1

The Memory Function

Memory is the second function of the computer, between computation and communication. Since the invention of the modern computer in the last century, memory has received the full attention of researchers, and later, of industry. It takes physical form as a component or a system, such as a memory subsystem or a mass storage device.

This chapter begins with a presentation of the characteristics of a generic memory. The most essential is likely its size, and the standardized multiples of its unit of measure are detailed in this context. The technological and historical aspects provide an opportunity to present the different memory families. These different memory families form a hierarchy in the computer, a concept that is then defined.

1.1. Main characteristics

Memory is mainly characterized by its medium, its total storage[1] capacity C and its organization, the memory access method or policy, the type of access or operation, and the permanence (or non-volatility) of the information.

1.1.1. *Memory medium*

The type of medium characterizes the technology used to store the information. Information storage can be done through the presence or absence of a material (mechanical technology), such as a hole in a punched card (see section 5.6 of Darche

1 In English, the term “storage” is generally reserved for a direct or sequential access device, such as an HDD or magnetic tape storage, while the term “memory” is used for random-access components like random access memory (RAM) or read-only memory (ROM). This distinction also applies in French, with the terms “*stockage*” and “*mémoire*”.

(2003) and section 3.3 of Darche (2000)). Today, it involves a semiconductor (SC) or a magnetic, optical or magneto-optical material. In electronics, storage is done, for example, by moving or storing electric charges. In mass storage memory, it uses the change in orientation of a ferromagnetic material's magnetic field (originally iron oxide) on a flexible (floppy disk[2] or FD) or rigid (hard disk or HD) support, or, as in rewritable optical memory, a phase change.

French term	English term	Format n (bits)
Chiffre binaire	Bit (b)	1
Quartet ou demi-octet	Nibble	4
Octet	Byte[a] (B)	8
Mot	Word Dualoct	16
Double mot	Double word Quadlet quadoct	32
Quadruple mot	Quad(ruple) word Octlet (IEEE 1996) Octbyte	64

a. A contraction of the English word "bite", meaning "a mouthful", created to follow the word "bit".

Table 1.1. *Vocabulary used to describe a group of bits*

1.1.2. *Storage capacity and units of measurement*

The binary elements, 0 and 1, were named "bits" by statistician John Wilder Tukey, as a contraction of binary digit, although other terms such as bigit and binit were also suggested. The term was adopted by the community following its appearance in his famous article on information theory (Shannon 1948). The unit of measurement for memory capacity C is therefore the bit. Table 1.1 reminds us of the vocabulary used to describe a set of bits. In the era of 16-bit microprocessors (μP, microprocessor unit or MPU) in the 1980s, a word represented 16 bits. With the advent of 32-bit microprocessors, the term double word was introduced. Then came the next generation with the term quad word. But be careful – the word "word" can sometimes refer to any format depending on context, and not necessarily a multiple of the byte. We also speak of memory density, which refers to the number of bits stored per unit area (in^2 or mm^2).

2 The corresponding unit is called the floppy disk drive or FDD (FD drive).

IEC symbols and prefixes	Origin	Factors	Examples	SI symbols and prefixes (reminder)
Ki, kibi	Kilobinary	2^{10} (= 1 024)	1 Kib (formerly 1 KB)	k (kilo) = 10^3
Mi, mebi	Megabinary	2^{20} (= 1 048 576)	1 Mib (formerly 1 Mb)	M (mega) = 10^6
Gi, gibi	Gigabinary	2^{30}	1 Gib (formerly 1 Gb)	G (giga) = 10^9
Ti, tebi	Terabinary	2^{40}	1 Tib (formerly 1 Tb)	T (tera) = 10^{12}
Pi, pebi	Petabinary	2^{50}	1 Pib (formerly 1 Pb)	P (peta) = 10^{15}
Ei, exbi	Exabinary	2^{60}	–	E (exa) = 10^{18}
Zi, zebi	Zettabinary	2^{70}	–	Z (zetta) = 10^{21}
Yi, yobi	Yottabinary	2^{80}	–	Y (yotta) = 10^{24}

Table 1.2. *New standardized prefixes for capacity units*

Table 1.2 shows the symbol and prefix (or name) for the multiples of the bit. To distinguish the SI prefix kilo (10^3) from the kilo used in memory sizing, the uppercase letter K is commonly used in computing. For example, we have 1 Kb (kilobit) meaning 1,024 bits and 1 KB (kilobyte) meaning 1,024 bytes. A technical term used is computer kilo. To eliminate ambiguity, the International Electrotechnical Commission (IEC), an international standards organization, approved a new naming convention for powers of 2 with the kilobinary symbol Ki, named kibi (IEC 2000). Self (1999) explains this aspect in detail. The Institute of Electrical and Electronics Engineers (IEEE), a US-based technical professional standards body, later standardized it (IEEE 2002a, 2002b).

Figure 1.1 shows a graphic reminder of these new measurement units for semiconductor memory and the relationships between them.

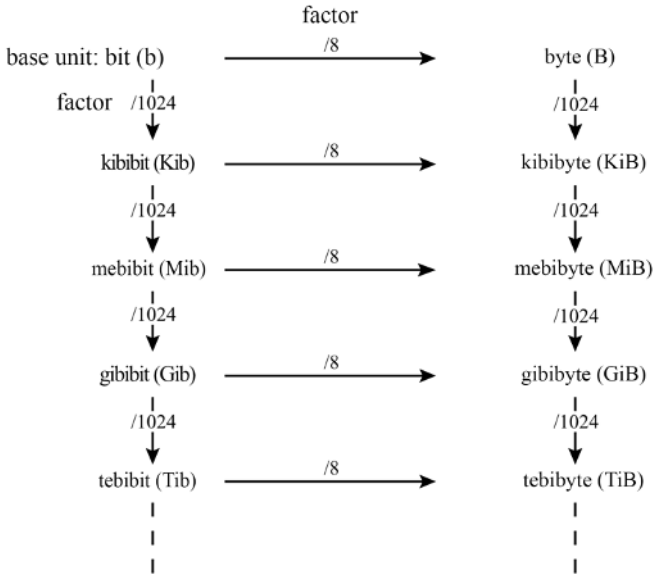


Figure 1.1. *Base unit and standardized prefixes for a semiconductor memory*

Currently, not all professionals use these two new standards, which causes some confusion. Manufacturers of semiconductor memory continue to use the old prefixes. This confusion persists with the sizes of hard disk mass memory units, where the ratio or factor between this unit and its multiples is equal to $10^{3 \times k}$ ($k \in \mathbb{N}^*$). For examples and details, see section 7.2.2 and the associated exercise 7.3 of Darche (2003).

Figure 1.2 recalls the unit of measurement for mass memory capacity and the ratios between it and its multiples, specifying its first multiples. The shaded area defines a nonexistent capacity, since the first industrial disk, the IBM 350 disk storage unit (1956) by IBM, had a capacity of 3.75 MB (5×10^6 characters or 6-bit words). Its presence is justified only to recall the kilo ratio of 10^3.

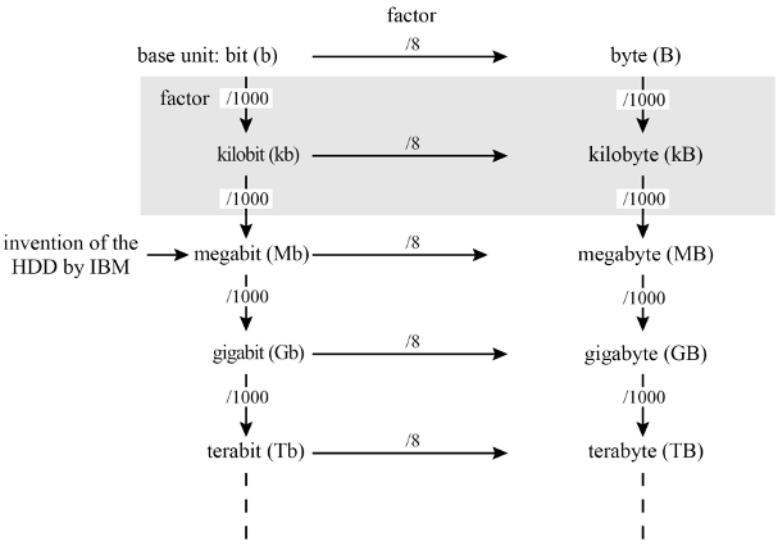


Figure 1.2. *Unit of measurement of mass memory capacity and its first multiples*

And to be complete, operating systems (OS) convert mass storage capacity into a power of 2. The Linux OS even uses the new standardized prefixes. Universal serial bus (USB) keys, which use semiconductor memories, still follow powers of 2!

1.1.3. *Organization*

The memory cell is the smallest subdivision (atomic entity) of memory in which it is possible to read or write information. A cell or memory word has a format or width[3] l or n or a size. We speak, for example, of the width of a register (concept detailed in section 3.1 of Darche (2021c)). This format mainly depends on the memory system format and the data bus width or mechanisms such as interleaving (see section 5.2.4). It can be a byte or a word multiple of the byte or another format (n = 9 bits, for example, in the case of error detection and correction, see section 6.4). Each bit bi of the cell of format n has a position i numbered, from right to left, from 0 to n–1 ($i \in [0, n–1]$; see Figure 1.3).

3 Some authors have used the term “length” for the format (Meinadier 1988 or Ciminiera and Valenzano 1987). This term was not retained in this book because it is reserved here for the number of words.

The organization refers to the physical arrangement of the cells in the memory. It allows us, from the total size, to specify the distribution between the format n of the memory cell and the number L of cells or depth D. We speak of input/output (I/O) organization length L × width n, for example, 16 Ki × 16 bits. In the case where there are several internal banks (see section 4.3), we speak of bank organization B × length L × width n, where B is the number of banks, for example, 8 × 2 Ki × 16 bits. The total capacity C of the memory is then equal to:

$$C = B \times L \times n \qquad [1.1]$$

1.1.4. *Access policies*

The memory cell is called by Cragon the "memory-addressable unit" (AU) (Cragon 1996). It is the atomic element (i.e. the smallest addressable word) that the MPU can address in the main memory (or primary memory, acronym MP), i.e. a byte or a word of n bits.

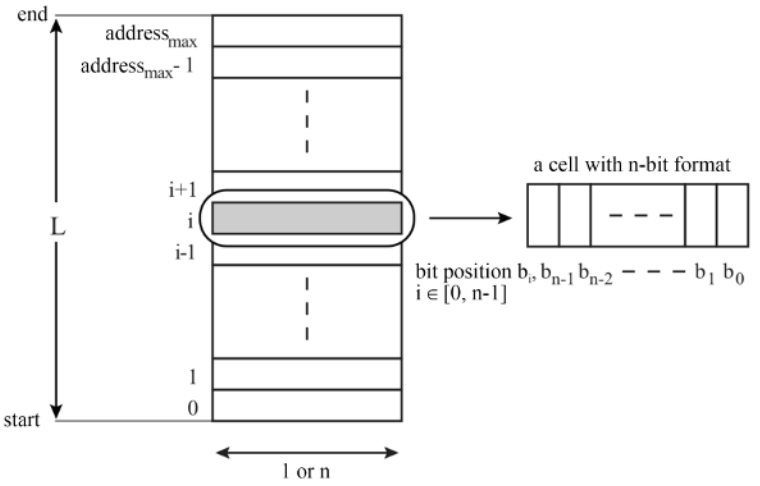


Figure 1.3. *Organization and addressing of a memory*

The memory access method or policy specifies how the memory is accessed. In the case of location-based or coordinate addressing, the memory cell is accessed using an address (address-based storage and retrieval) which is an integer. This integer generally belongs to the subset of natural numbers, a counterexample being the address used to move to increasing or decreasing addresses relative to a

reference address (see section 1.2.3.2 of Darche (2021d)). If we number each memory cell starting from 0[4], then the cell's address is this number (see Figure 1.3). The interest of this figure is to facilitate the transition from this abstract model, which I call "the dresser"[5] by analogy, to the logical implementation of Figure 2.5.

If the cell is the smallest addressable element, then the format m of address A numbered from 0 is equal to:

$$m = \lfloor \log_2 address_{max} \rfloor + 1 \; bits \quad [1.2]$$

where the mathematical symbols "⌊"and"⌋" represent the "smallest integer less than or equal to" the function (floor() function) and with the $address_{max} \neq 0$.

And we have:

$$L = 2^{m} \quad [1.3]$$

The access policy (management strategy or policy) specifies how to access a memory cell. Figure 1.4 classifies them. In address-based access, random access[6] means that any memory cell can be accessed directly by providing its address with a constant access time t_a (see section 1.1.7). In the case of serial or sequential access, the memory cells are accessed in a specific order that characterizes the policy. An example of a policy is "first data in is first data out" (FIFO) for a queue, or "last data in is first data out" (LIFO) for a stack. It should not be confused with serial location, an example of which is magnetic tape, which is not a semiconductor memory. Its records are read one after another by reading their header to determine which one is sought. In content-based access (by content), the information is accessed by partial match, as we would do with a phone directory where the contact name is used to retrieve the number (Belady et al. 1981). The data must therefore contain a key or an identifier. This type of policy is well suited for information retrieval. Thus, content-addressable memory (CAM), which is associative memory, is particularly used in network devices and in the cache (see Volumes 7 and 8, respectively). Associative addressing is a generalization of content-based addressing because it does not require an exact match (Chisvin and Duckworth 1989).

4 Justification of this initial value in section 2.2.6.

5 By analogy with this piece of furniture where each numbered drawer is a memory cell.

6 This adjective should especially not be confused with randomness or a probability law. Direct access could be a synonym. However, it is actually reserved to define one of the modes of addressing a memory location by a processor (microprocessor or microcontroller). For a mass memory like an HDD, we also speak of direct addressing, but it must be known that the access time then depends on the location of the information and the position of the read head, unlike a semiconductor memory where this time is constant.

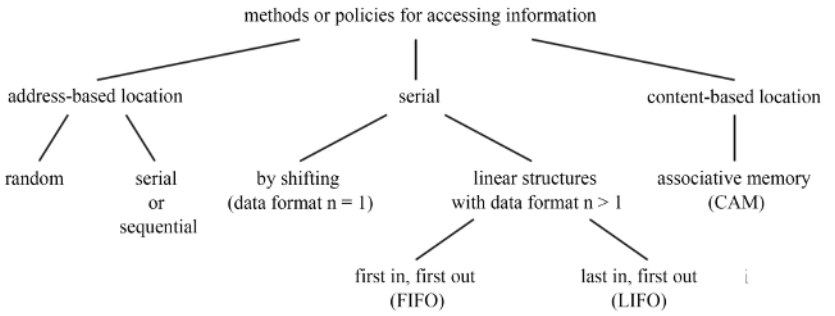


Figure 1.4. *Memory access policies*

1.1.5. *Types of access*

The type of access specifies the requested operation, which can be read, write, read–write or read–modify–write. Read–write means the stored data can be read and then another data stored at the same address during the same cycle. The last operation is a special case where the data read is modified and then stored at the same address, still in the same cycle. This last type of access is useful, for example, for error detection and correction (ED(A)C, error detection (and) correction, ECC error-correcting code; see section 6.4). The type of access may have given the memory its name, such as ROM (read-only memory), an electronic component, or WORM or WOM (write-once read-many-times memory), a mass memory device.

1.1.6. *Data permanence*

Data permanence indicates whether information is lost when power is off, or if it remains stored. This permanence is characterized by the data-retention time. A memory that loses its stored data when power is lost or due to the storage material's properties is called "volatile". It is also called RAM.

Considering data retention or usage duration, memory may be termed immediate-, very short-, short-, mid- or long-term memory. For example, a register is a very short-term memory, while a magnetic tape is a very long-term memory.

1.1.7. *Timing characteristics*

To characterize a memory in time (see section V2-1.2 for the semiconductor version), the read access and cycle times (access time and cycle time, respectively)

are specified. Access time t_a is the time from presenting the address to presenting the output data. Note, it is defined only for the read operation. Cycle time t_c is the total time for a read or write operation. This cycle time can vary, depending on the operation, into a write cycle time t_{WC} and a read cycle time t_{RC}. Note that t_a is less than or equal to t_c (actually t_{RC}). An important note: in random access, both times are independent of the memory cell's position.

At the module or component level, other times are used to refine timing characterization. They depend on memory type and the organization of the subset. So, they are detailed in relevant chapters and volumes.

For successive accesses, also called block or burst mode (see Volumes 3 and 4), throughput must be considered. Throughput is divided into I/O rate and data rate (Chen and Patterson 1993). The first, measured in accesses per second, applies when few bytes are transferred per access, such as in transaction processing. This definition applies to storage systems such as hard disk drives.

In this book, we distinguish two types of throughput: raw throughput or data rate, and effective throughput. Raw throughput is the maximum rate a component can provide at the hardware level. Effective throughput is the average rate available to the requester, usually a memory controller or processor. Data rate is measured in bits (or bit multiples, typically bytes) transferred per unit of time. It depends on the data format n. So, the basic unit is bits per second (bit/s). Throughput multiples follow powers of 10. Thus, 1 KB/s or kbps (note the lowercase k from SI) equals 1,000 bits/s. For semiconductor memory, raw throughput can be expressed in b/s/pin[7] (or bpspp), i.e. n = 1 bit. If the data transfer format is larger, total throughput equals that value multiplied by n. Throughput is the inverse of t_c (see Exercise V2-E1.1). Access is usually by blocks[8] to maximize throughput. This is called burst mode access. Table 1.3 lists some typical I/O throughput values (see Exercise E1.1 for the first three values).

A more relevant parameter is the fill frequency (FF). It describes the global performance of the memory system. It is defined by the equation:

$$\text{fill frequency (Hz)} = \frac{throughput\ (b/s)}{memory\ size\ (b)} \qquad [1.4]$$

7 From the component's package.
8 As opposed to random access.

This fill frequency applies to a full memory system and to the component itself, helping to quantify market demand (market requirement FF) (Przybylski 1996). Typical values are: 10 Hz for a workstation, 30 Hz for a PC (personal computer) and 100 Hz for a GUI (graphical user interface) accelerator. This indicator indicates here that a graphics accelerator needs high bandwidth, or that a workstation's memory system bandwidth is well used. Also, if a memory component's FF is below FF market requirements, it is not suitable (see Figure 2 of Przybylski (1996)).

Application or interface	Bandwidth	Raw throughput
High-fidelity (Hi-Fi) mono sound	20 kHz	88.2 KB/s (CD (compact disc))
Fast Ethernet (100Base-T) – IEEE Std 802.3™	–	100 Mb/s
Luminance video signal	Approximately 6 MHz	135 MB/s
Uncompressed digital video image (n = 24 bits) XGA (extended graphics array)	–	118 MB/s
Gigabit Ethernet (1000Base-T) – IEEE Std 802.3™	–	1 Gb/s
Disk interface PATA (parallel advanced technology attachment)	–	133 MB/s
Disk interface SATA (serial advanced technology attachment)	–	150 MB/s (SATA 1.0a – 2003) 300 MB/s (SATA 2.0 – 2004) 600 MB/s (SATA 3.0 – 2009)
DVI (digital visual interface) single link	165 MHz	3.96 Gb/s
PCI (peripheral component interconnect) Express bus	–	4 Gb/s (× 1, unencoded) 64 Gb/s (× 16, unencoded)

Application or interface	Bandwidth	Raw throughput
USB	–	1.5 et 12 Mb/s (USB 1.0 – 1996) 480 Mb/s (USB 2.0 – 2000) 5 Gb/s (USB 3.0 – 2008) 10 Gb/s (USB 3.1 – 2013) 20 Gb/s (USB 3.2 – 2017) 40 Gb/s (USB 4.0 – 2019) 40 Gb/s (USB 4.2 – 2019)
HDMI (high-definition multimedia interface)	165 MHz 340 MHz 600 MHZ 1,200 MHZ	4.95 Gb/s (1.0 – 2002) 10.2. Gb/s (1.3 – 2006) 18 Gb/s (2.0 – 2013) 48 Gb/s (2.0 – 2013)

Table 1.3. *Bandwidth of various applications and interfaces*

It is also useful to have an external characterization of the memory subsystem at the level of a bus master such as a processor. Katayama (1997) proposes to evaluate system performance using equations [1.5] and [1.7], comparable to Gene Amdahl's empirical rules. Coefficients k_1 and k_2 adjust the performance. B is the number of banks and N is the total number of memory components. We can deduce that the term in [1.9] is actually a fill frequency when comparing it to relation [1.4]:

$$\text{MPU performance} = k_1 \times \text{memory system bandwidth} \quad [1.5]$$

$$= k_1 \times \frac{N}{K} \times \text{throughput of one memory} \quad [1.6]$$

and

$$\text{MPU performance} = k_2 \times \text{memory system capacity} \quad [1.7]$$

$$= k_2 \times N \times \text{size of one memory} \quad [1.8]$$

From [1.6] and [1.8], we can deduce:

$$\text{throughput of one memory} = B \times \text{size of one memory} \times \frac{k_1}{k_2} \qquad [1.9]$$

Another important parameter is latency[9], which is the total time between presenting the address and the availability of the data (read) or its acknowledgment (write). It includes all delays in the data path (DP, see section 3.2.2.1 of Darche (2021a)). This is an upper bound. In block access, latency concerns access to the first element. So, as before, throughput must also be considered. It depends on memory-related times (read access and cycle time), subsystem-related times (organization, controller) and exchange times between the memory subsystem, bus and processor. For the bus, this includes its access protocol, especially arbitration, its clock frequency and its width. For the processor, this concerns its protocol and cycle time. Latency and throughput(s) are the two most used parameters to evaluate the performance of a memory subsystem.

More generally, in computing, latency is the time elapsed between initiating an operation and the moment its result appears. This characteristic also appears in the networking domain.

1.2. Modeling

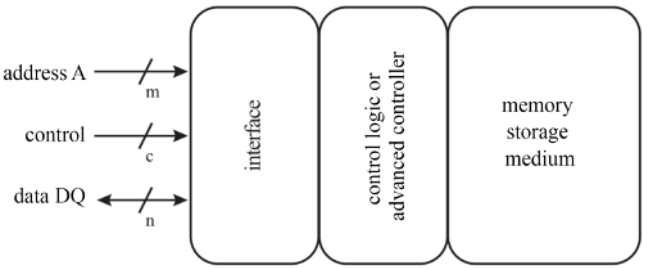


Figure 1.5. *Memory modeling*

A memory, as a component or subsystem, can be modeled with three subunits: the storage medium, the controller and the interface (see Figure 1.5; see also Figure 9.1). The latter two can perform complex operations. The storage medium

9 For an HDD memory unit, rotational latency or rotational delay is the time it takes for the head to find the beginning of a given sector once the arm is positioned on the addressed track (see section 7.2 of Darche (2003)).

may be removable, such as in a CD-ROM, or fixed. At the interface level, address information A in *m*-bit format can be exchanged bit by bit (serial interface) or by *n*-bit ($n > 1$) in a single exchange (parallel interface). The name DQ refers to the signals of a D-type flip-flop (see section 2.1). This exchange can be synchronous, meaning an external clock is needed to pace the operation (see clock notion in sections 3.1.2 and 3.1.4 of Darche (2002)). Otherwise, it is asynchronous, meaning it occurs within a bounded time frame. The control signal group includes at least one signal specifying the access type (R/#W or #WE), and generally, a memory select signal (#CE for chip enable or #CS for chip select). The latter controls the component's sleep mode. In a synchronous approach, a clock signal Clk is present. The controller generates internal control signals from the external control signals.

A computer system can be graphically described using symbolic elements, as shown in Figure 1.6 with a parallel memory architecture of the NUMA (non-uniform memory architecture) type. The compute element, such as a microprocessor or core, can be represented by a compute element or PE/PU (processing element/unit), a memory by a memory element or ME/MU (memory element/unit) and a peripheral (input-output) with its controller by an I/O element or IOE/IOU (I/O element/unit).

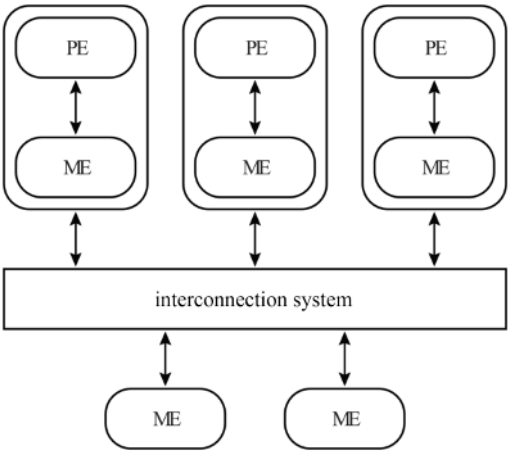


Figure 1.6. *Shared and distributed NUMA memory architecture*

1.3. Technological and historical aspects

It is impossible to discuss computing without addressing the technologies used or still used to design computer elements. This section details them, serving as a

pretext to begin examining the various memory components or systems. It then becomes possible to finally define the concept of memory hierarchy.

1.3.1. *Mechanical calculator technologies*

The pursuit of researchers in computer architecture and manufacturers has been to invent memory technologies that are reliable, fast, and low-cost or economically viable. Any physical phenomenon or system with n stable states (n = 2 in the case of binary logic, the most common) is a candidate for use in a memory system. Memory can also be analog.

To find memory in a machine, we must go back about 300 years before Christ with the Chinese abacus, or to the 17th century with the mechanical accumulators in the mechanical calculators of Blaise Pascal (idea dating from 1642) and Gottfried Wilhelm Leibniz (1646–1716). The 18th and 19th centuries saw the emergence of programmable mechanical machines. These were the looms of Jacques de Vaucanson (in 1728, see Figure 1.8 of Darche (2021)) and Joseph-Marie Jacquard (in 1804), whose weaving programs were stored as punched cards. This latter format had been invented by Basile Bouchon in 1725 as a punched paper tape for the weaving industry. Jean-Baptiste Falcon improved it by transforming the tape into a series of punched cards linked by cords. These cards stored the weaving pattern. More details and illustrations are given in Marguin (1994).

Another example in this technology is the second difference engine of Charles Babbage (see Figure 1.6 of Darche (2021a)). It introduced the basic architecture of a computer and its programming (Hartree 1948). It included a mill playing the role of the modern CPU and a store serving as main memory (see Figure 1.7 of Darche (2021a)). There were also the concepts of registers (major axes) and data buses (transfer paths). To program it, Babbage also proposed the punched card. Detailed explanations of its operation are provided in Bromley (1982). This memory principle was reused with Herman Hollerith's punched card, which stored the data and instructions of tabulating machines (1890) (IEEE Spectrum 00; Kistermann 1991, 2005).

1.3.2. *Main memory technologies in modern computers*

The term "modern computer" often refers to electronic technology, but generation 1 (Darche 2021a) were electromechanical calculators (1937–1945). The basic component is the electromechanical relay or EMR (see Figure 1.7).

The vacuum tube (or thermionic tube, Figure 1.8) was then the first electronic component used. Tube-based memories do not all use the same principle. The flip-flop, also called bistable (English or French term), is the basic element for data storage and synchronization, and by extension, for the operation of automata, including microprocessor cores (see Darche (2002)).

The first flip-flop implementation was the circuit by Eccles and Jordan (1919), made of two triode vacuum tubes, also called electronic tubes, and six resistors (see an example with the electronic numerical integrator and computer (ENIAC) in Burks (1947)).

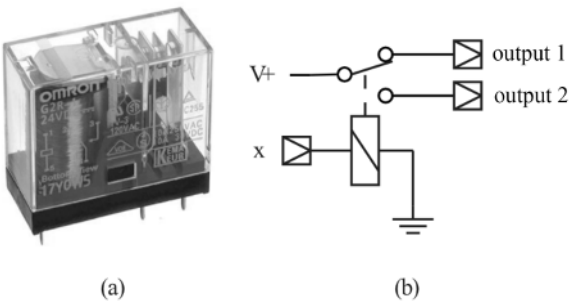


Figure 1.7. *A modern electromechanical relay and its equivalent circuit diagram (source: left part of Figure 1.11 from Darche (2020a))*

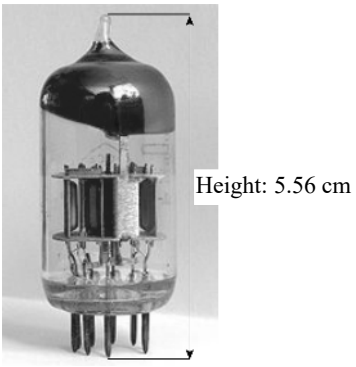


Figure 1.8. *An RCA 5965 type vacuum tube (source: left part of Figure 1.12 from Darche (2020a))*

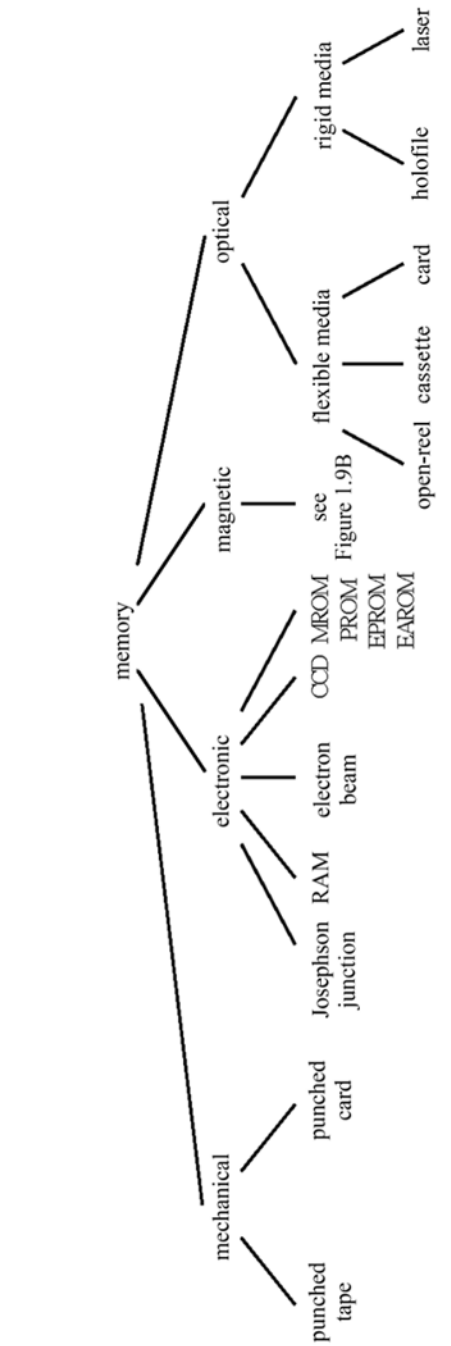


Figure 1.9(A). *Memory classification in 1978 (according to Puthuff (1978))*

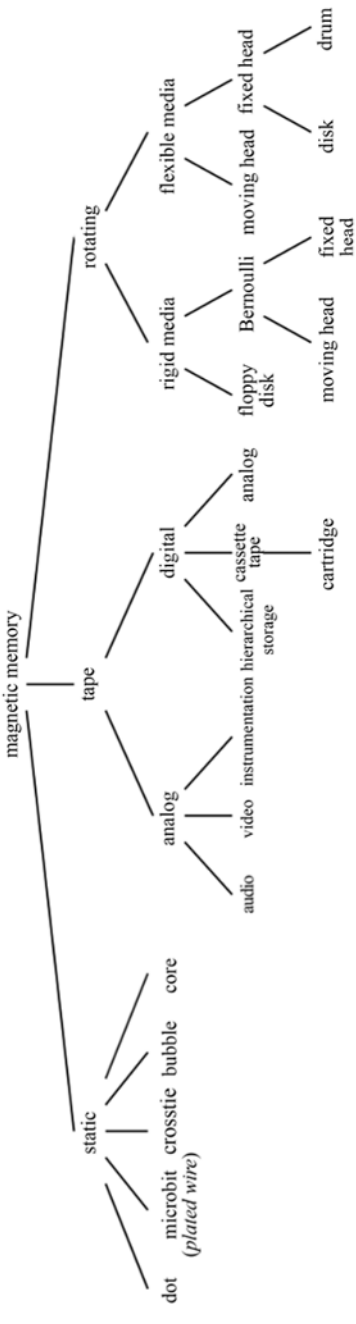


Figure 1.9(B). *Memory classification in 1978 (according to Puthuff (1978)) (continued)*

A main memory of several KiB made with tube-based flip-flops would have had significant weight and, most of all, occupied too much space. Also, researchers tested other technologies. Some of these made use of the properties of the electrostatic tube (see section 1.3.2.1). The first main memory (MM) technologies used the propagation time of a signal through a medium or component (delay line, see section 1.3.2.2) or the physical properties of materials, for example, magnetic properties with the core (see section 1.3.2.3). Large size, heavy weight and high power consumption characterized these early technologies.

Bell et al. (1978) reproduced a classification of the memory components and subassemblies of the time, taken from a study by Puthuff (1978) from Memorex (see Figure 1.9(A)). Four technological domains, both old and modern, are present: mechanical, electronic, magnetic and already optical. Computer peripherals using paper or cardboard as storage media were described in sections 5.6 and 6.5 of Darche (2003). Semiconductor technology (see section 1.3.2.4) with the transistor gradually replaced the electronic tube in the early 1960s and was later integrated. Note the presence of charge-coupled device (CCD) memory, which is also used as an optical sensor.

It was gradually replaced by the transistor (see Figure 1.17(a)).

The magnetic domain has the most branches, as shown in Figure 1.9(B). The magnetic medium in the form of a tape or disk is in motion. If it is static, it may be used as a component forming the cell of an MM. The core in particular is often used (see section 1.3.2.3).

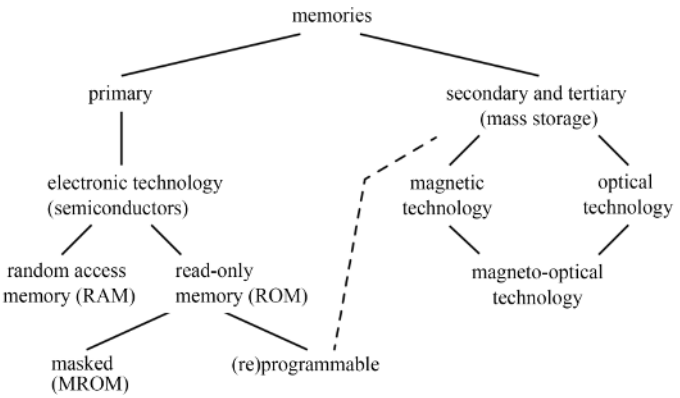


Figure 1.10. *The different memory technologies of a modern computer*

In 2025, technology is electronic, magnetic or optical. But the current trend is toward the elimination of moving components (electric motors, platters (i.e. disk)

arms, etc.). Thus, the use of reprogrammable ROM such as flash memory (dotted line in Figure 1.10; see also Volume 6) as components of secondary memory is becoming dominant. Each level of the memory hierarchy corresponds to one of these three technologies, as shown in Figure 1.10.

The next three sections detail some representative legacy families.

1.3.2.1. *Electrostatic memories*

The first representative is the Williams–Kilburn tube (Chu and Klein 1952). Eckert (1949) and Julian and Samuel (1949) describe it as a classical cathode-ray tube, or CRT (see Figure 1.11), which stores binary information as a small light dot on the screen or as a dash.

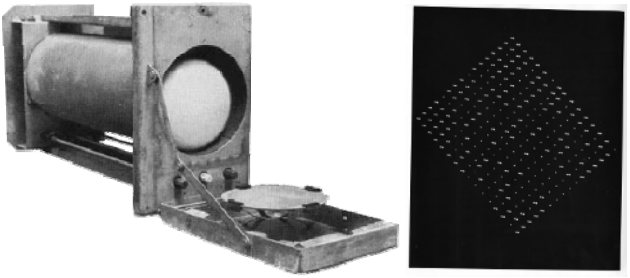


Figure 1.11. *The Williams–Kilburn tube and a view of its screen (source: left image: https://en.wikipedia.org/wiki/File:Williams-tube.jpg; right image: National Institute of Standards and Technology, Public Domain, https://commons.wikimedia.org/w/index.php?curid=31539966)*

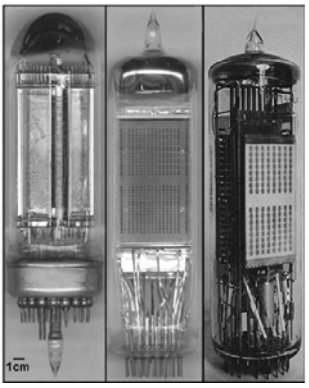


Figure 1.12. *The Selectron (source: https://rhartshorne.com/summer-2013/jamesralph/Multimedia%20Project/Evolution%20of%20RAM.html)*

The second is the Selectron (Selective Electrostatic Storage Tube), which was a special electronic tube (see Figure 1.12) able to store the value of a bit as a stable electric potential (holding beam memory tube). This type of memory could store 256 bits in a tube 3'' (= 7.62 cm) in diameter and 7'' (= 17.78 cm) tall. The highest capacity achieved was 4,096 bits. The access time was approximately 10 µs. This should be compared to the access time of early semiconductor memories, which was several hundred ns. It was first described in 1947 (Rajchman 1948) and given a detailed description by Rajchman (1949, 1951) and Knoll and Kazan (1952). This tube was manufactured by RCA (Radio Corporation of America). Due to lack of reliability, the Williams tube was preferred.

1.3.2.2. *The delay line*

A delay line slows the propagation of an electrical or mechanical signal (acoustic delay line). If the delay is long enough, it can store a train of logic pulses, provided the end of the line is looped back to its input. Thus, the delay line can serve as memory (delay-line memory). The delay can be caused by a material such as mercury (mercury sonic delay line) or quartz, or by a passive electronic component such as an inductor (i.e. a coil). The access time varies depending on the structure, and its maximum value equals the time to cross the line.

For example, the BINAC (Binary Automatic Computer) had, depending on the source, 18 memory channels of 32 words of 36 bits, giving a total capacity of 20,736 bits (20.25 KiB) (Eckert 1953), or 16 channels of 32 words of 31 bits, totaling 15,872 bits (15.5 KiB) (Schmitt 1988). The UNIVAC (Universal Automatic Computer), the first commercial computer, had 100 channels of 10 words of 91 bits, giving a total of 91,000 bits (nearly 90 KiB).

For more information on this component, Eveleth (1965) presents an overview of ultrasonic delay lines, and Zilberstein (1965) describes a variable-delay version.

1.3.2.3. *Ferrite core memory*

Ferromagnetic technology was the first used to create a non-volatile memory (ferromagnetic memory). Binary data was stored as a change in the orientation of a magnetic field in a fired toroidal ferrite core. The material properties are described by Greifer (1969). Figure 1.13 shows the curve $\Phi = f(I)$ of a ferrite core with a current-carrying wire passing through its center. Depending on the direction of the current, the core can be magnetized in one of two directions. It has two stable states when there is no supply current.

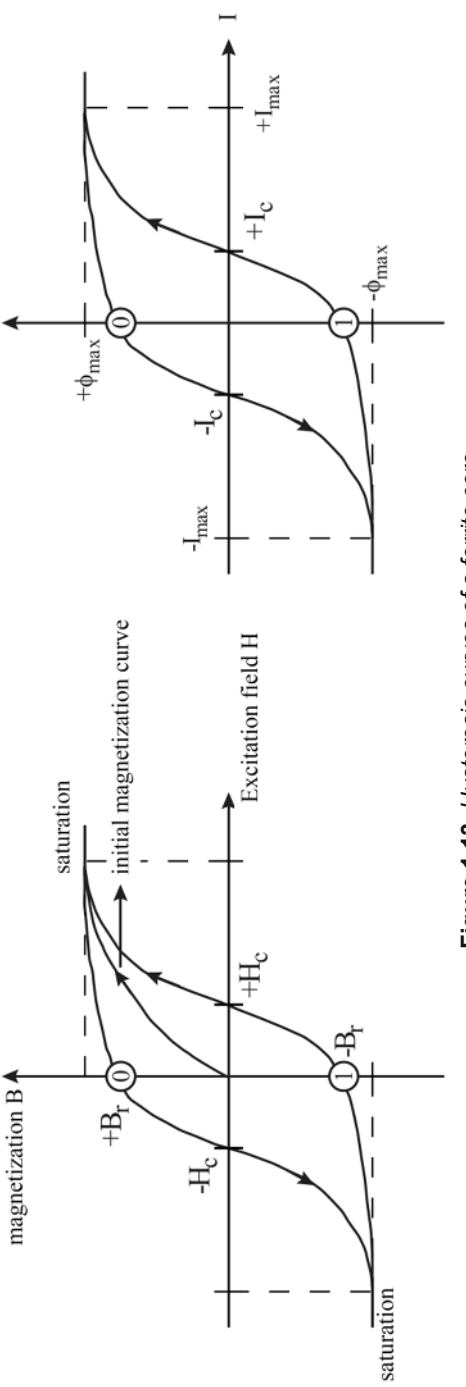


Figure 1.13. *Hysteresis curves of a ferrite core*

The sides of the loop were deliberately slanted to clearly show the key points of the cycle. Those of a real component are nearly vertical. It is thus easy to model the loop as a rectangle, as shown in Figure 1.14.

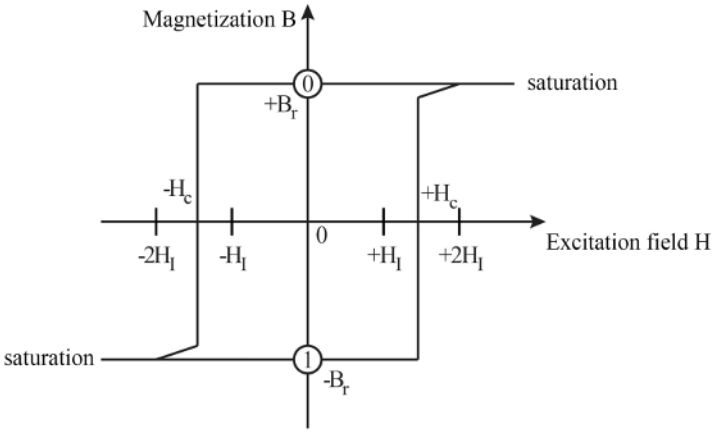
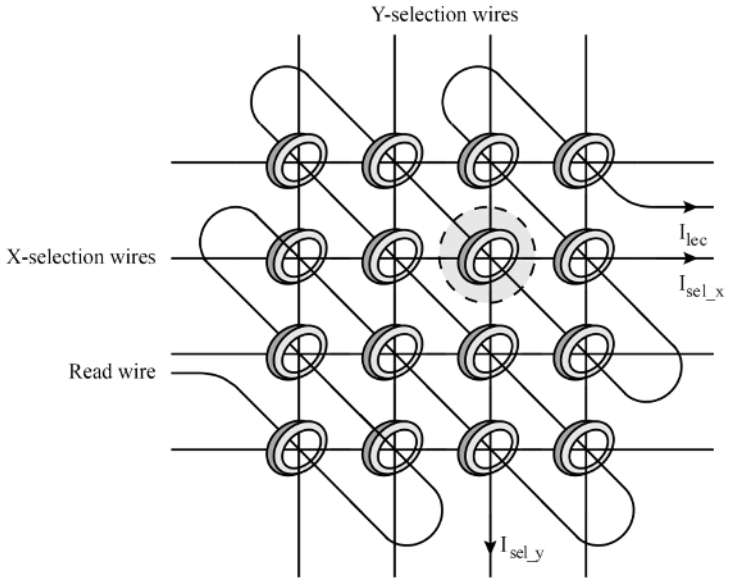


Figure 1.14. *Modeling of a ferrite core hysteresis loop*

The memory area was a wired matrix, with a core mounted at each intersection (gray oval in Figure 1.15), hence the name core memory. The typical outer diameter ranged from 0.08" (2.032 mm) to 0.4" (10.16 mm), with an inner diameter of 0.05" (1.27 mm). A specific core was addressed by coincidence. Writing was done by sending a current pulse simultaneously on one row and one column (coincident current selection). The polarity of these currents determined the magnetization direction. A single pulse was not enough to flip the magnetization of other cores on the same row or column as the selected core (half-select currents). Reading required a dedicated read wire passing through all the cores in series (three-wire core). Reading was performed by writing a known logic state, say "0". If the selected core held the opposite state, a large current spike appeared on the read line. If not, no current spike was detected. Reading was therefore destructive. A rewrite was required if the state had changed.

A trick even allowed doubling the number of cores (i.e. capacity) per address plane by using the anti-current concept. Indeed, for a given core, there are four possible current combinations: two useful (i.e. switching possible) and two that cancel out the produced magnetic field. Depending on the current direction, one of two cores could be selected, as shown in Table 1.4 and Figure 1.16.

a)



b)

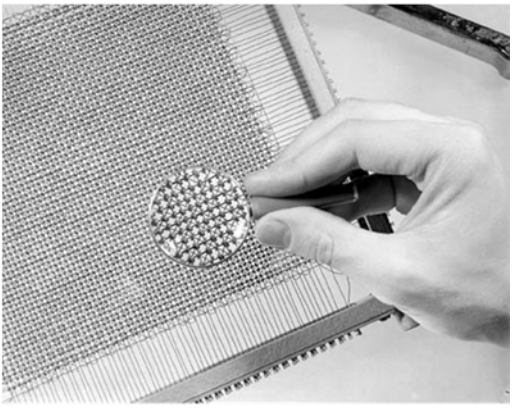


Figure 1.15. *a) Matrix-organized core memory (according to Rajchman (1957)), and b) a real memory (source: The MITRE Corporation)*

Selection		State	
X	Y	C_1	C_2
–	–	0	–
–	+	–	1
+	–	–	1
+	+	1	–

Table 1.4. *Truth table: state = f (current direction)*

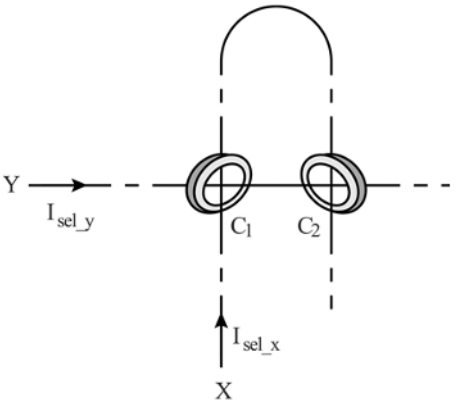


Figure 1.16. *Selection of one of two cores using anti-current technique*

The selection current ranged from 400 to 800 mA. The write time was between 3 and 5 µs. Since reading (a pulse of 50–75 mV) was destructive, the information had to be regenerated. Depending on the source, the invention is attributed to An Wang, researcher at Harvard Computing Lab in 1948 (Wang 1955), or to Jay Forrester of the Whirlwind project (Forrester 1951), who held the patent (Forrester 1956). Wang and Woo (1950) proposed a magnetic delay-line approach, while Forrester (1951) preferred the matrix-based solution. He even discussed a three-dimensional structure (3D, see section 3.7). An example of the patent is Bornhauser (1964). Rajchman (1957) reviews the early memories. In 1968, a core memory had a capacity of 4,096 bits with an access time of 120 ns. The Apollo 11 AGC (Apollo Guidance Computer) used core memory organized as 2 Ki × 15-bit RAM and 36 Ki × 15-bit RAM. Using that much ROM was not surprising, as it helped resolve the problem of overwriting of information (code or data) by a program, whether malicious or faulty (see memory protection problem in Volume 9). This memory was later developed in various formats, such as toristor, thin-film dot memory and cylindrical (magnetic rod).

Pugh et al. (1981) and Chapter 7 of Bashe et al. (1986) provide details about these memories in IBM (International Business Machines Corporation) computers. For detailed specifications, see Scrupski (2001). ANSI/IEEE standard 393 (ANSI/IEEE 1991) defines vocabulary and test procedures related to this component. With the development of discrete-transistor and then integrated memories in 1965–1966, core memories gradually disappeared since they could not compete in size, weight, power consumption, capacity and cost with their integrated counterparts. The final core models had an outer diameter as small as 0.0015" (0.0381 mm). DRAM in its modern form, first commercialized by Intel in 1971, marked the end of this technology. There remained niche markets, such as radiation-hardened memory, since radiation could cause storage errors (soft errors; see section 6.4). For instance, NASA's first shuttle used core memory of 104 Ki × 32 bits with an access time t_a de 400 ns. A presentation of this memory is given by Pugh (2000).

1.3.2.4. *Semiconductor memories*

Semiconductor memories, initially in discrete form, using diodes and then transistors (invention by Bardeen and Brattain (1948)), and from the 1970s in integrated form (see Figure 1.17(b)), have continually increased in capacity. Manufacturing technologies have evolved over 60 years, but the memory principles and internal structure have paradoxically changed very little. Memories represent one of the largest market shares of integrated circuits, but paradoxically yield little profit.

a) b)

Figure 1.17. *a) Germanium transistor and b) Intel's 1103 P-channel MOS DRAM (source: Wikipedia)*

The industrial market is mainly segmented into three parts: static RAM (SRAM), dynamic RAM (DRAM) and programmable ROMs, represented by EEPROMs (electrically EPROM), flash, or FEEPROM. They are used in all subsystems and components of a computer – inside the processor, in primary memory, in

intermediate memory between the processor and main memory (cache), in I/O interfaces, and in peripherals.

The static model uses a memory principle based on self-sustained electrical states. Thus, the storage is persistent without auxiliary mechanisms and is not very sensitive to external disturbances such as radiation or electrical noise (see Volume 2).

The dynamic model, however, uses the principle of storing electric charge in a capacitor (stored-charge memory). This capacitor is imperfect and has a leakage current. The stored information is not persistent and must be periodically refreshed. DRAM originally operated with a clock (synchronous model), then quickly became asynchronous in 1971 (see Volume 3). Since 1996, dynamic RAM evolved again toward synchronous operation for speed, with models such as synchronous communication per cycle (SDRAM; see Volume 4) and communication per packet (see Volume 5).

The information in ROM was implemented using a photolithographic mask (MROM or mask-programmed ROM (JEDEC)). Programmable models, or PROM (programmable ROM), then reprogrammable models, erasable by ultraviolet light (UV), such as EPROM (erasable PROM), or electrically via EAROM (electrically alterable ROM) and EEPROM (electrically EPROM) later appeared (see Volume 6).

1.3.2.5. *Historical timeline*

In summary, Figure 1.18 presents a chart inspired by Proebster (1992) that recalls, for each level of the memory hierarchy, the technologies used to store information. Technological shifts were gradual due to practical and economic reasons, which are not shown in the figure for graphical clarity. Starting in the 1970s, the LSI (large scale integration) generation enabled the emergence of early memories with microprocessors built using bipolar and unipolar technologies. The first category included the "standardized" TTL (transistor-transistor logic) and ECL (emitter-coupled logic; see section 2.3.3 of Darche (2004)) families. The second was called MOS (metal–oxide semiconductor), with PMOS (positive-channel), then NMOS (negative-channel) and later CMOS (complementary MOS) subfamilies. Gallium arsenide (GaAs) technology is mentioned anecdotally. VLSI circuits (very LSI) appeared in the 1980s. Subsequent decades saw the emergence of ULSI (ultra LSI) and GSI (gigascale integration), the latter allowing the integration of an entire digital system or SoC (system on a chip). It is noteworthy that punch cards were still (occasionally) used in the 1980s, for example, to dump memory on Bull's Mitra 625–725 machines (personal experience). A mass memory, or MSD (mass storage device), stores large quantities of data. Excluding tape and punch cards, the initial technology was magnetic, with hard disk drives and floppy disks. A mass storage

system, or MSS, automatically manages a set of magnetic tapes as cartridges or cassettes (see section 1.5). For more information, Williams (1997) presents a historical overview of all the technologies discussed in this chapter. Eckert (1953), in turn, presents other technologies based on optical, magnetic or even thermal properties! Pugh (1984, 1985, 2000) details the key role played by IBM in developing memory technologies.

1.4. Memory in a microcomputer

Figure 1.19 shows memory in a modern PC-type microcomputer. The main element is the microprocessor, the sole master of data exchanges[10]. The other components, which are slaves, are the main memory and the I/O interfaces. All these subsystems communicate via buses. In electronics, a bus is a set of electrical wires or copper traces on a printed circuit board. A bus controller manages access (see section 1.6 of Darche (2021b)). It thus prevents conflicts and ensures data validity. The figure shows a dual independent bus (DIB) architecture, with a front-side bus (FSB) and a back-side bus (BSB). The BSB connects the microprocessor to the cache memory buffer (see Volume 8). Bus controllers are integrated into a component called the HUB or bridge, which acts as a true communication hub for all subsystems.

Today, most I/O interfaces are integrated into a single chip, called the ICH (I/O controller hub) or south bridge by Intel, to reflect its diagram position. Only two interfaces (network and mass storage) are shown in the figure.

However, the display interface communicates primarily with the microprocessor and main memory, except in low-end microcomputers, where it may be integrated into the north bridge, then called the GMCH (graphics and memory controller hub) by Intel. This north bridge connects to the microprocessor via the FSB. The BIOS (basic input/output system; see Chapter 4 of Darche (2003)), the software layer (software or SW) that manages I/O controllers, is stored in the ROM (firmware or fw); for a PC, this ROM is connected to the south bridge. The main memory is then partitioned, either statically or dynamically, to be shared between the microprocessor and the display controller. This is known as unified memory architecture (UMA). Without this interface, the bridge is called the MCH (memory controller hub).

10 For our level of discussion, as other components can also manage exchanges.

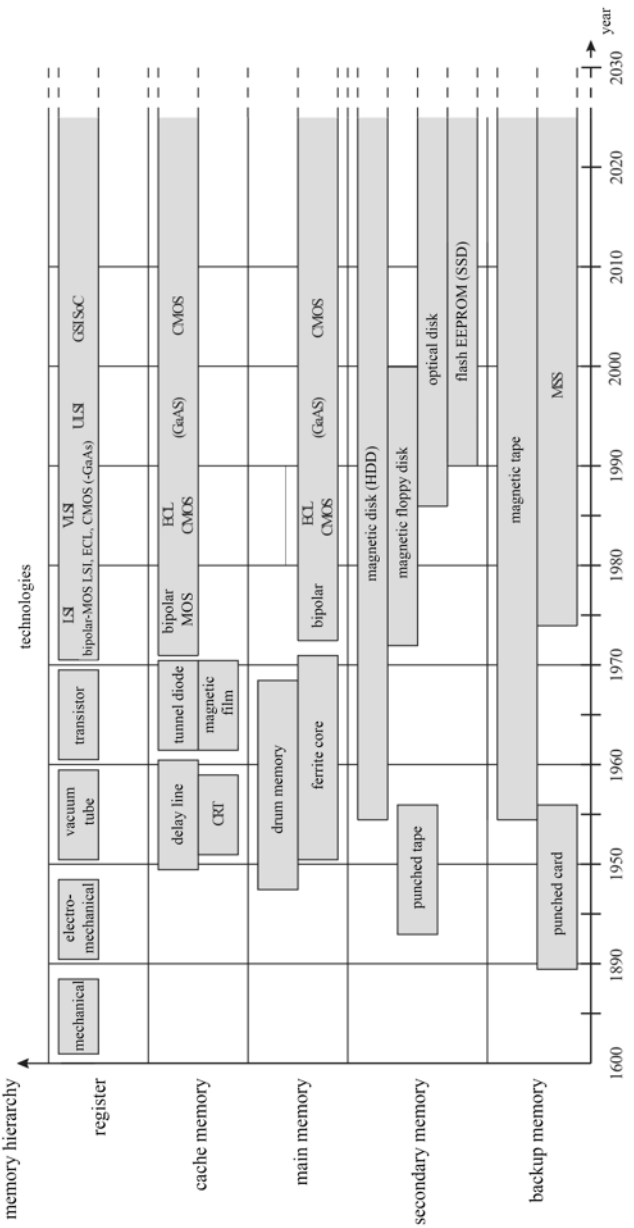


Figure 1.18. *Historical evolution of memory technologies (inspired by Proebster (1992))*

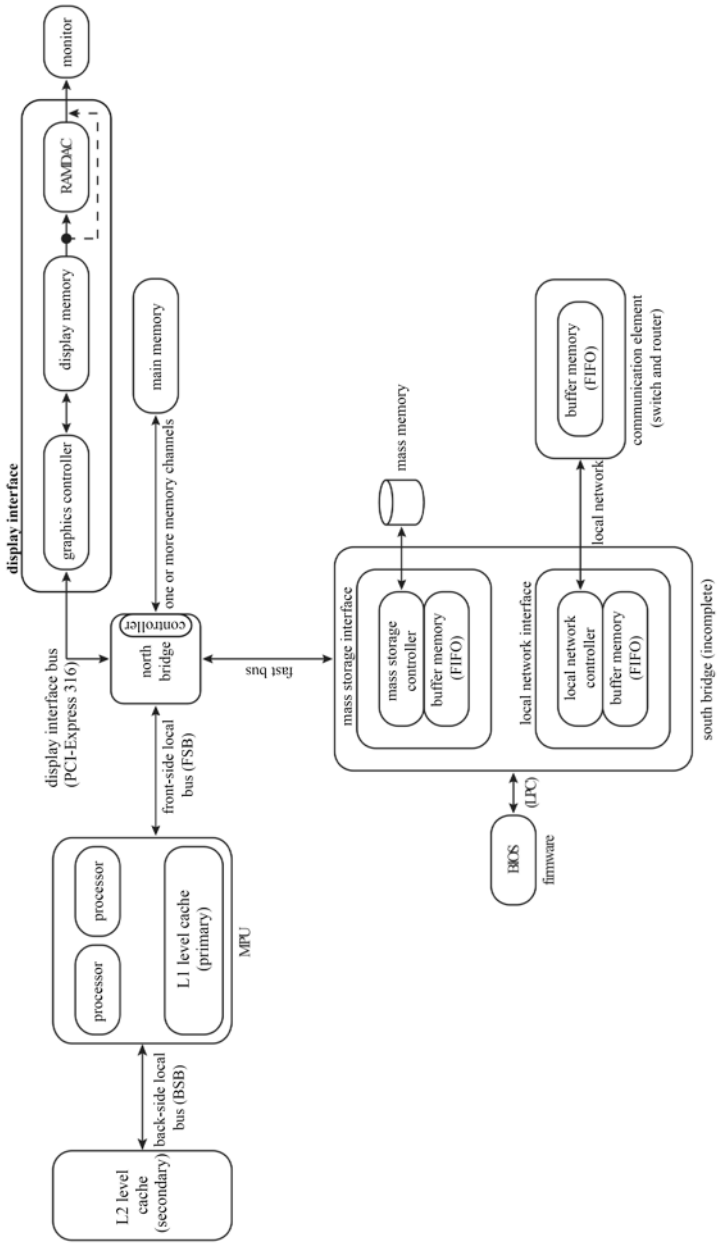


Figure 1.19. *Semiconductor memory in a modern PC-type microcomputer in 2001*

The main memory was initially a scarce resource, as the first computers had a memory of approximately a kilobyte. Mechanisms had to be created to improve its use, such as overlay (a Pascal language "overlay" module or file), virtual memory (VM) or swapping (i.e. exchanging). It must often meet contradictory criteria. For example, it must be large and fast, while taking up little space, using little power and costing less. Using buses with their associated logic introduces latencies, but integration advances have made more space on the chip for RAM and related logic. As a result, bandwidth increases while latency decreases (Patterson et al. 1997). Today, memory size is comfortable, but access time is still generally 10 times slower than that of a microprocessor register (see Table 1.5). RAM remains the second largest cost after the microprocessor when buying a personal computer. In 2004, main memory in PCs represented 50% of the DRAM market (data from Rajan (2009)).

Cache memory, inserted into the data path, improved the latency of primary memory. It is a key element in computer performance and in spatial footprint. We only have to look at a microphotograph of a multicore chip such as Core™2 Duo (Intel) to see that cache takes up nearly half its surface area (left part of Figure 1.20).

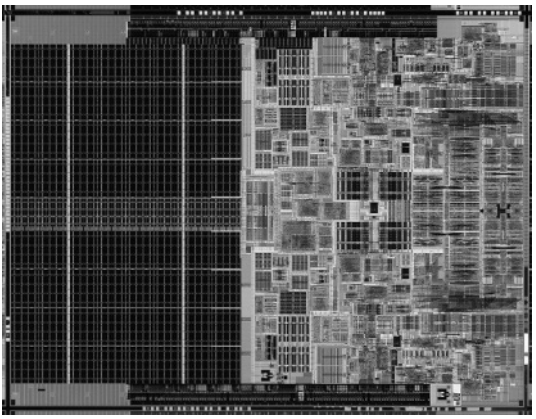


Figure 1.20. *Microphotograph of the Core™2 Duo microprocessor chip (source: Intel)*

As can be seen, no memory family, each of which has sometimes opposite characteristics, meets all the needs of electronic systems, whether in terms of capacity/density, volatility, access time (read) or cycle time, hence the idea of a hierarchy.

1.5. Memory hierarchization

> We are therefore forced to recognize the possibility of constructing a hierarchy of memories, each of which has greater capacity than the preceding but which is less quickly accessible (Burks et al. 1946).

It is now accepted that memory capacity and access speed needs cannot both be fully satisfied at a reasonable cost. Boland and Dollas (1994) showed that the number of processor cycles needed to access main memory doubled approximately every 6.2 years. A memory built with the same technology will, after n years, require a number of cycles equal to:

$$L(n) = L_0 \times 2^{\frac{n}{6.2}} \quad [1.10]$$

L_0 is the base latency.

This is called the performance gap (i.e. speed) between memory and the processor (memory gap). This gap between the processor cycle time and memory time first appeared for microprocessors in the early 1980s. As a result, processors were delayed during external accesses, as shown in Figure 1.21 (f_{DRAM} = access time^{-1}). Synchronous architectures (see Volume 4) reduced this gap, though without eliminating it.

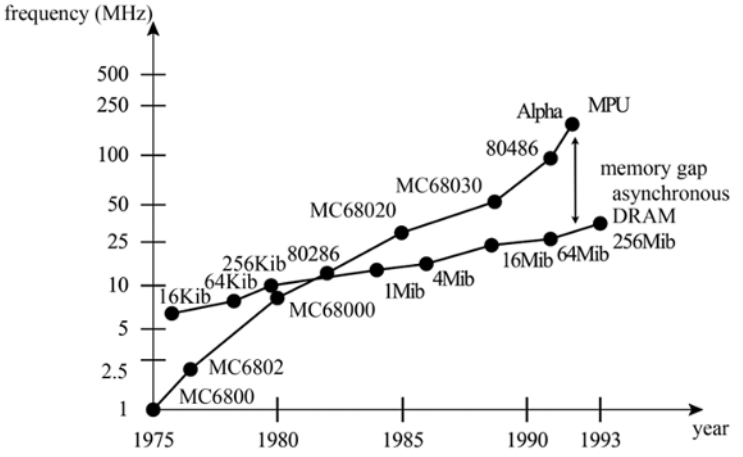


Figure 1.21. *Speed gap between MPU and asynchronous DRAM (Slater et al. 1993)*

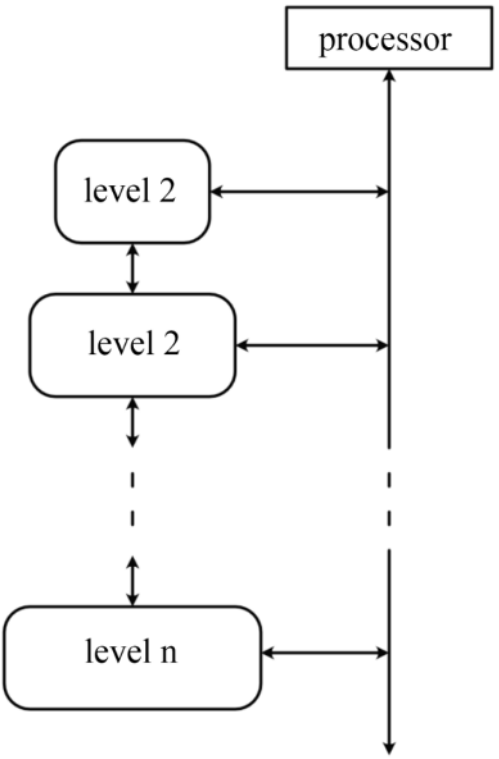


Figure 1.22. *Processor access to the memory hierarchy*

Therefore, to reduce the average access times for processor instructions and data, and for cost reasons, several memory technologies were inserted into the data path. Figure 1.22 shows the access paths available to a processor. The area of the rectangles represents access time and capacity (not to scale). Communication between levels is possible thanks to specialized controllers such as cache or DMA (direct memory access) controllers.

These different memories were modeled under the name “memory or storage hierarchy”. This abstraction organizes storage components, subsets and peripherals vertically by levels (multi-level memory hierarchy) or layers, depending on their position in the data path and on a technical characteristic. The hierarchy is usually shown as a triangle or even a pyramid (Nakagomi 1993). The lower an element is, the farther it is from the PE.

Figure 1.23(b) shows a classic hierarchy. At its top is the flip-flop (in static logic implementation), a basic memory element (i.e. one bit). Next, it consists of the register, which is a set of flip-flops. The following layers include on-chip cache (internal to the MPU), or tightly coupled memory (TCM; see section 5.3), then possibly an off-chip cache, then primary, main or central memory, followed by secondary memory, and finally tertiary memory, also known as backup memory (tertiary memory system[11], offline backup storage or backing store (Handy 1998)).

Not shown here, archival memory allows for storing data over decades. These mass storage systems or MSS consist of libraries of hundreds of magnetic tape cartridges or optical disks, moved by robotic arms to insert them in reading/writing devices. Local memory is the memory integrated into the processor, i.e. registers, embedded memory and internal cache.

Since manufacturing technologies are heterogeneous, each level or layer has its own technical features, whose values rise or fall discretely, i.e. by steps (see Figure 1.23(a)). The surface or base of the trapezoid is meant to represent the magnitude of these values. In general, we focus only on four features: total storage capacity, cycle and access times, bandwidth and finally the cost per bit stored. The latter decreases as capacity increases. Thus, memory hierarchy offers the largest amount of memory at the lowest cost, while providing the fastest possible access. In general, and especially for cache levels, the frequency of access increases as you go up the hierarchy.

Figure 1.24 details types of components or devices used at each level. Tertiary memory, used for backup in case of failure, includes systems that are online, near-line (cartridge) or offline. The hierarchy is managed either by hardware or software. Cache levels are handled by dedicated controllers integrated, for example, in PCs' north bridge chipset (see section 3.3 of Darche (2021e)), now integrated into the MPU. Primary memory is managed by the dynamic memory controller. Secondary memory is managed by the virtual memory controller or MMU (memory management unit) associated with the OS.

11 Following footnote 1 of this chapter, secondary and tertiary memories are called mass storage memories: secondary and tertiary storages. The origin of these terms is found in psychophysiology, in the work of James (1890). This American psychologist called primary memory the fleeting store of thought in consciousness and secondary memory the permanent but unconscious store.

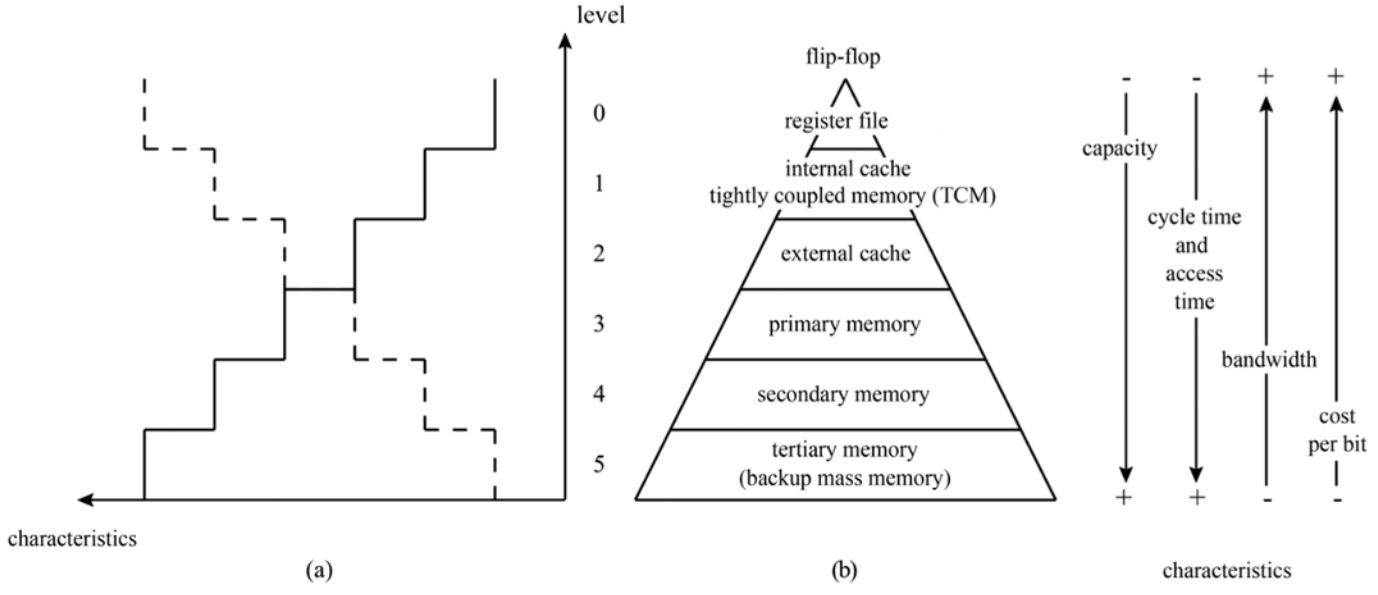


Figure 1.23. *Memory hierarchy*

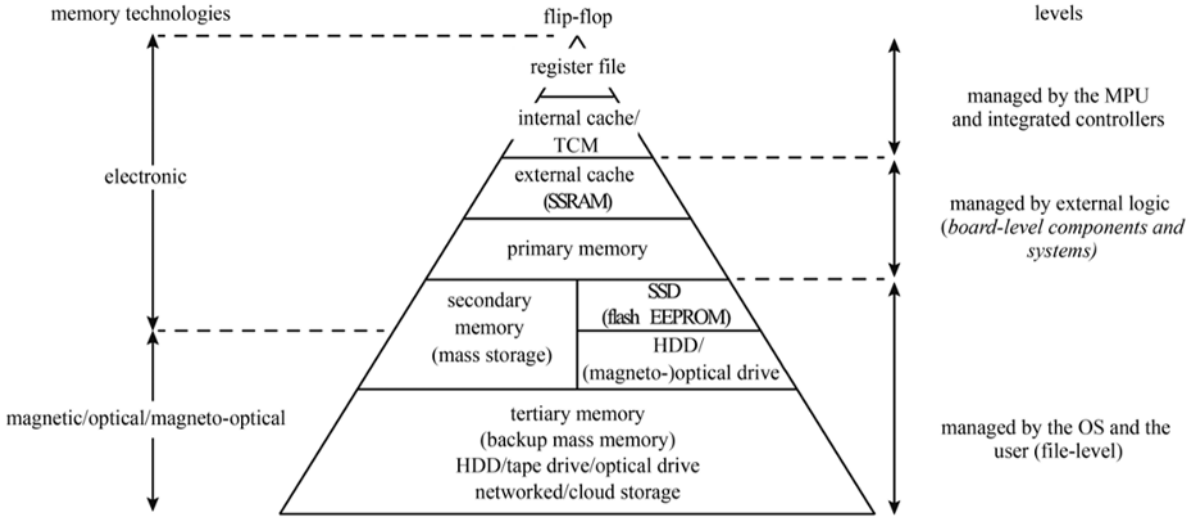


Figure 1.24. *Detailed memory hierarchy*

Cohen et al. (1989) presents an example of a memory hierarchy for a large system (mainframe) IBM 3090 (see Figure 1.25). The central unit consists of three processors, each with a cache called the HSB (high-speed buffer) (level 0). Levels 1 and 2 represent central memory (L3 and L4). Mass storage consists of hard disk units called the DASD (direct access storage device), preceded by a disk cache (Grossman 1985).

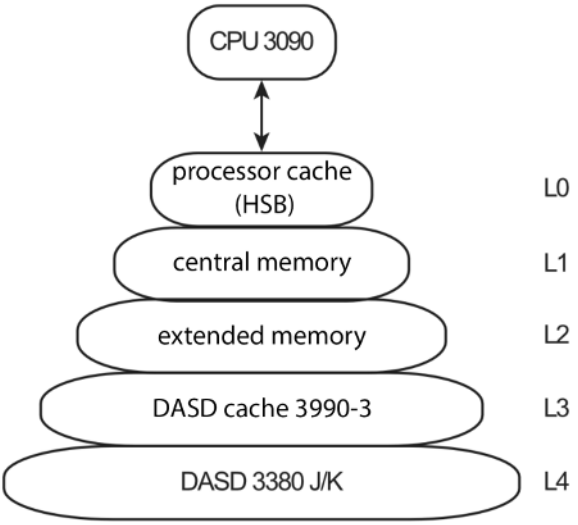


Figure 1.25. *Simplified memory hierarchy of an IBM 3090 (according to Cohen et al. (1989))*

The hierarchy is therefore not just a simple classification of memory elements by capacity, as some books have presented. It represents the various memory elements accessed successively during the execution of an instruction or when accessing data. Disregarding backup mass memory, the memory hierarchy can be viewed as a single address space available to the software architect. It is therefore a part of the hardware architecture, as defined in section 3.1.4 of Darche (2021a). To be fully complete, we must also consider the data flow exchanged between the processor and these elements, called the "storage hierarchy data flow". This diagram represents the physical structure of the hierarchy of a mainframe computer. Figure 1.26 shows the one from the previous example. It highlights a key element in operation, the system controller (system control element or SCE), which manages memory access for the processors.

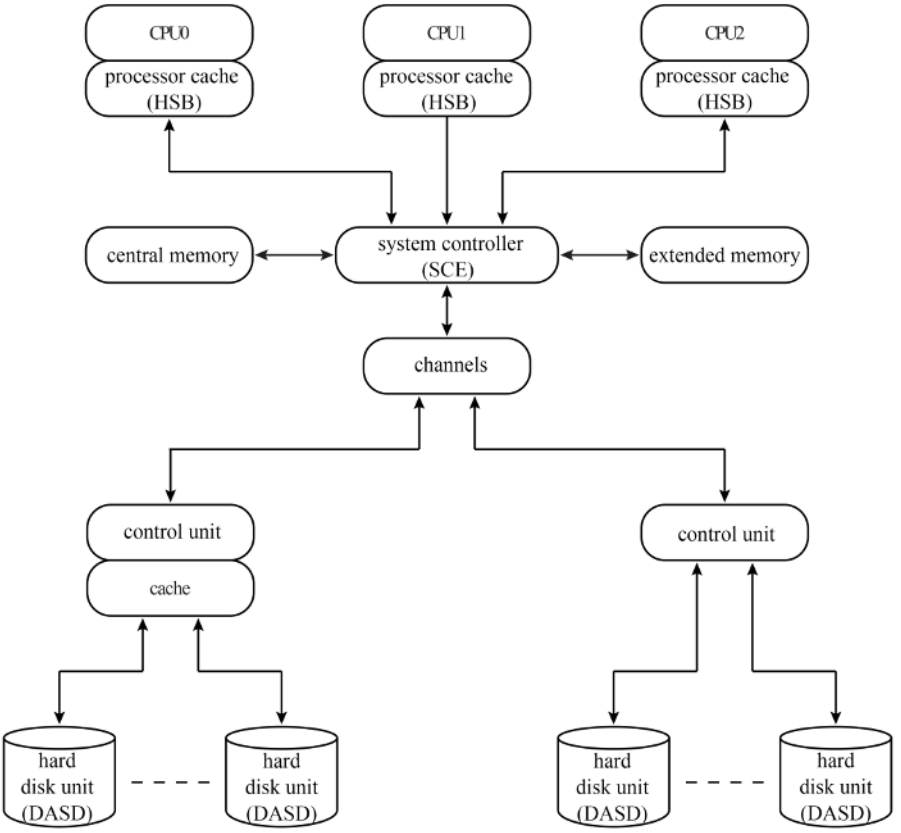


Figure 1.26. *Data flow of the memory hierarchy of an IBM 3090 (according to Cohen et al. (1989))*

Thanks to queueing theory, it is possible to model the transfer of information between the different subsystems of a computer. An example of such a study can be found in Rege (1976). From that, we can calculate theoretical throughputs. The choice of technical characteristics at each level will affect the overall computer performance and especially that of the operating system. For example, Hake and Homberg (1990) studied the impact of memory organization on matrix multiplication performance. Kaneko (1974) studied the influence of memory hierarchy on the OS task switching policy. Table 1.5 summarizes current values of the characteristics for each level. Note that the raw throughput of a memory is the intrinsic throughput of the memory. It does not involve the interface. To provide historical context, Pugh (1971) presents data from 1952 to 1972.

Level	Type	Size (bytes)	Average access time	Factor	Raw throughput (order of magnitude from access time for SC memories)	Cost/MB ($)
0	Register	Number: 8/16/32 (CISC) to 32/128/256/512 (RISC)	0.25 ns	1	32 GB/s (n = 64)	–
1	Internal L1 cache (Itanium 2 MP Intel)	32 Ki	1 ns (1 cycle)	4	8 GB/s (n = 64)	–
2	External L2 cache (Itanium 2 MP Intel)	256 Mi	5 ns (5 cycles)	20	1.6 GB/s (n = 64)	–
3	External L3 cache (Itanium 2 MP Intel)	9 Mi	12 ns (12 cycles)	48	667 MB/s (n = 64)	1.5
4	Primary memory	1 Gi	30 ns	120	267 MB/s (n = 64, random)	0.01
5	Secondary memory	1 T	10 ms (HDD)	40×10^6	250 MB/s (SATA Revision 3.0)	0.0002
6	Tertiary memory	> 10 T	30 s (average)	120×10^9	160 MB/s (LTO Ultrium 5)	0.0001

Table 1.5. *Order of magnitude of technical specs for memory levels in a microcomputer (2006)*

Level	Type	Size (bytes)	Average access time	Factor
0	Register (without pipeline)	Number: 8/16/32 (CISC) to 32/128/256/512 (RISC)	0.175 ns	1
1	Internal L1 cache/core (Zen 4 CCD AMD)	32/32 Ki (data/instructions)	0.175 ns (1 cycle)	1
2	Internal L2 cache/core (Zen 4 CCD AMD)	1 Mi	2.45 ns (14 cycles)	14
3	L3 cache/core (Zen 4 CCD AMD)	4 Mi	8,75 ns (50 cycles)	50
4	Primary memory	32 Gi (DDR5 SDRAM (DDR for Double Data Rate))	10 ns	57
5	Secondary memory	1 T	0.1 ms (SSD)	570×10^3
6	Tertiary memory	> 10 T	30 s (average)	170×10^9

Table 1.6. *Order of magnitude of technical specs for memory levels in a microcomputer (2023)*

Williams (1973) studied a memory hierarchy where the read time of mass memory (backing storage) is very small compared to that of conventional mass memories or compared to the write time, as is the case with optical mass memory units. This type of unit is called “asymmetric backing storage” and the associated hierarchy, “asymmetric memory hierarchy”.

Table 1.5 should be compared with Table 1.6. Between the two dates, Moore's law remains (mostly) valid (see section 1.5 of Darche (2021a)). Integration continues across all cache levels, and multicore architecture becomes standard in MPUs. The clock frequency of processors stagnates at approximately 6 GHz. Secondary memory has changed technology. It has gradually shifted from magnetic to electronic using FEEPROM memory in SSDs. This choice improves access time by a factor of 100 compared to using an HDD.

Designing this hierarchy well is essential for computer performance. This issue is summarized as the "memory wall" by Wulf and McKee (1995), showing processor performance grows faster than memory.

1.6. Conclusion

This chapter was dedicated to the memory function. The main characteristics of a generic memory, such as capacity, organization, policies and types of access, and timing characteristics, were defined. Historical and technological aspects were then presented, leading to the presentation of the memory hierarchy.

The designer of a memory system must make a compromise between storage density, latency and throughput, power consumption and the cost of each layer in the hierarchy.

Chapter 2 presents the internal organization of a random-access semiconductor memory with its main functional subunits.