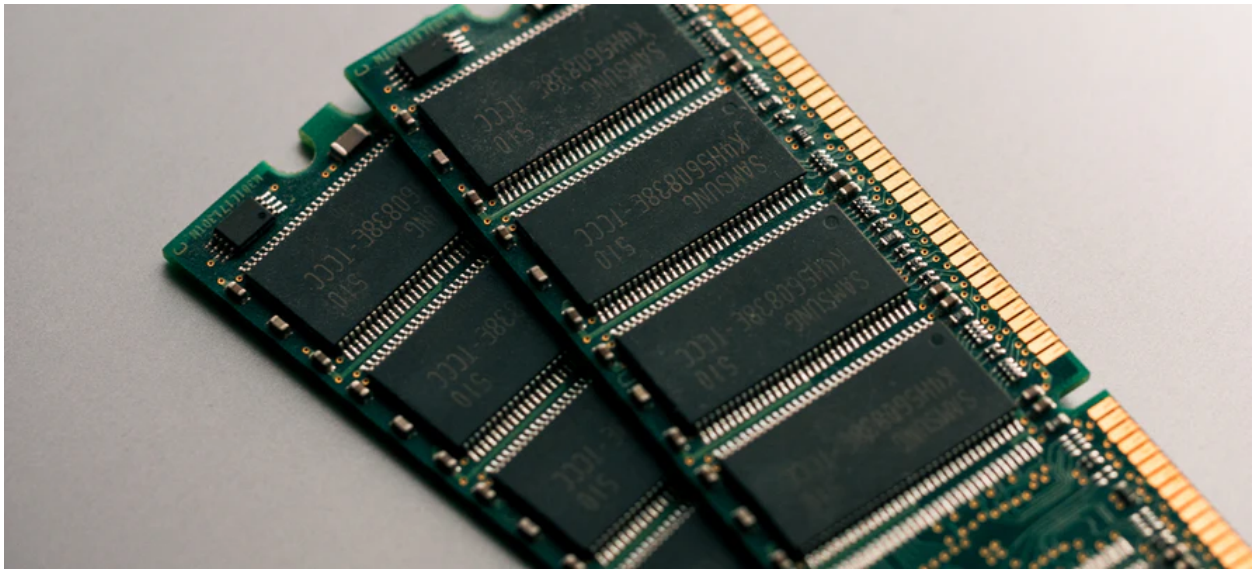
PIPELINE AND MEMORY HIERARCHY

Increasing computer speed and retrieving information from memory.



We already know how computer components perform calculations/solve problems, but there are still issues that have not been resolved. Each part of the logic circuit has a different purpose, but so far we can use it to successfully solve one problem at the time. Each circuit is like a tool used by the ALU, but we only allow it to be used by one person (program) at a time. To overcome this issue, we would like different programs to use the same part of the circuit simultaneously, but in such a way that will not cause problems. And even the best computers cannot just solve equations, they always have to save the results somewhere. Both of those issues are solved with the use of pipelining and memory allocation. And that is what will be described below.

CLOCK CYCLE AND BUS

Increasing clock cycle

If we tried to manually calculate two binary numbers it would definitely take us more than 1 second. For computers, that is an eternity. In the beginning, processors were usually made faster by improving the switching time of the transistors inside the chip, but that can speed things up only so much. Next improvement was adding specialized circuits that perform specific

operations or calculations. Instead of solving problems like division by just subtracting the numbers until we reach zero or negative (e.g 15/5 is same as 15-5-5-5) - it does the job, but it's not very efficient because it takes a lot of clock cycles, we have implemented circuits in hardware that do such operations. Result is increased speed but more complexity.

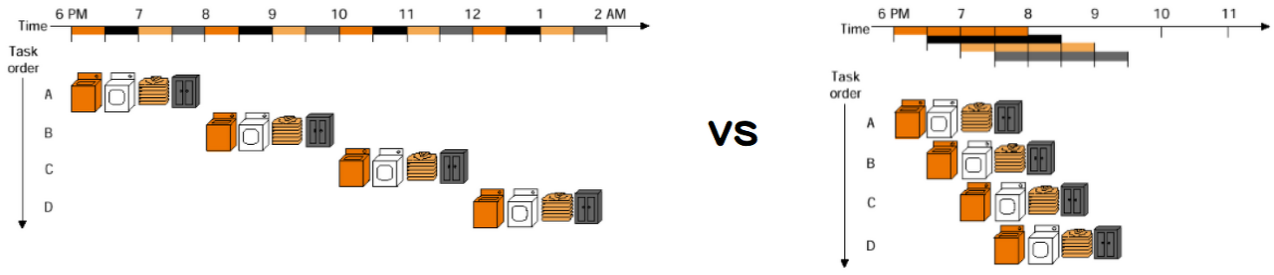
Bus

Bus is a name for connection between components inside the computer, or between computers, and it's used as a communication channel. Only one component can send data at a time, so processors usually have more than one bus connection to communicate with memory.

Modern processors operate at Gigahertz speed. But high clock speeds lead to another problem - getting data to the CPU at that speed. It's like having a powerful steam locomotive, but no way to shovel in coal fast enough. In our case, the bottleneck is the RAM. To speed things up, we have designated memory right on the CPU called cache, that temporarily stores data from the RAM, so there is no waiting time.

PIPELINE

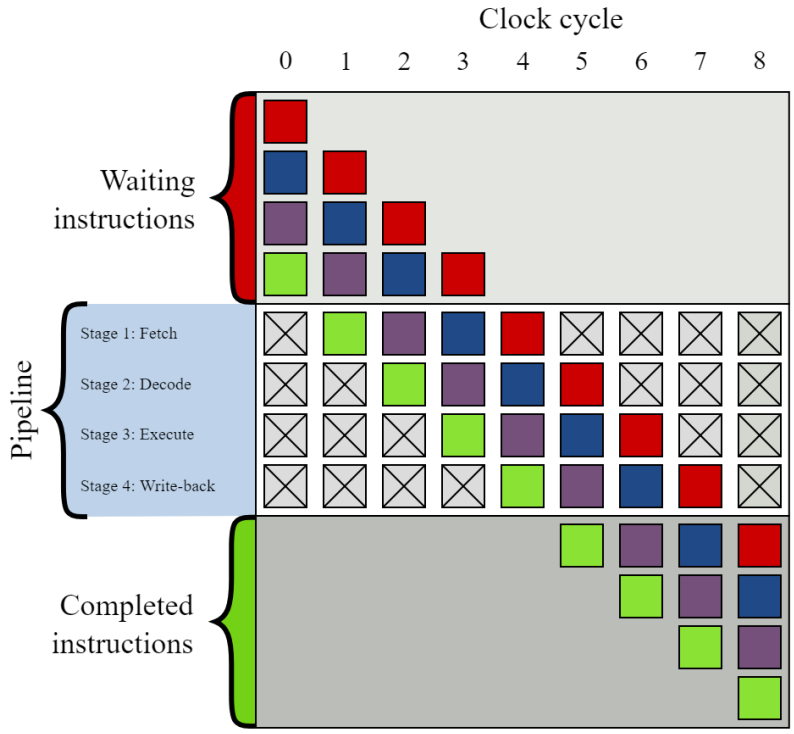
Another way to increase the speed is using instruction pipelining. If you work in a hotel and you need to wash the sheets, you wouldn't wait for them to be washed and dried before you start washing another batch, you'd probably start washing the next batch while the first one is drying.



https://it-eldorado.tistory.com/43

Computers do the same thing with pipelines, that allow them to operate multiple tasks simultaneously. That way the processor is constantly kept busy. The von Neumann cycle has

four basic operations to perform: FETCH, DECODE, EXECUTE, WRITE BACK. If we start fetching the next set of data while loading the first, we will speed things up!



https://upload.wikimedia.org/wikipedia/commons/c/cb/Pipeline%2C_4_stage.svg

Hazards

In a pipelined design few instructions are in some stage of execution. There are possibilities for some kind of dependency amongst these sets of instructions and thereby limiting the speed of the Pipeline. The dependencies occur for a few reasons which we will be discussing soon. The dependencies in the pipeline are called Hazards as these cause hazards to the execution. We use the word Dependencies and Hazard interchangeably as these are used so in Computer Architecture. Essentially an occurrence of a hazard prevents an instruction in the pipe from being executed in the designated clock cycle. We use the word clock cycle, because each of these instructions may be in a different machine cycle of theirs.

There are three kinds of hazards: structural, daza and control hazards. Here we will discuss the structural hazard. Structural hazards arise due to hardware resource conflict amongst the instructions in the pipeline. A resource here could be the Memory, a Register in GPR or ALU.

This resource conflict is said to occur when more than one instruction in the pipe is requiring access to the same resource in the same clock cycle. This is a situation where the hardware cannot handle all possible combinations in an overlapped pipelined execution [source].

TYPES OF MEMORY

Take the memory types and group them with respect to access speed. Explain why some of them are faster (due to lack of complicated circuit, small space, simplicity etc.) Then repeat with the same about the capacity. At the end, once again find purpose for each type of memory with explanation, why with such speed and capacity is it the best option for such and such function.

All data is stored within some kind of a memory unit. There are many different types of memory, differentiating in access speed and capacity, but in general we can distinguish the following:

Flip Flop

Flip flop is a hardware unit capable of storing only 1 bit of data (1 or 0). Due to the simplicity, it is also the fastest of all and the base for creating every other memory unit.

Register

Registers are internal memory of CPU, they are Incredibly fast, but very expensive. This is where current instruction data is stored, and it's approximate size is 100 B.

Cache

Cache is a small amount of fast memory, even faster than RAM, and this is where program Variables are stored. It reduces the amount of RAM read/write operations. So how does it work? Data from RAM is usually read sequentially, but with cache, instead of reading 1 address from RAM, a chunk of several hundred Bytes is read and stored in cache. So when the CPU needs information from the next address in RAM, the next reading will probably be read from cache, and that is important because it saves us time (earlier mentioned bottleneck).

A cache miss is a failed attempt to read or write a piece of data in the cache, which results in a main memory access with much longer latency. Opposite to that is cache hit, where data can be

found in cache and there is no need to retrieve data from the main memory. Most modern CPUs have up to 90% cache hit! But can we make things even faster? Yes.

The cache memory is then divided in more layers, known also as cache hierarchy, or multi-level caches. Cache closest to the processor is called level-one cache or L1, it is usually directly on the processor, and its size is approximately 10kB. If the required data is not there, the computer will look for it in L2 or L3, which are usually not on the processor and are larger in size (1-64 MB).

RAM

Random Access Memory, or it's street name - RAM, is the buffer between CPU storage and external storage. This is where the running Code is stored. After catche, it's the next faste, but volatile memory - content not stored after the device is turned off. Its size can vary from 1 to 32 GB.

HDD / SSD

Hard Disk Drive and Solid state drive are where the whole program is stored. It's a Non-Volatile memory, so the content will be saved after the device is turned off. For HHD usual size is 1 TB, and SSD usual size is between 125 GB and 1 TB.

MEMORY HIERARCHY - ACCESS SPEED AND CAPACITY

With the knowledge about types and functionality of different memory units, we can divide and sort them based on two factors. The access time, which describes the speed of performing any kind of operation on the memory stored in a specific component, and the capacity, which tells us how much data can we in fact store. Both of those parameters are unfortunately inversely proportional, as with increase of the capacity, there is much more data to search in, as well as different layers of both security and functions have to be passed.
You can save an entire movie on your flash drive, as well as copy files, change their properties (names, extensions) and much much more, whereas a flip-flop latch has only one function- store one bit value of 1 or 0.

Due to that dependency, it is important to know what are our needs of both speed and capacity before designing an electronic device, so that we will be able not only to save money, but also increase time and space efficiency of our creation.

The hierarchical order for both factors, and addition information about prices is nicely described by the following graph: