Computer Architecture and operating systems

Theoretical background

Part I

Krystian Wojtkiewicz

Wrocław, Poland

https://i-cu.eu

Table of contents

© Krystian Wojtkiewicz

Computer definitions

Computer – general definition

A data processing device equipped with the ability to input, store and output of data

Computer – Oxford dictionary

an electronic machine that is used for storing, organising, and finding words, numbers, and pictures, for doing calculations, and for controlling other machines

Computer – Technopedia

a machine or device that performs processes, calculations and operations based on instructions provided by a **software** or **hardware** program. It is designed to execute applications and provides a variety of solutions by combining integrated hardware and software components.

Computer - Stanford course

Computers have two main parts: hardware and software. „Hardware" refers the physical parts of the computer, and „software" refers to the code that runs on the computer.

© Krystian Wojtkiewicz

Computer architecture taxonomy

Taxonomies are used as a tool for classification providing means for building hierarchies and defining properties

Flynn taxonomy [1]

Pretty simple, must to know[1].

Skillicorn taxonomy [4]

Allows for better understanding of computer structure. It might be useful for the course comprehension[2].

[1] M. J. Flynn. "Some Computer Organizations and Their Effectiveness". In: *IEEE Transactions on Computers* C-21.9 (Sept. 1972), pp. 948–960. ISSN: 0018-9340. DOI: 10.1109/TC.1972.5009071.

[2] David B. Skillicorn. "A Taxonomy for Computer Architectures". In: *Computer* 21.11 (Nov. 1988), pp. 46–57. ISSN: 0018-9162. DOI: 10.1109/2.86786. URL: https://doi.org/10.1109/2.86786.

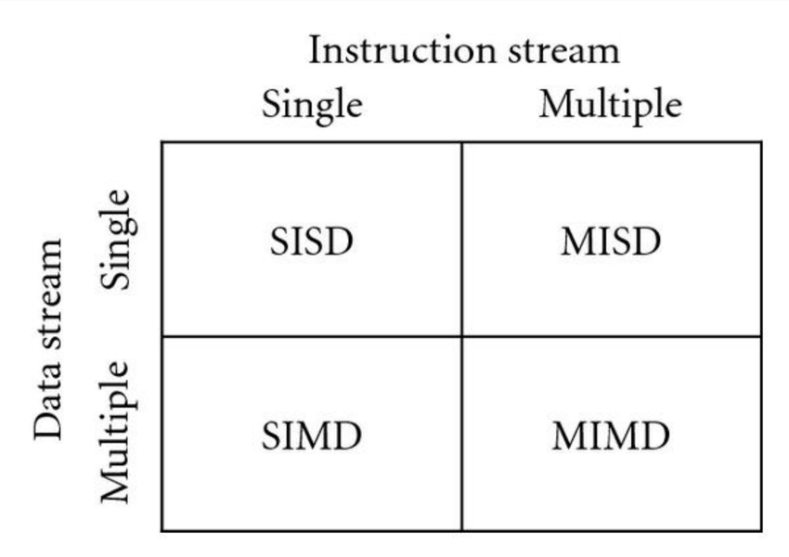
© Krystian Wojtkiewicz

Flynn taxonomy

- Proposed in mid 70's by J. Flynn
- It assumes computer to be a devise computing streams of data according to stream of instructions
- Classifies computers according to number of streams both, instructions and data

© Krystian Wojtkiewicz

- SISD: Single instruction single data
  - Classical von Neumann architecture
  - most common architecture
- SIMD: Single instruction multiple data
- MISD: Multiple instructions single data
  - Non existent, just listed for completeness
- MIMD: Multiple instructions multiple data
  - Most common and general parallel machine



What about 0 (zero) data streams?

Basically speaking, the computer without data is not considered as computer by Flynn. However, we could try to put here *automaton*. If we have to...

What about 0 (zero) instruction streams?

They might not be computers, but they got their name – **dataflow architecture**[3] [5]. There are uni and multi dataflow architectures.

[3] Arthur Veen. "Dataflow Machine Architecture.". In: *ACM Comput. Surv.* 18 (Dec. 1986), pp. 365–396. DOI: 10.1145/27633.28055.

General properties

Data provide information how they should be evaluated.

Token

It is a basic processing unit. It consist of **data** and a **tag** depicting its content.
The **tag** is similar in function to instruction, i.e. it provide means for data processing.

Processing

The effect of data processing is a new token with possibly modified data and a new tag.
Such a new token can be further processed.

and...?

No real devices using dataflow architecture has ever been build for commercial use.
Dataflow processing paradigm is in use, however it does not concern hardware.

© Krystian Wojtkiewicz

History

At present the Flynn taxonomy is considered to be important only for its historical value.

Good to know

It is considered to be important to understand acronyms used for each architecture type, since they are still being used.

© Krystian Wojtkiewicz

Skillicorn taxonomy

- Proposed around 1988 by David Skillicorn
- It assumes computer to be build from multiple elements
- There are two levels of complexity, however we will go into detail only with one

© Krystian Wojtkiewicz

Abstract taxonomy elements

They are abstract because there is no connection between Skillicorn model elements and physical computer elements

Elements

- Instruction Processors (IP) – a functional unit that acts as interpreter for instructions, when such things explicitly exist in the model on computation,
- Data Processor (DP) – a functional unit that that acts as a transformer of data, usually in ways that correspond to basic arithmetic operations,
- Data Memory Hierarchy (DMH) – an intelligent storage device that passes data to and from the processors,
- Instruction Memory Hierarchy (IMH) – an intelligent storage device that holds instructions,
- switch – an abstract device that provides connectivity between other functional units.

© Krystian Wojtkiewicz

Remarks

- Processors do not have any storage elements,
- Each processor have one linked MH, even if it is not a case on physical level,
- Instruction Processors is linked with Instruction Memory Hierarchy,
- Data Processor is linked with Data Memory Hierarchy,
- the number of DP can be 1 or higher,
- the number of IP can be either 0, 1 or N

© Krystian Wojtkiewicz

Elements connections

- Processors with memory hierarchies
- Data Processors with Instruction Processors
- Each type of Processors with same type of Processors

Possible multiplicity

- $1 - 1$
- $1 - N$
- $N - N$
- $N \times N$

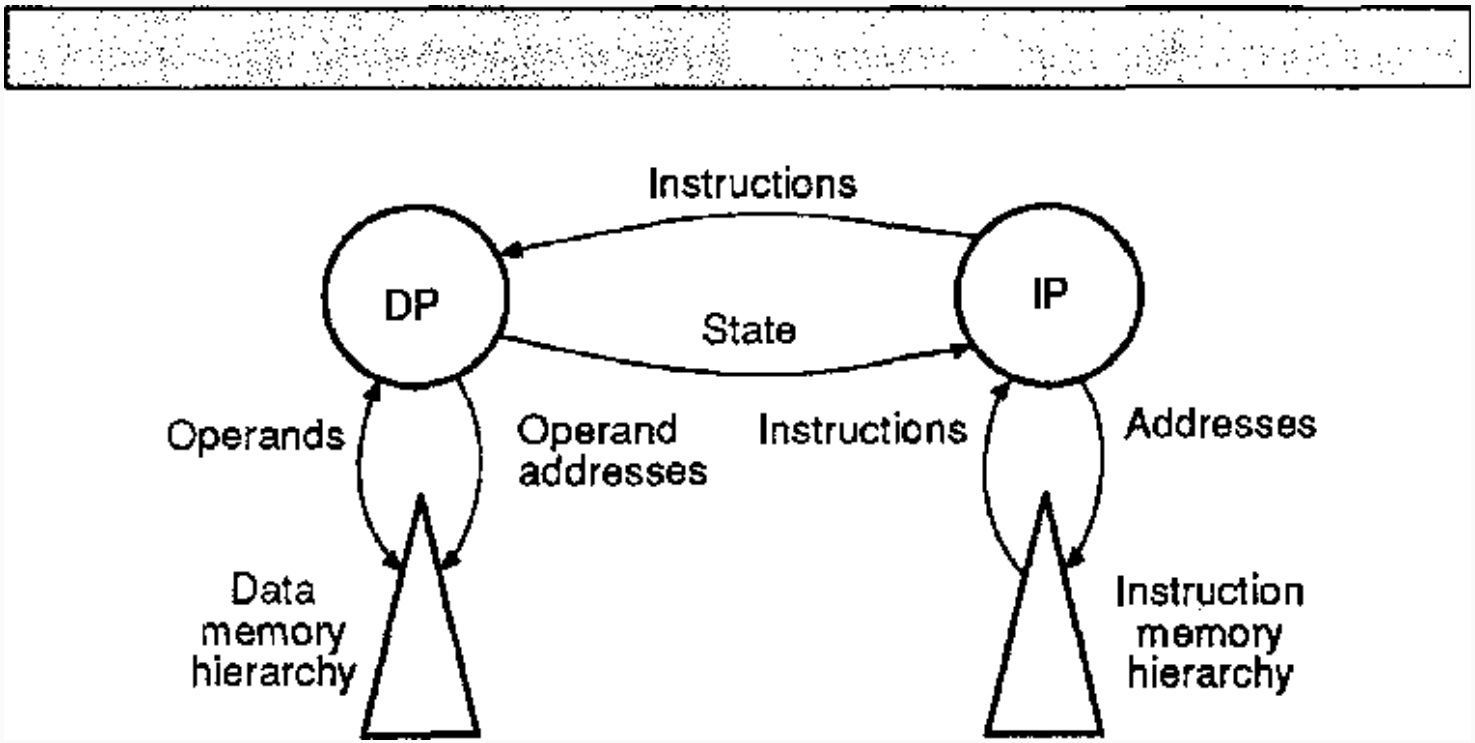


Figure: Arrangement of the von Neumann abstract machine functional units [4]

Instruction Processors (IP)

1. Determine the label of the next instruction to be executed, using information from its own internal storage and the state information conveyed to it by the data processor.
2. Instruct the memory hierarchy to provide the instruction with that label.
3. Receive the instruction and decode it. This involves determining which operation, if any, the data processor will be required to do to “execute” the instruction.
4. Inform the data processor of the operation required.
5. Determine the labels of the operands and pass them to the data processor.
6. Receive the state information from the data processor (after completion of the operation).

© Krystian Wojtkiewicz

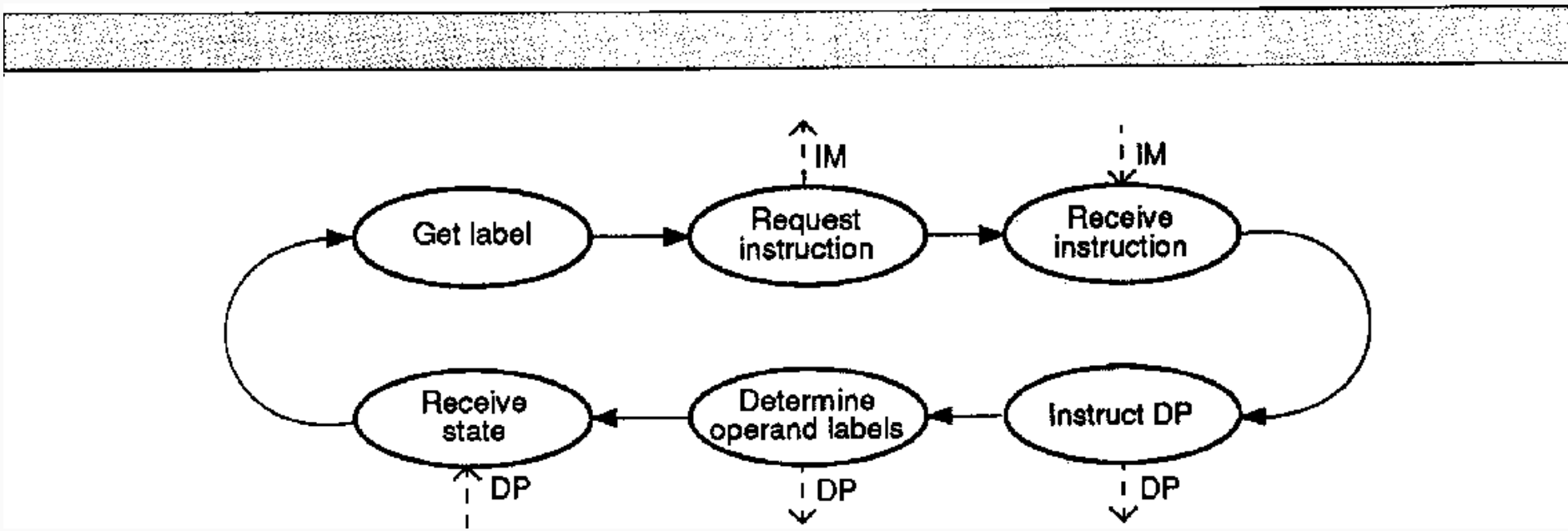


Figure: The internal structure of a von Neumann instruction processor [4]

Data Processors (DP)

1. Receive an instruction type from the instruction processor.
2. Receive operand labels from the instruction processor.
3. Instruct the memory hierarchy (separate from that of the instruction processor) to provide the values of the operands.
4. Receive operand values from the memory hierarchy.
5. Carry out the required operation (the execute phase).
6. Return state information to the instruction processor.
7. Provide a result value to the memory hierarchy.

© Krystian Wojtkiewicz

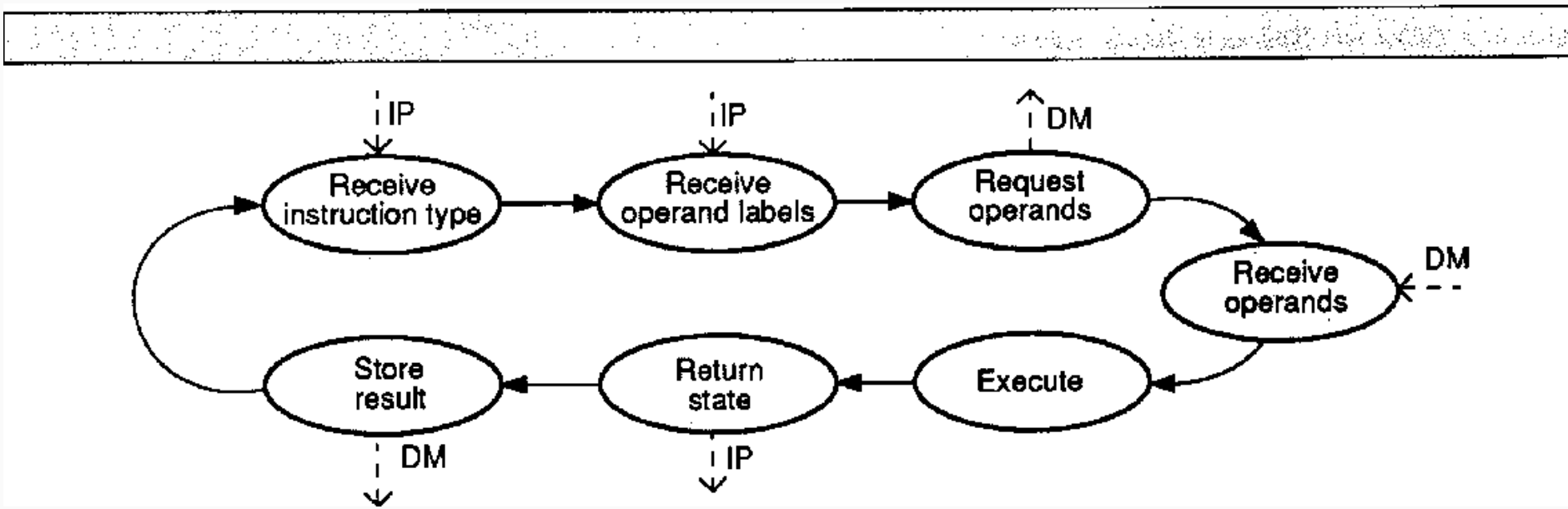


Figure: The internal structure of a von Neumann data processor [4]

Table 1. Possible architectures.

Class	IPs	DPs	IP-DP	IP-IM	DP-DM	DP-DP	Name
1	0	1	none	none	1-1	none	reduct/dataflow uniprocessor
2	0	n	none	none	n-n	none	separate machines
3	0	n	none	none	n-n	n×n	loosely coupled reduct/dataflow
4	0	n	none	none	n×n	none	tightly coupled reduct/dataflow
5	0	n	none	none	n×n	n×n	
6	1	1	1-1	1-1	1-1	none	von Neumann uniprocessor
7	1	n	1-n	1-1	n-n	none	
8	1	n	1-n	1-1	n-n	n×n	Type 1 array processor
9	1	n	1-n	1-1	n×n	none	Type 2 array processor
10	1	n	1-n	1-1	n×n	n×n	
11	n	1	1-n	n-n	1-1	none	
12	n	1	1-n	n×n	1-1	none	
13	n	n	n-n	n-n	n-n	none	separate von Neumann uniprocessors
14	n	n	n-n	n-n	n-n	n×n	loosely coupled von Neumann
15	n	n	n-n	n-n	n×n	none	tightly coupled von Neumann
16	n	n	n-n	n-n	n×n	n×n	
17	n	n	n-n	n×n	n-n	none	
18	n	n	n-n	n×n	n-n	n×n	
19	n	n	n-n	n×n	n×n	none	Denelcor Heterogeneous Element Processor
20	n	n	n-n	n×n	n×n	n×n	
21	n	n	n×n	n-n	n-n	none	
22	n	n	n×n	n-n	n-n	n×n	
23	n	n	n×n	n-n	n×n	none	
24	n	n	n×n	n-n	n×n	n×n	
25	n	n	n×n	n×n	n-n	none	
26	n	n	n×n	n×n	n-n	n×n	
27	n	n	n×n	n×n	n×n	none	
28	n	n	n×n	n×n	n×n	n×n	

Figure: Possible architectures [4]

© Krystian Wojtkiewicz

There can be around 30 different architecture variations built on top of Skillicorn taxonomy. Not all of them make sense.

Sensible architectures

- uniprocessor dataflow
- multiprocessor dataflow
- von Neumann uniprocessor (SISD)
- vector processor (SIMD)
- loosely coupled multiprocessor (MIMD)
- a tightly coupled multiprocessor (MIMD)

Coupling

- it makes sense only for data processors
- loose – only through communication channel between processors
- tight – through common access to Memory Hierarchies (practically common memory)

© Krystian Wojtkiewicz

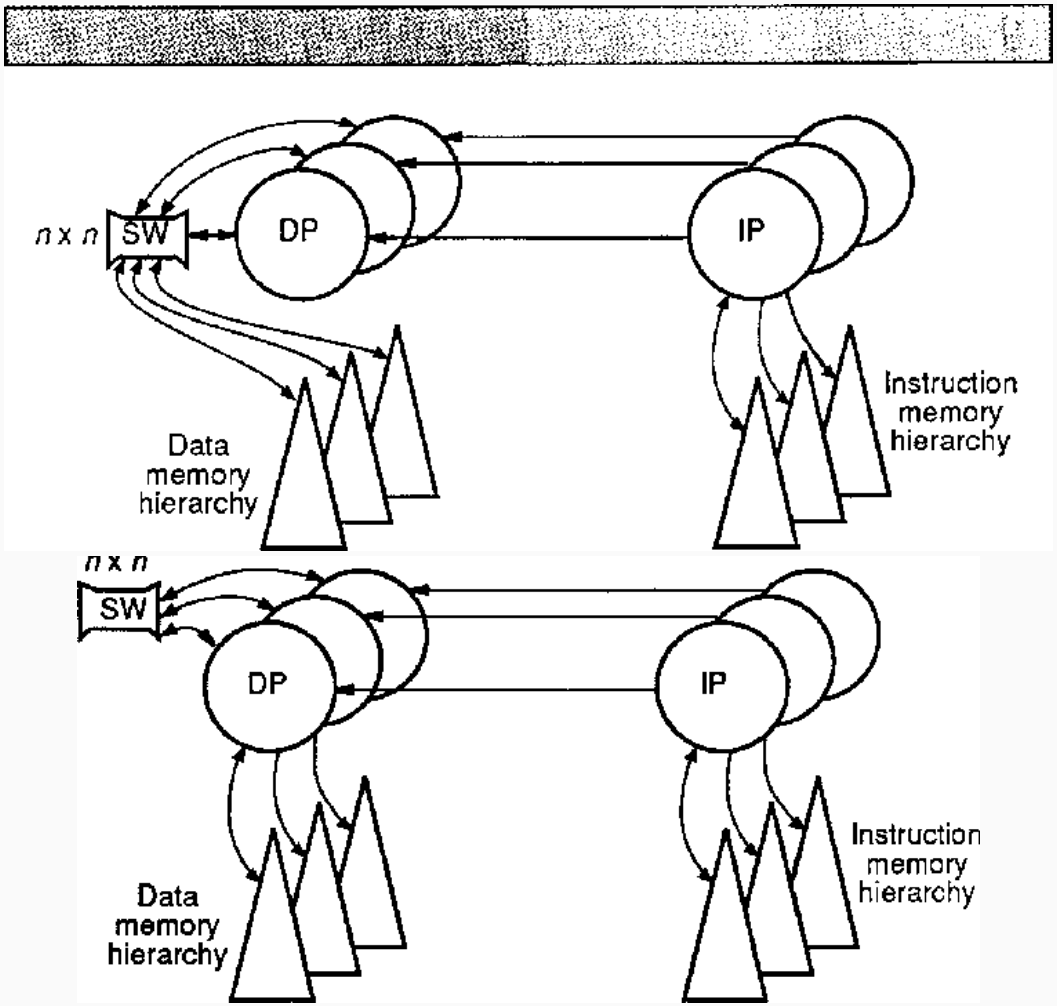


Figure: The abstract machine of a typical loosely and tightly coupled multiprocessor. [4]

Memory

General remarks

- it is impossible to build an arbitrary unlimited memory storage with infinitely short access time
- access time grows with size
- with hierarchical layer architecture every next layer has bigger size with higher access time

© Krystian Wojtkiewicz

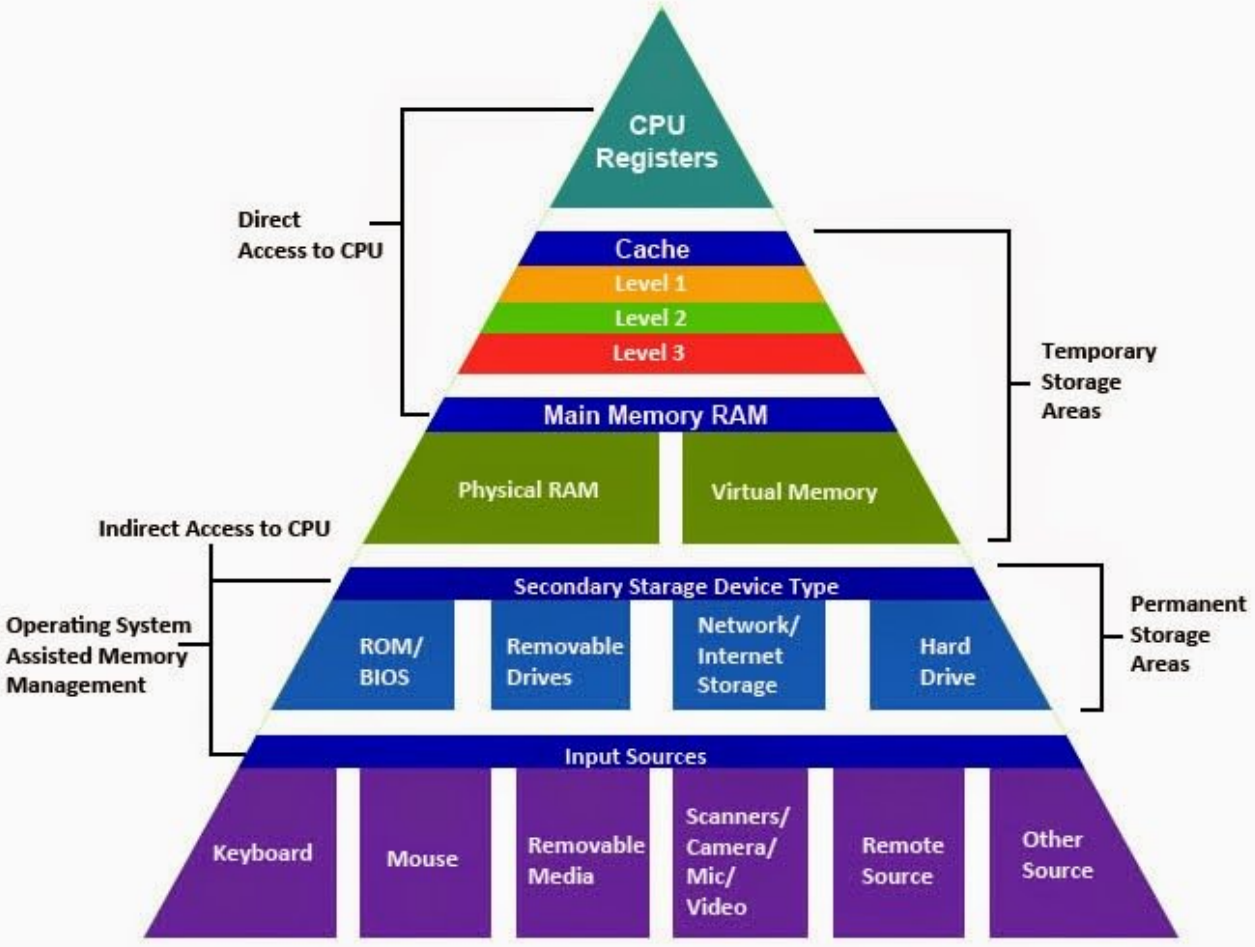


Figure: Memory Hierarchy [2]

General remarks

Memory can be generalised into five hierarchies based upon intended use and speed. If what the processor needs isn't in one level, it moves on to the next, to look for what it needs:

- volatile memory
  - registers,
  - cache,
  - main memory,
- permanent storage
  - secondary memory,
  - removable memory.

© Krystian Wojtkiewicz

The basic unit of computer storage is the **bit**. A bit can contain one of two values, 0 or 1. All other storage in a computer is based on collections of bits. Given enough bits, it is amazing how many things a computer can represent: numbers, letters, images, movies, sounds, documents, and programs.

A **byte** is 8 bits, and on most computers it is the smallest convenient chunk of storage. For example, most computers don't have an instruction to move a bit but do have one to move a byte. A less common term is word, which is a given computer architecture's native unit of data. A word is made up of *one or more* bytes. For example, a computer that has 64-bit registers and 64-bit memory addressing typically has 64-bit (8-byte) words. A computer executes many operations in its native word size rather than a byte at a time[3].

© Krystian Wojtkiewicz

Computer storage, along with most computer throughput, is generally measured and manipulated in bytes and collections of bytes.

a **kilobyte**, or KB, is 1,024 bytes

a **megabyte**, or MB, is $1{,}024^2$ bytes

a **gigabyte**, or GB, is $1{,}024^3$ bytes

a **terabyte**, or TB, is $1{,}024^4$ bytes

a **petabyte**, or PB, is $1{,}024^5$ bytes[4]

[4] Computer manufacturers often round off these numbers and say that a megabyte is 1 million bytes and a gigabyte is 1 billion bytes. Networking measurements are an exception to this general rule; they are given in bits (because networks move data a bit at a time).

© Krystian Wojtkiewicz

References

References I

[1] M. J. Flynn. “Some Computer Organizations and Their Effectiveness”. In: *IEEE Transactions on Computers* C-21.9 (Sept. 1972), pp. 948–960. ISSN: 0018-9340. DOI: 10.1109/TC.1972.5009071.

[2] Mark Lungo. *Memory Hierarchy / Useful Notes*. Dec. 2015. URL: https://tvtropes.org/pmwiki/pmwiki.php/UsefulNotes/MemoryHierarchy.

[3] A. Silberschatz, G. Gagne, and P.B. Galvin. *Operating System Concepts*. Wiley, 2018. ISBN: 9781119439257. URL: https://books.google.pl/books?id=VFV1DwAAQBAJ.

[4] David B. Skillicorn. “A Taxonomy for Computer Architectures”. In: *Computer* 21.11 (Nov. 1988), pp. 46–57. ISSN: 0018-9162. DOI: 10.1109/2.86786. URL: https://doi.org/10.1109/2.86786.

[5] Arthur Veen. “Dataflow Machine Architecture.”. In: *ACM Comput. Surv.* 18 (Dec. 1986), pp. 365–396. DOI: 10.1145/27633.28055.

© Krystian Wojtkiewicz