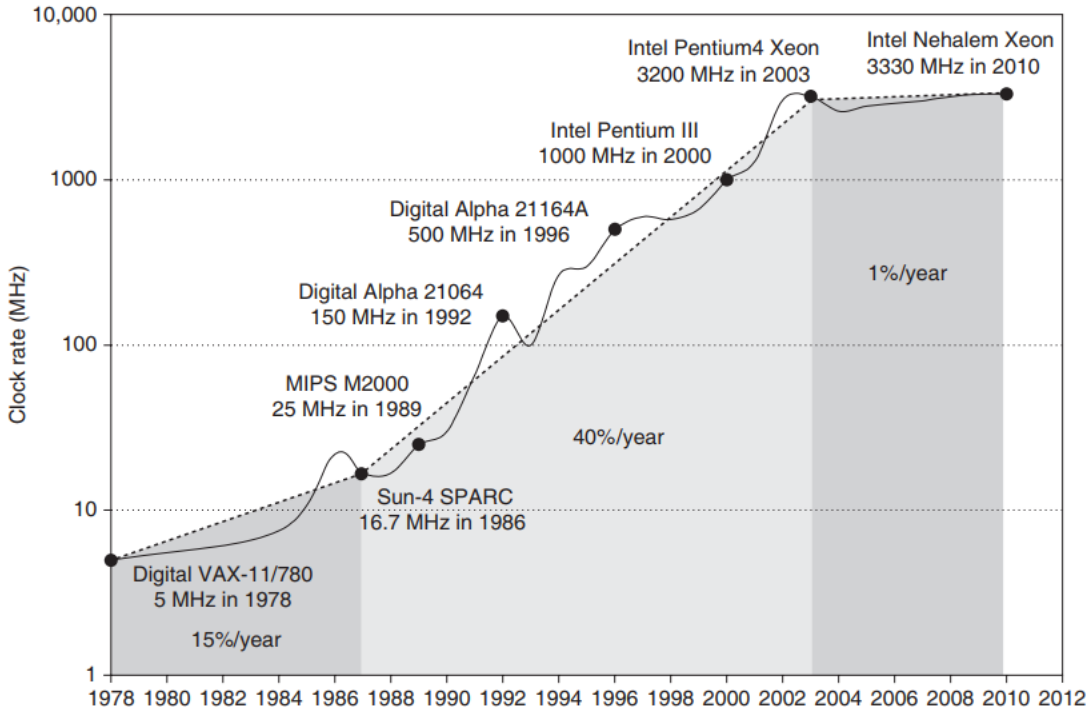

Figure 1.11 Growth in clock rate of microprocessors in Figure 1.1. Between 1978 and 1986, the clock rate improved less than 15% per year while performance improved by 25% per year. During the "renaissance period" of 52% performance improvement per year between 1986 and 2003, clock rates shot up almost 40% per year. Since then, the clock rate has been nearly flat, growing at less than 1% per year, while single processor performance improved at less than 22% per year.

Distributing the power, removing the heat, and preventing hot spots have become increasingly difficult challenges. Power is now the major constraint to using transistors; in the past, it was raw silicon area. Hence, modern microprocessors offer many techniques to try to improve energy efficiency despite flat clock rates and constant supply voltages:

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).* The second technique comes directly from the formulas above. Personal mobile devices, 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 GHz, 1.8 GHz, and 1 GHz. The overall server power savings is about 10% to 15% for each of the two steps.
3. *Design for 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 at lower rates when idle to save power. Alas, you cannot access DRAMs or disks in these modes, so you must return to fully active

   above, microprocessors for PCs have been designed instead for a more typical case of heavy use at high operating temperatures, relying on on-chip temperature sensors to detect when activity should be reduced automatically to avoid overheating. This "emergency slowdown" allows manufacturers to design for a more typical case and then rely on this safety mechanism if someone really does run programs that consume much more power than is typical.
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 example, the 3.3 GHz Core i7 can run in short bursts for 3.6 GHz. Indeed, the highest-performing microprocessors each year since 2008 in Figure 1.1 have all offered temporary overclocking of about 10% over the nominal clock rate. For single threaded code, these microprocessors can turn off all cores but one and run it at an even higher clock rate. Note that while 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 due to room temperature!

Trends in Cost

Although the costs of integrated circuits have dropped exponentially, the basic process of silicon manufacture is unchanged: A *wafer* is still tested and chopped into *dies* that are packaged (see Figures 1.13, 1.14, and 1.15). Thus, the cost of a packaged integrated circuit is

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

In this section, we focus on the cost of dies, summarizing the key issues in testing and packaging at the end.

Learning how to predict the number of good chips per wafer requires first learning how many dies fit on a wafer and then learning how to predict the percentage of those that will work. From there it is simple to predict cost:

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

The most interesting feature of this first term of the chip cost equation is its sensitivity to die size, shown below.

Dependability

- *Module reliability* is a measure of the continuous service accomplishment (or, equivalently, of the time to failure) from a reference initial instant. Hence, the *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*). Thus, an MTTF of 1,000,000 hours equals $10^9/10^6$ or 1000 FIT. Service interruption is measured as *mean time to repair* (MTTR). *Mean time between failures* (MTBF) is simply the sum of MTTF + MTTR. Although MTBF is widely used, MTTF is often the more appropriate term. If a collection of modules has exponentially distributed lifetimes—meaning that the age of a module is not important in probability of failure—the overall failure rate of the collection is the sum of the failure rates of the modules.
- *Module availability* is a measure of the service accomplishment with respect to the alternation between the two states of accomplishment and interruption. For nonredundant systems with repair, module availability is

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

Note that reliability and availability are now quantifiable metrics, rather than synonyms for dependability. From these definitions, we can estimate reliability of a system quantitatively if we make some assumptions about the reliability of components and that failures are independent.

1.3 PARALLEL COMPUTER STRUCTURES

Parallel computers are those systems that emphasize parallel processing. The basic architectural features of parallel computers are introduced below. We divide parallel computers into three architectural configurations:

- Pipeline computers
- Array processors
- Multiprocessor systems

A pipeline computer performs overlapped computations to exploit *temporal parallelism.* An array processor uses multiple synchronized arithmetic logic units to achieve *spatial parallelism.* A multiprocessor system achieves *asynchronous parallelism* through a set of interactive processors with shared resources (memories, database, etc.). These three parallel approaches to computer system design are not mutually exclusive. In fact, most existing computers are now pipelined, and some of them assume also an "array" or a "multiprocessor" structure. The fundamental difference between an array processor and a multiprocessor system is that the processing elements in an array processor operate synchronously but processors in a multiprocessor system may operate asynchronously.

New computing concepts to be introduced in this section include the *data flow computers* and some *VLSI algorithmic processors.* All these new approaches demand extensive hardware to achieve parallelism. The rapid progress in the VLSI technology has made these new approaches possible.

Table 1-1 Five Generations of Electronic Computers

Generation	Technology and Architecture	Software and Applications	Representative Systems
First (1945-54)	Vacuum tubes and relay memories, CPU driven by PC and accumulator, fixed-point arithmetic.	Machine/assembly languages, single user, no subroutine linkage, programmed I/O using CPU.	ENIAC, Princeton IAS, IBM 701.
Second (1955–64)	Discrete transistors and core memories, floating-point arithmetic, I/O processors, multiplexed memory access.	HLL used with compilers, subroutine libraries, batch processing monitor.	IBM 7090, CDC 1604, Univac LARC.
Third (1965-74)	Integrated circuits (SSI/-MSI), microprogramming, pipelining, cache, and lookahead processors.	Multiprogramming and time-sharing OS, multiuser applications.	IBM 360/370, CDC 6600, TI-ASC, PDP-8.
Fourth (1975-90)	LSI/VLSI and semiconductor memory, multiprocessors, vector supercomputers, multicomputers.	Multiprocessor OS, languages, compilers, and environments for parallel processing.	VAX 9000, Cray X-MP, IBM 3090, BBN TC2000.
Fifth (1991-present)	ULSI/VHSIC processors, memory, and switches, high-density packaging, scalable architectures.	Massively parallel processing, grand challenge applications, heterogeneous processing.	Fujitsu VPP500, Cray/MPP, TMC/CM-5, Intel Paragon.

1.4 ARCHITECTURAL CLASSIFICATION SCHEMES

Three computer architectural classification schemes are presented in this section. *Flynn's classification* (1966) is based on the multiplicity of instruction streams and data streams in a computer system. *Feng's scheme* (1972) is based on serial versus parallel processing. *Händler's classification* (1977) is determined by the degree of parallelism and pipelining in various subsystem levels.

1.4.1 Multiplicity of Instruction-Data Streams

In general, digital computers may be classified into four categories, according to the multiplicity of instruction and data streams. This scheme for classifying computer organizations was introduced by Michael J. Flynn. The essential computing process is the execution of a sequence of instructions on a set of data. The term *stream* is used here to denote a sequence of items (instructions or data) as executed or operated upon by a single processor. *Instructions* or *data* are defined with respect to a referenced machine. An *instruction stream* is a sequence of instructions as executed by the machine; a *data stream* is a sequence of data including input, partial, or temporary results, called for by the instruction stream.

Computer organizations are characterized by the multiplicity of the hardware provided to service the instruction and data streams. Listed below are Flynn's four machine organizations:

- Single instruction stream-single data stream (SISD)
- Single instruction stream-multiple data stream (SIMD)
- Multiple instruction stream-single data stream (MISD)
- Multiple instruction stream-multiple data stream (MIMD)

These organizational classes are illustrated by the block diagrams in Figure 1.16. The categorization depends on the multiplicity of simultaneous events in the system components. Conceptually, only three types of system components are needed in the illustration. Both instructions and data are fetched from the *memory modules*. Instructions are decoded by the *control unit*, which sends the decoded instruction stream to the *processor units* for execution. Data streams flow between the processors and the memory bidirectionally. Multiple memory modules may be used in the shared memory subsystem. Each instruction stream is generated by an independent control unit. Multiple data streams originate from the subsystem of shared memory modules. I/O facilities are not shown in these simplified block diagrams.

SISD computer organization This organization, shown in Figure 1.16*a*, represents most serial computers available today. Instructions are executed sequentially but may be overlapped in their execution stages (pipelining). Most SISD uniprocessor systems are pipelined. An SISD computer may have more than one functional unit in it. All the functional units are under the supervision of one control unit.

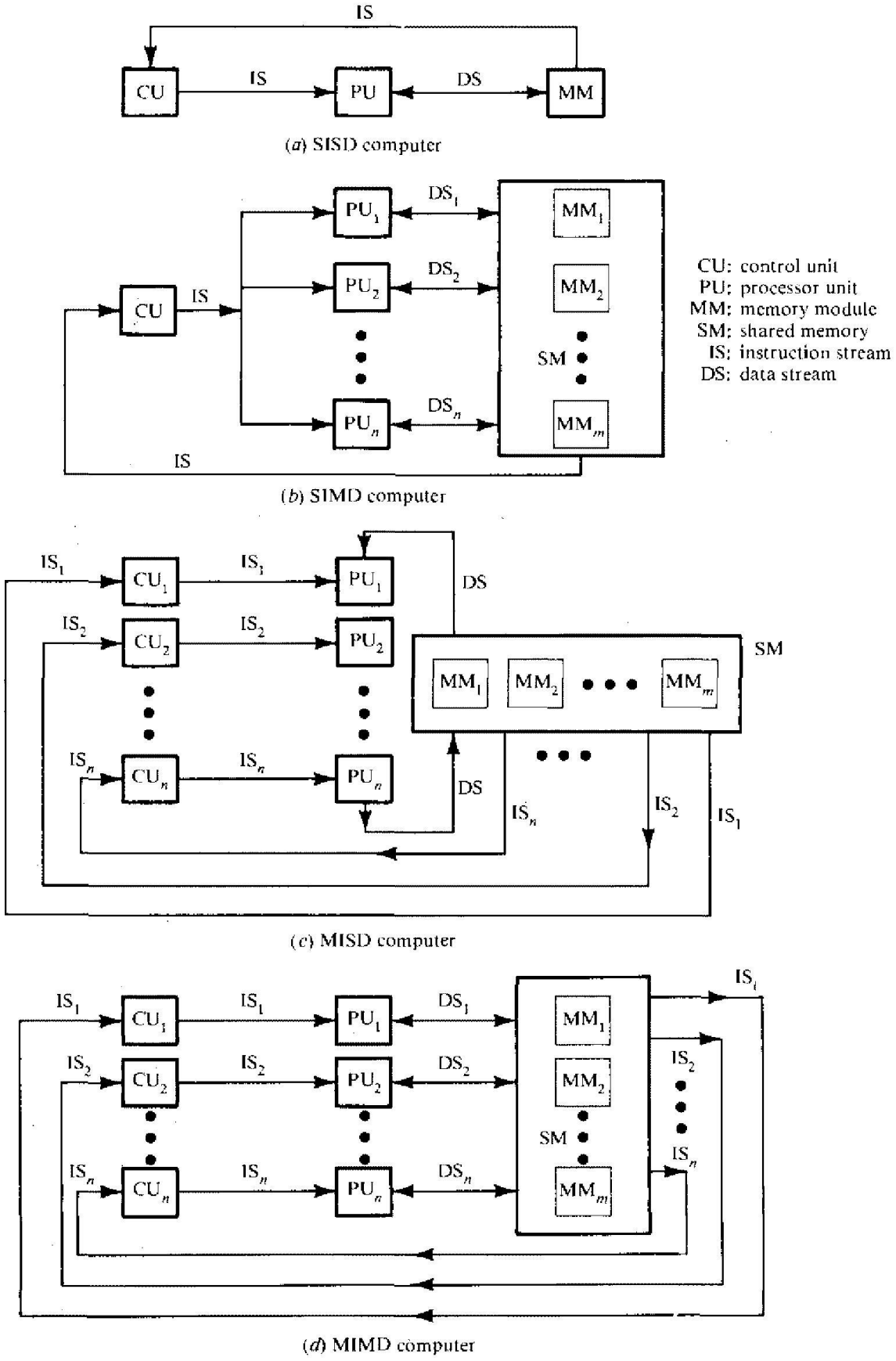

Figure 1.16 Flynn's classification of various computer organizations.

SIMD computer organization This class corresponds to array processors, introduced in Section 1.3.2. As illustrated in Figure 1.16*b*, there are multiple processing elements supervised by the same control unit. All PEs receive the same instruction broadcast from the control unit but operate on different data sets from distinct data streams. The shared memory subsystem may contain multiple modules. We further divide SIMD machines into *word-slice* versus *bit-slice* modes, to be described in Section 1.4.2.

MISD computer organization This organization is conceptually illustrated in Figure 1.16*c*. There are n processor units, each receiving distinct instructions operating over the same data stream and its derivatives. The results (output) of one processor become the input (operands) of the next processor in the macropipe. This structure has received much less attention and has been challenged as impractical by some computer architects. No real embodiment of this class exists.

MIMD computer organization Most multiprocessor systems and multiple computer systems can be classified in this category (Figure 1.16*d*). An *intrinsic* MIMD computer implies interactions among the n processors because all memory streams are derived from the same data space shared by all processors. If the n data streams were derived from disjointed subspaces of the shared memories, then we would have the so-called multiple SISD (MSISD) operation, which is nothing but a set of n independent SISD uniprocessor systems. An intrinsic MIMD

Table 1.3 Flynn's computer system classification

Computer class	Computer system models (chapters where the system is quoted or described)
SISD (uses one functional unit)	IBM 701 (1); IBM 1620 (1); IBM 7090 (1); PDP VAX11/780 (1).
SISD (with multiple functional units)	IBM 360/91 (3); IBM 370/168UP (1); CDC 6600 (1); CDC Star-100 (4); TI-ASC (4); FPS AP-120B (4); FPS-164 (4); IBM 3838 (4); Cray-1 (4); CDC Cyber-205 (4); Fujitsu VP-200 (4); CDC-NASF (4); Fujitsu FACOM-230/75 (4).
SIMD (word-slice processing)	Illiac-IV (6); PEPE (1); BSP (6)
SIMD (bit-slice processing)	STARAN (1); MPP (6); DAP (1).
MIMD (loosely coupled)	IBM 370/168 MP (9); Univac 1100/80 (9); Tandem/16 (9); IBM 3081/3084 (9); C.m* (9)
MIMD (tightly coupled)	Burroughs D-825 (9); C.mmp (9); Cray-2 (9). S-1 (9); Cray-X MP (9); Denelcor HEP (9)