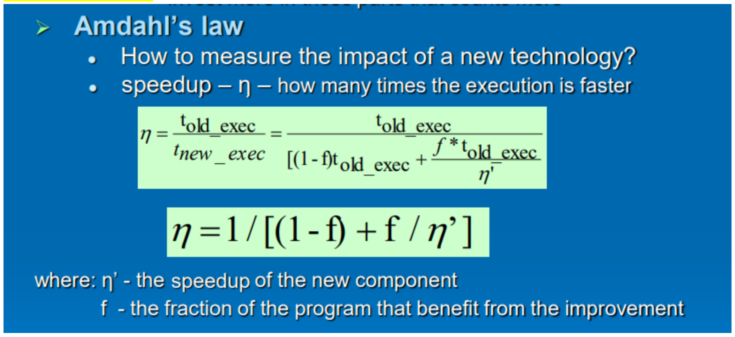
1. Calculate the speedup of a processor if the speed of floating point instruction that happen 20% of the time is 2 times faster.



- Amdahl's law
  - How to measure the impact of a new technology?
  - speedup – η – how many times the execution is faster

$$\eta = \frac{t_{old\_exec}}{t_{new\_exec}} = \frac{t_{old\_exec}}{[(1-f)t_{old\_exec} + \frac{f * t_{old\_exec}}{\eta'}}$$

$$\eta = 1 / [(1-f) + f / \eta']$$

where: η' - the speedup of the new component

f - the fraction of the program that benefit from the improvement

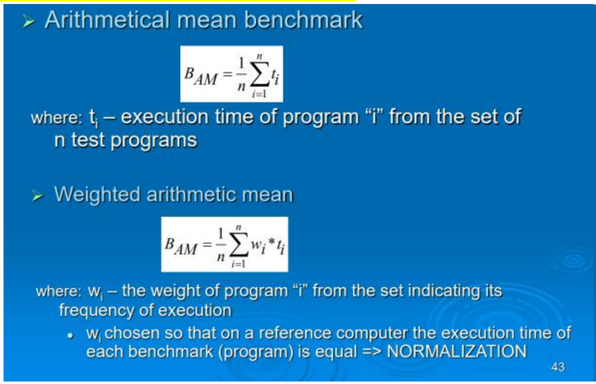
f = 20%, n' = 2

n = 1/((1 - 0.2) + 0.2/2) = 1/(0.8 + 0.1) = 1\0.9 = 1.11111111…

2. What is temporal locality and when is it useful?

If a location is accessed at a given time it has a high probability of being accessed in the near future. Examples: exaction of loops (for, while, etc.), repeated processing of some variables

3. Ex cu normalizare benchmark



- Arithmetical mean benchmark

$$B_{AM} = \frac{1}{n}\sum_{i=1}^{n} t_i$$

where: t_i – execution time of program "i" from the set of n test programs

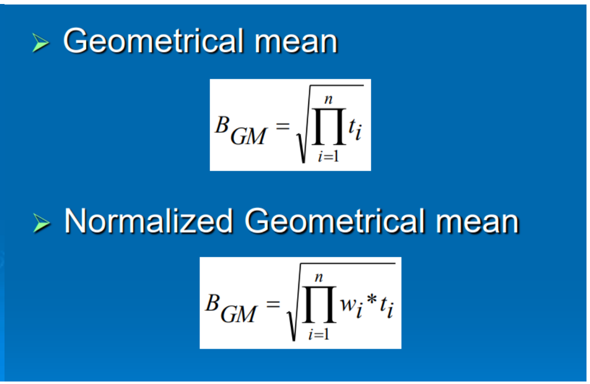
- Weighted arithmetic mean

$$B_{AM} = \frac{1}{n}\sum_{i=1}^{n} w_i * t_i$$

where: w_i – the weight of program "i" from the set indicating its frequency of execution

- w_i chosen so that on a reference computer the execution time of each benchmark (program) is equal => NORMALIZATION

43

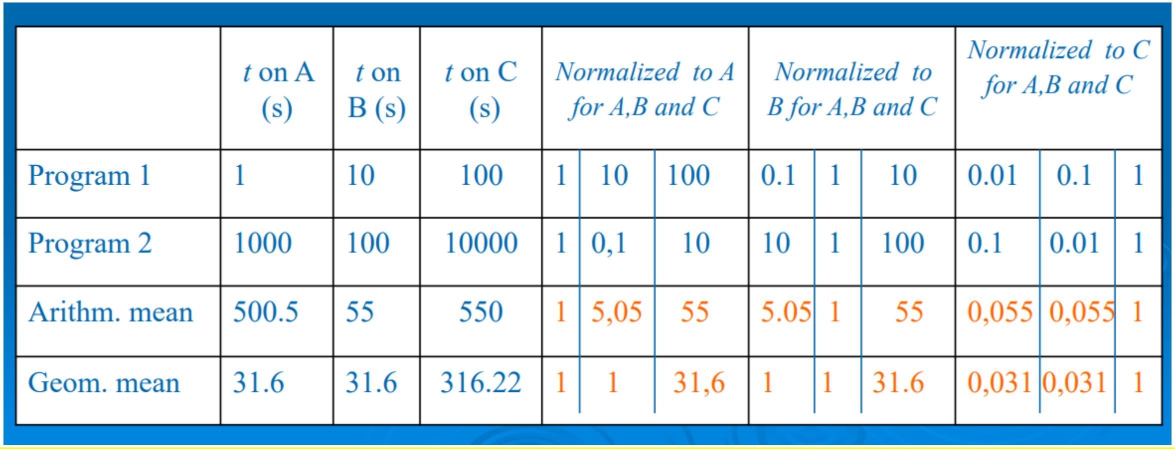


- Geometrical mean

$$B_{GM} = \sqrt[n]{\prod_{i=1}^{n} t_i}$$

- Normalized Geometrical mean

$$B_{GM} = \sqrt[n]{\prod_{i=1}^{n} w_i * t_i}$$



	t on A (s)	*t* on B (s)	*t* on C (s)	*Normalized to A for A,B and C*			*Normalized to B for A,B and C*			*Normalized to C for A,B and C*		
Program 1	1	10	100	1	10	100	0.1	1	10	0.01	0.1	1
Program 2	1000	100	10000	1	0,1	10	10	1	100	0.1	0.01	1
Arithm. mean	500.5	55	550	1	5,05	55	5.05	1	55	0,055	0,055	1
Geom. mean	31.6	31.6	316.22	1	1	31,6	1	1	31.6	0,031	0,031	1

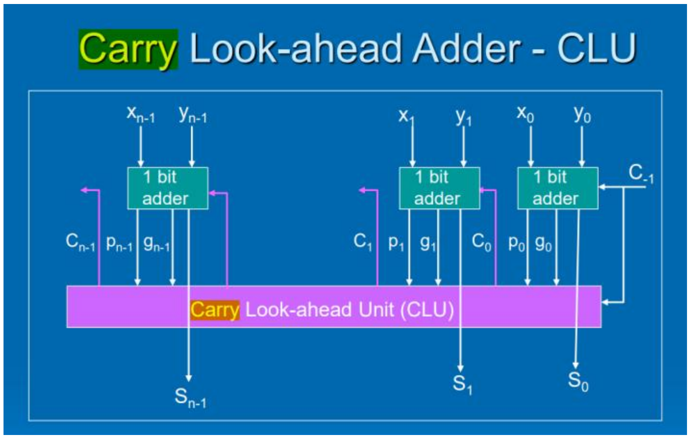
4. Calculate the maximum capacity/size of a processor if the bus has: x address bits, y data bits, z control bits (nu mi aduc aminte valorile)

This is most likely the problem to calculate the capacity of a memory. It's solved and explained later.

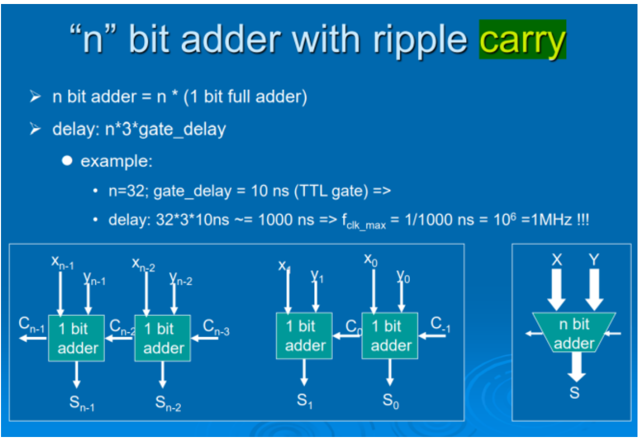
5. What factors influence the processor's clock frequency and why are newer processors limited in this aspect?

The clock frequency depends on: the integration technology – the dimension of a transistor and path lengths, supply voltage and relative distance between high and low states. To increase clock frequency the complexity of the CPU must be reduced: technological improvement – smaller transistors, through better lithographic methods, architectural improvement – simpler CPU, shorter signal paths

6. a. Scheme for sequential Carry adder și Carry Look ahead



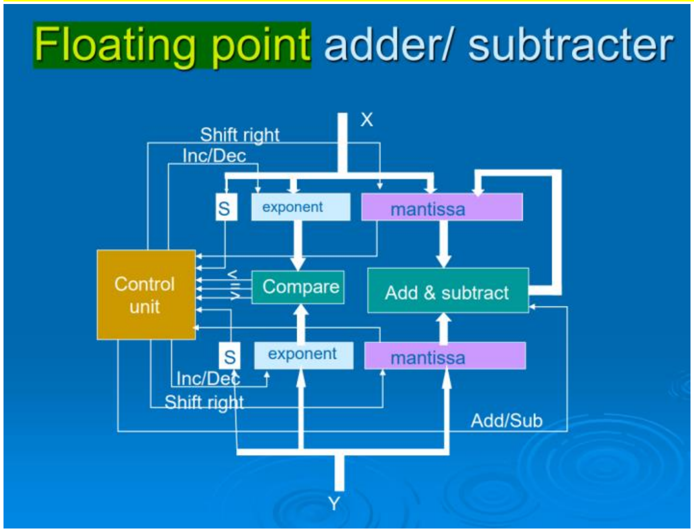
Carry Look-ahead Adder - CLU



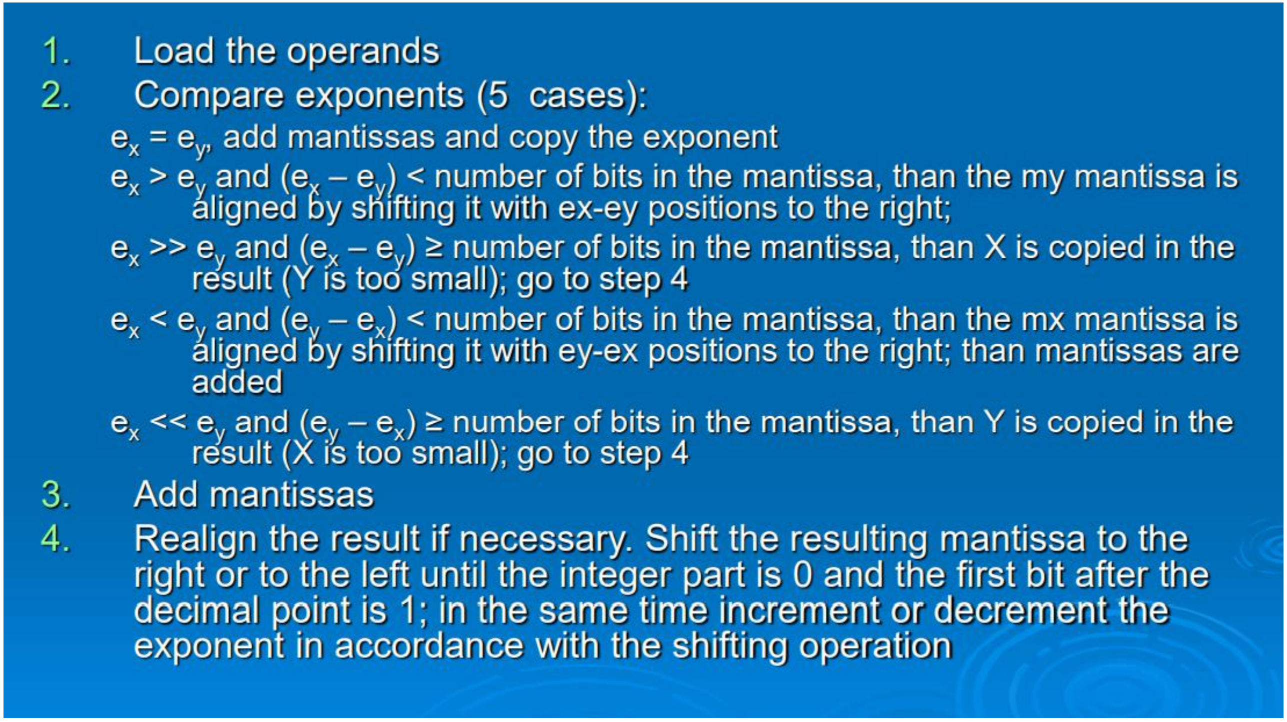
"n" bit adder with ripple carry

- n bit adder = n * (1 bit full adder)
- delay: n*3*gate_delay
  - example:
    - n=32; gate_delay = 10 ns (TTL gate) =>
    - delay: 32*3*10ns ~= 1000 ns => f_{clk_max} = 1/1000 ns = 10^6 =1MHz !!!

b. Floating point addition and subtraction (algorithm + scheme)



Floating point adder/ subtracter



1. Load the operands
2. Compare exponents (5 cases):

   $e_x = e_y$, add mantissas and copy the exponent

   $e_x > e_y$ and $(e_x - e_y) <$ number of bits in the mantissa, than the my mantissa is aligned by shifting it with ex-ey positions to the right;

   $e_x >> e_y$ and $(e_x - e_y) \geq$ number of bits in the mantissa, than X is copied in the result (Y is too small); go to step 4

   $e_x < e_y$ and $(e_y - e_x) <$ number of bits in the mantissa, than the mx mantissa is aligned by shifting it with ey-ex positions to the right; than mantissas are added

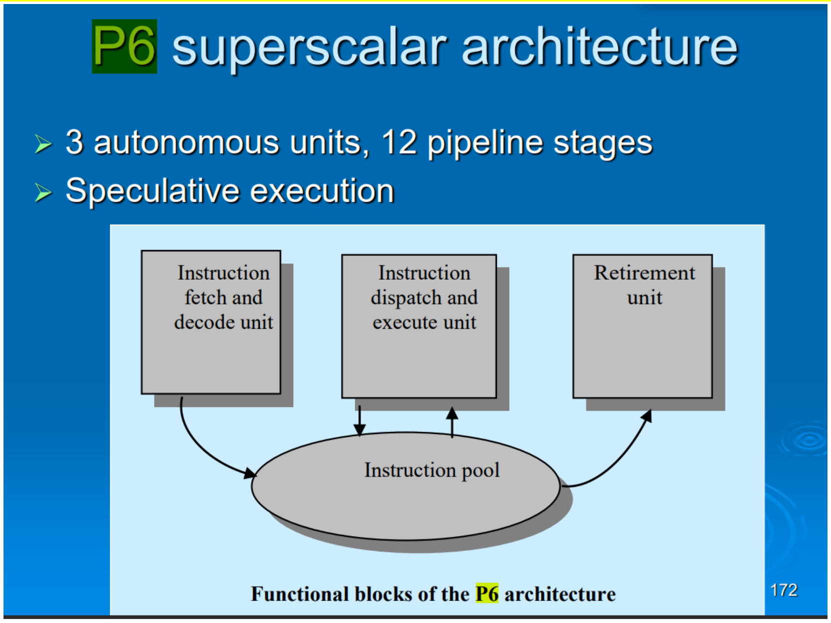
   $e_x << e_y$ and $(e_y - e_x) \geq$ number of bits in the mantissa, than Y is copied in the result (X is too small); go to step 4
3. Add mantissas
4. Realign the result if necessary. Shift the resulting mantissa to the right or to the left until the integer part is 0 and the first bit after the decimal point is 1; in the same time increment or decrement the exponent in accordance with the shifting operation

c. What is arithmetic with saturation?

If there is an overflow or underflow after an adding or subtraction the result should be the maximum or the minimum possible value. In saturated arithmetic, when a computation result exceeds the maximum value that can be represented by a variable, the result is "saturated" or clipped to the maximum representable value. Similarly, when the result of a computation falls below the minimum value that can be represented by a variable, the result is saturated or clipped to the minimum representable value.

7. Superscalar P6 architecture.

   a. Describe the architecture, show the differences between a simple pipeline, draw scheme



P6 superscalar architecture

- 3 autonomous units, 12 pipeline stages
- Speculative execution

Instruction fetch and decode unit

Instruction dispatch and execute unit

Retirement unit

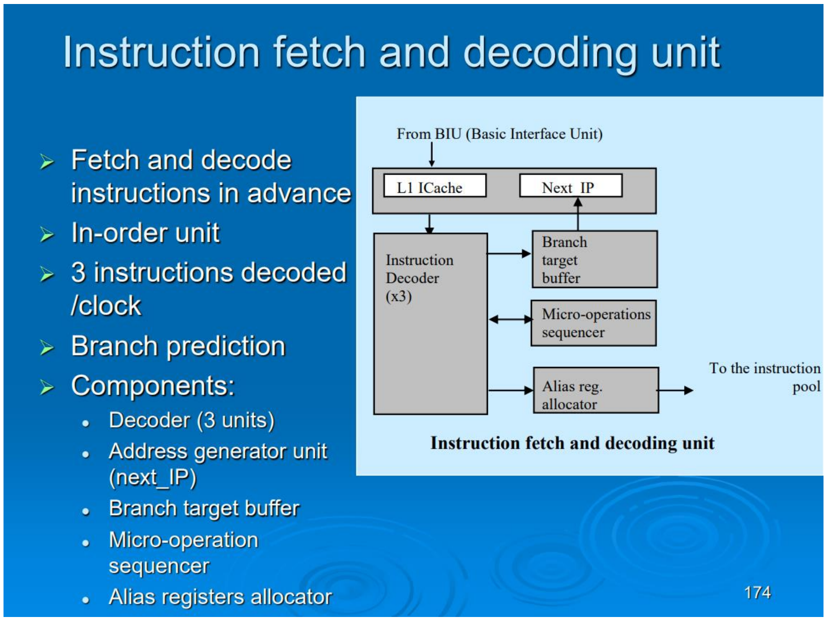
Instruction pool

Functional blocks of the P6 architecture

172

The P6 architecture is a microarchitecture used by Intel for its Pentium Pro processor. The P6 architecture is a superscalar architecture, which means it can execute multiple instructions in parallel.

   b. Describe the 3 autonomous units



Instruction fetch and decoding unit

- Fetch and decode instructions in advance
- In-order unit
- 3 instructions decoded /clock
- Branch prediction
- Components:
  - Decoder (3 units)
  - Address generator unit (next_IP)
  - Branch target buffer
  - Micro-operation sequencer
  - Alias registers allocator

From BIU (Basic Interface Unit)

L1 ICache

Next_IP

Instruction Decoder (x3)

Branch target buffer

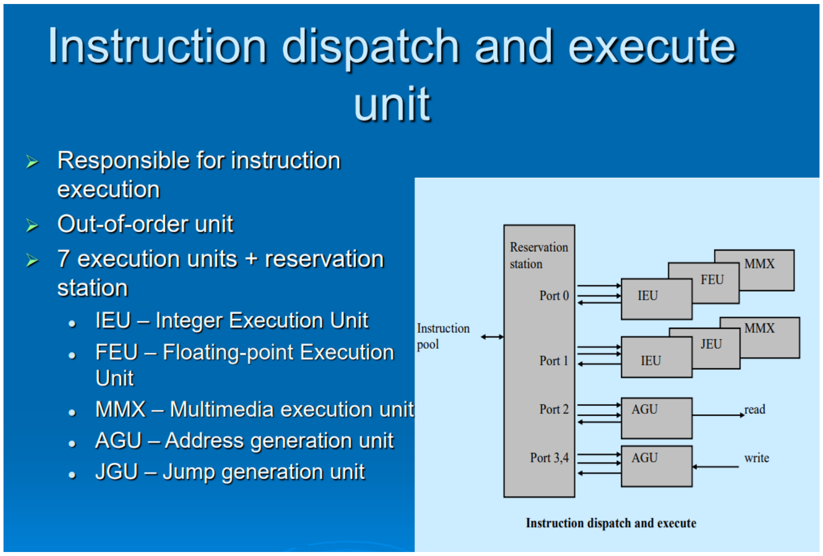
Micro-operations sequencer

Alias reg. allocator

To the instruction pool

Instruction fetch and decoding unit

174

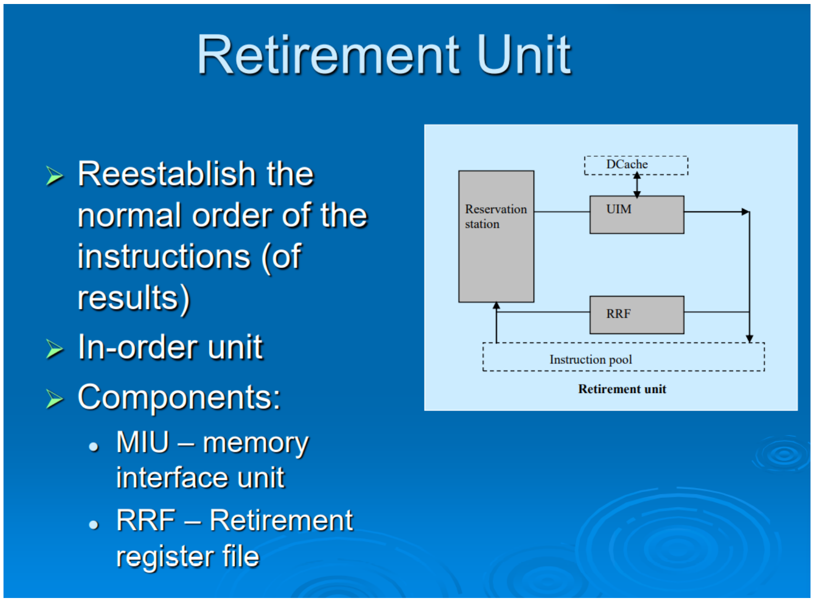


Instruction dispatch and execute unit

- Responsible for instruction execution
- Out-of-order unit
- 7 execution units + reservation station
  - IEU – Integer Execution Unit
  - FEU – Floating-point Execution Unit
  - MMX – Multimedia execution unit
  - AGU – Address generation unit
  - JGU – Jump generation unit

Instruction pool
Reservation station
Port 0
Port 1
Port 2
Port 3,4
IEU
FEU
MMX
IEU
JEU
MMX
AGU
read
AGU
write

Instruction dispatch and execute



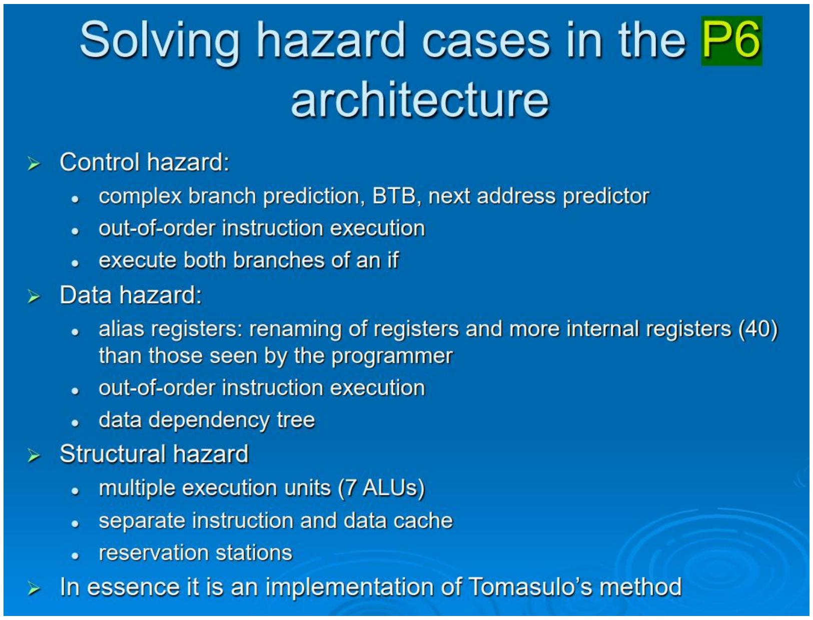
Retirement Unit

- Reestablish the normal order of the instructions (of results)
- In-order unit
- Components:
  - MIU – memory interface unit
  - RRF – Retirement register file

DCache
Reservation station
UIM
RRF
Instruction pool

Retirement unit

c. Explain how the P6 architecture solves hazards



Solving hazard cases in the P6 architecture

- Control hazard:
  - complex branch prediction, BTB, next address predictor
  - out-of-order instruction execution
  - execute both branches of an if
- Data hazard:
  - alias registers: renaming of registers and more internal registers (40) than those seen by the programmer
  - out-of-order instruction execution
  - data dependency tree
- Structural hazard
  - multiple execution units (7 ALUs)
  - separate instruction and data cache
  - reservation stations
- In essence it is an implementation of Tomasulo’s method

Dynamic Instruction Scheduling: The P6 architecture uses dynamic instruction scheduling to allow instructions to be executed out of order when it is safe to do so. This technique uses a hardware unit called the reservation station to track dependencies between instructions and execute them as soon as their input data is available.

Register Renaming: The P6 architecture uses register renaming to eliminate data dependencies between instructions. This technique allows multiple instructions to use the same register, even if they have conflicting data dependencies.

Speculative Execution: The P6 architecture uses speculative execution to predict the outcome of conditional branches and execute instructions before the outcome is known. This technique reduces the performance penalty of mispredicted branches.

Out-of-Order Execution: The P6 architecture allows instructions to be executed out of order, as long as the results of the instructions are still in the correct order. This technique allows the processor to take advantage of available execution resources and maximize instruction-level parallelism.

Branch Prediction: The P6 architecture uses a branch predictor to predict the outcome of conditional branches. The branch predictor uses information about the program's past behavior to make accurate predictions and reduce the performance penalty of mispredicted branches.

8. Cache memories

   a. What are cache memories and what is their role?

      It’s a high-speed low capacity memory between the CPU and the internal memory, keeps copies of the main memory’s zones (lines). The role of cache memory is to provide faster access to frequently used data and instructions, reducing the time it takes for the CPU to access data from main memory.

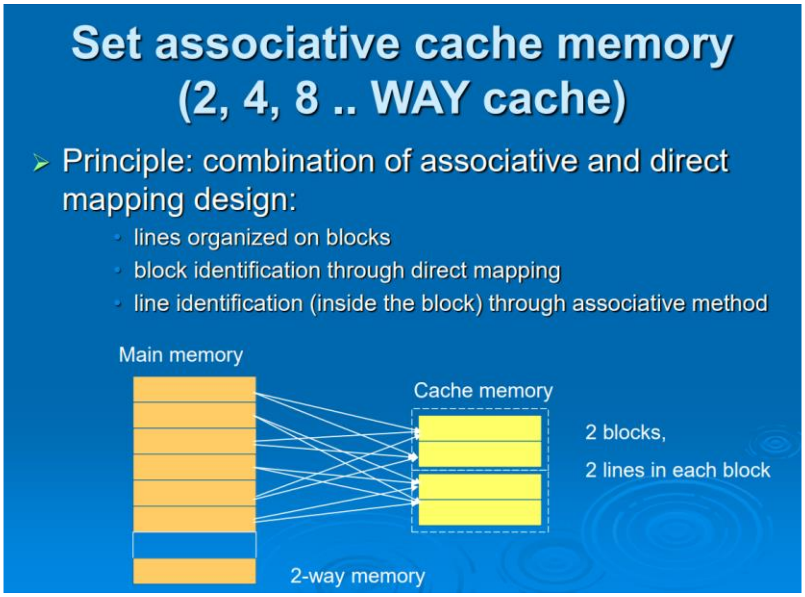
   b. Enabling conditions

      ? spacial + temporal locality

      The enabling conditions for cache memory are: the processor should have a cache controller that manages the transfer of data between the cache and main memory, and the cache should have sufficient capacity to store frequently accessed data and instructions.

   c. Explain direct mapping and disadvantages for each solution set associative (with examples and schemes)

      Direct mapping is when the address of the line in the cache memory is determined directly from the location’s physical address.

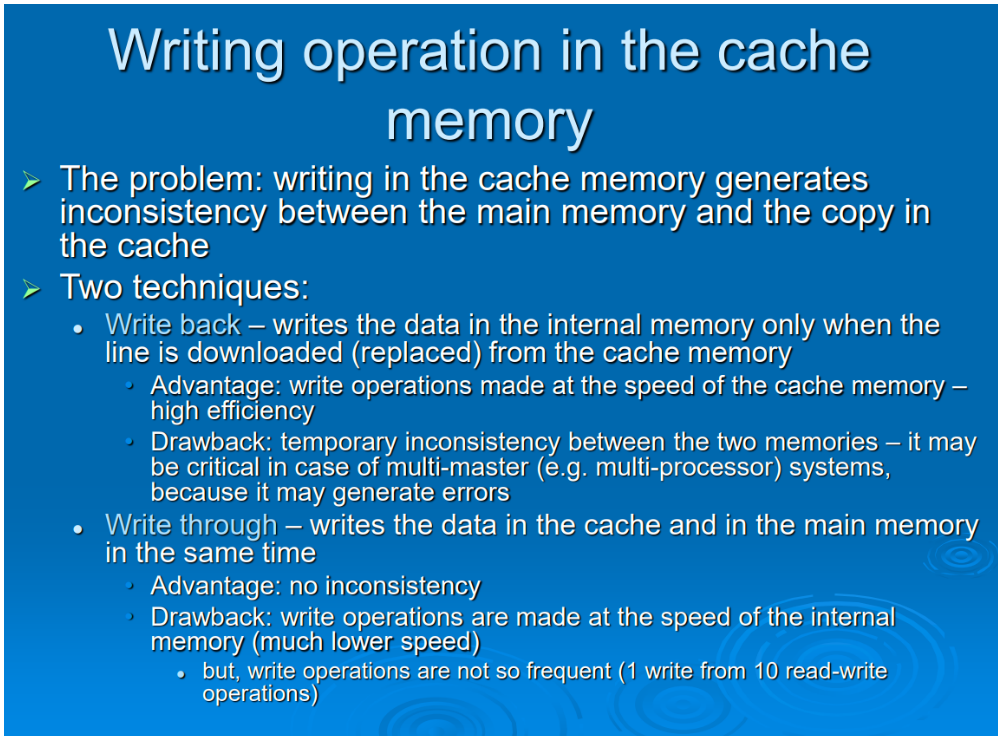


Set associative cache memory (2, 4, 8 .. WAY cache)

- Principle: combination of associative and direct mapping design:
  - lines organized on blocks
  - block identification through direct mapping
  - line identification (inside the block) through associative method

The drawback is that the implementation is more complex.

d. How to solve writing issues problems, what are the advantages and

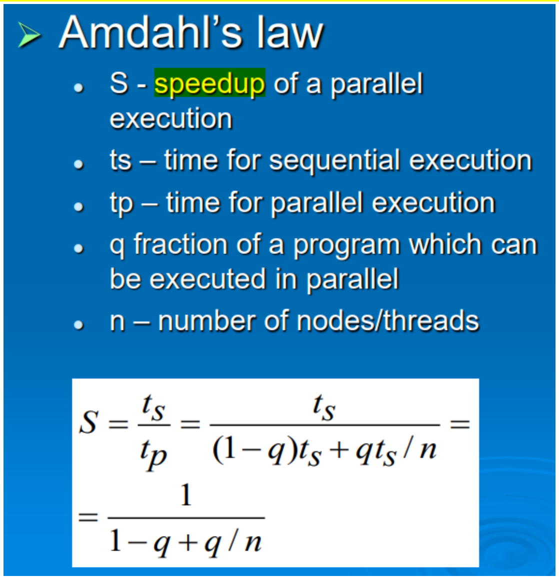


Writing operation in the cache memory

- The problem: writing in the cache memory generates inconsistency between the main memory and the copy in the cache
- Two techniques:
  - Write back – writes the data in the internal memory only when the line is downloaded (replaced) from the cache memory
    - Advantage: write operations made at the speed of the cache memory – high efficiency
    - Drawback: temporary inconsistency between the two memories – it may be critical in case of multi-master (e.g. multi-processor) systems, because it may generate errors
  - Write through – writes the data in the cache and in the main memory in the same time
    - Advantage: no inconsistency
    - Drawback: write operations are made at the speed of the internal memory (much lower speed)
      - but, write operations are not so frequent (1 write from 10 read-write operations)

---

1. Speedup of a processor with 4 cores and 80% parallel execution



- Amdahl's law
  - S - speedup of a parallel execution
  - ts – time for sequential execution
  - tp – time for parallel execution
  - q fraction of a program which can be executed in parallel
  - n – number of nodes/threads

$$S = \frac{t_S}{t_P} = \frac{t_S}{(1-q)t_S + q t_S / n} = \\ = \frac{1}{1-q+q/n}$$

q = 80%, n = 4

1/(1-0.8 + 0.8/4) = 1/(0.2 + 0.2) = 1/0.4 = 2.5

2. Tabel de normalization cu 3 programe si 2 procesoare de unde rezulta ca procesoru a ii de 2 ori mai incet ca B

already solved

3. Ceva dubios care nu am mai vazut in niciun subiect

4. Moor's law and explain if it's still true today

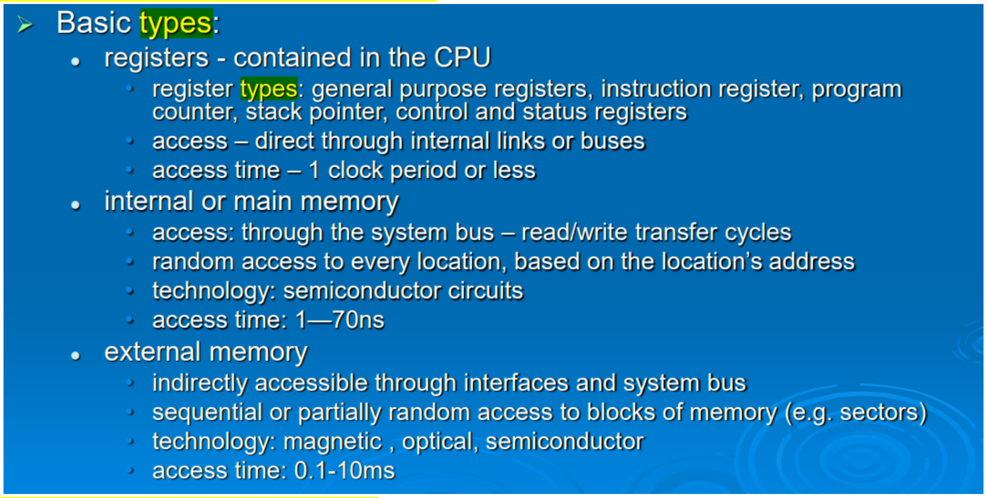
Moore's Law is an observation made by Intel co-founder Gordon Moore in 1965, which states that the number of transistors that can be placed on a microchip doubles approximately every two years, while the cost per transistor decreases. This observation has held true for several decades and has been a driving force behind the rapid growth and advancement of the semiconductor industry. However, as transistor sizes approach physical limits, it has become increasingly difficult and costly to continue to scale down transistor sizes while maintaining performance improvements. This has led some experts to predict that Moore's Law may slow down or come to an end in the near future. Nonetheless, many researchers and engineers are still pushing the limits of semiconductor technology, and new approaches such as quantum computing and neuromorphic computing may continue to drive progress in the field of computing.

5. List 5 characteristics of a HPC
    - high speed: 1-20.000 Tflops
    - high memory capacity: 1-700 TBytes
    - parallelism: 1024 – 1500000 cores for CPUs
    - high speed connections: InfiniBand, Ethernet for fast data transfer and communication
    - high power consumption: 10KW- 10MW
    - very expensive
    - used for scientific computing, simulation, cryptography
6. How is division performed

a),b),c),d)

find in course I guess

7. Memories
    a. types of memories(3 characteristics)

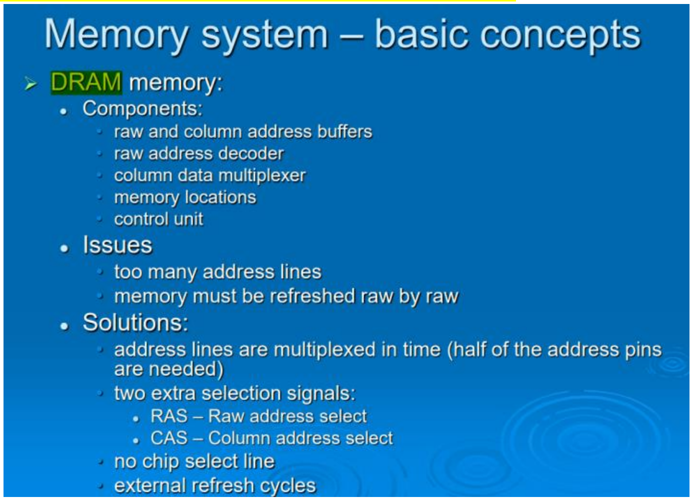


- Basic types:
    - registers - contained in the CPU
        - register types: general purpose registers, instruction register, program counter, stack pointer, control and status registers
        - access – direct through internal links or buses
        - access time – 1 clock period or less
    - internal or main memory
        - access: through the system bus – read/write transfer cycles
        - random access to every location, based on the location's address
        - technology: semiconductor circuits
        - access time: 1—70ns
    - external memory
        - indirectly accessible through interfaces and system bus
        - sequential or partially random access to blocks of memory (e.g. sectors)
        - technology: magnetic , optical, semiconductor
        - access time: 0.1-10ms

    b. ceva despre memorii in general

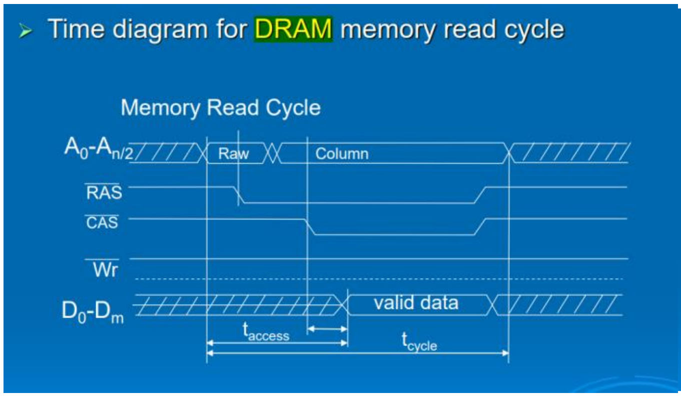
    Memory is an electronic component used in computers and other digital devices to store and retrieve information quickly. There are different types of memory used in computers, including Random Access Memory (RAM), Read-Only Memory (ROM), and non-volatile memory such as flash memory and hard disk drives. The primary purpose of memory is to provide fast and efficient access to data and instructions needed by the processor. The capacity of memory varies depending on the type and purpose of the memory.

c. DRAM(diagram of the read operation, etc.)

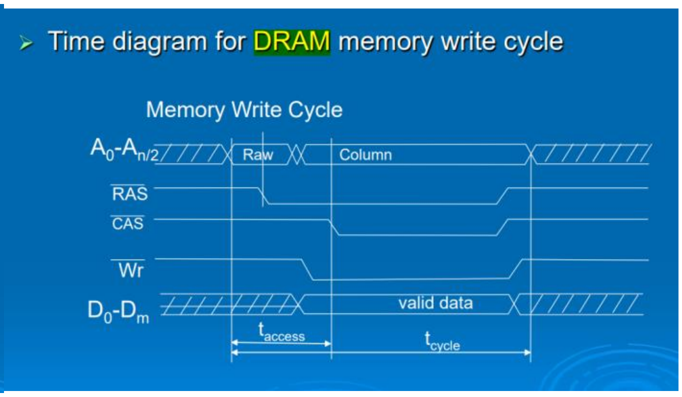


Memory system – basic concepts

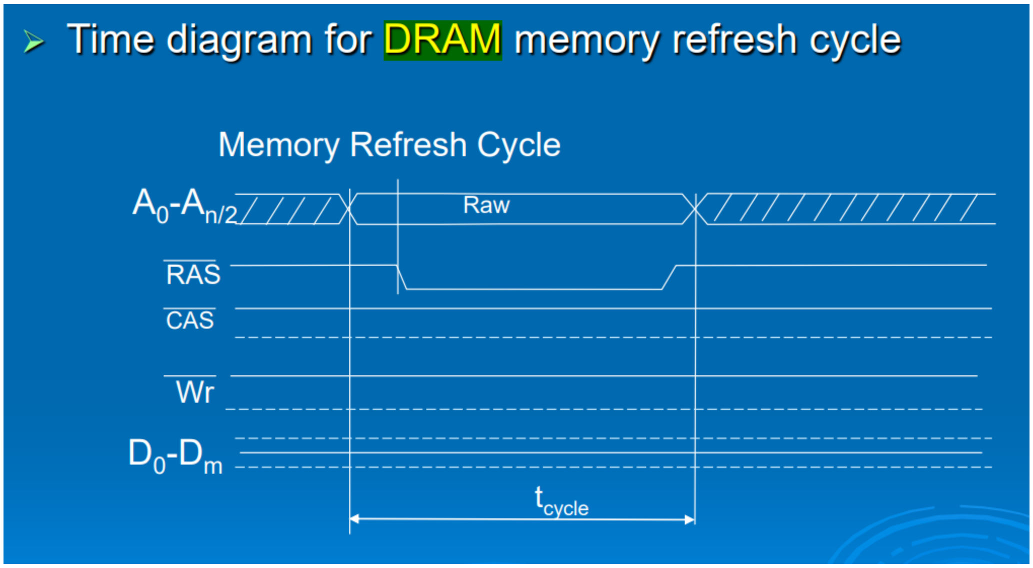
- DRAM memory:
  - Components:
    - raw and column address buffers
    - raw address decoder
    - column data multiplexer
    - memory locations
    - control unit
  - Issues
    - too many address lines
    - memory must be refreshed raw by raw
  - Solutions:
    - address lines are multiplexed in time (half of the address pins are needed)
    - two extra selection signals:
      - RAS – Raw address select
      - CAS – Column address select
    - no chip select line
    - external refresh cycles



- Time diagram for DRAM memory read cycle



- Time diagram for DRAM memory write cycle



- Time diagram for DRAM memory refresh cycle

d. how to speedup DRAM

smaller access time = higher speed (curs)

Here are some ways to speed up DRAM:

Increase Clock Speed: One of the easiest ways to increase the speed of DRAM is to increase its clock speed. This can be done by adjusting the frequency of the memory bus or by overclocking the RAM.

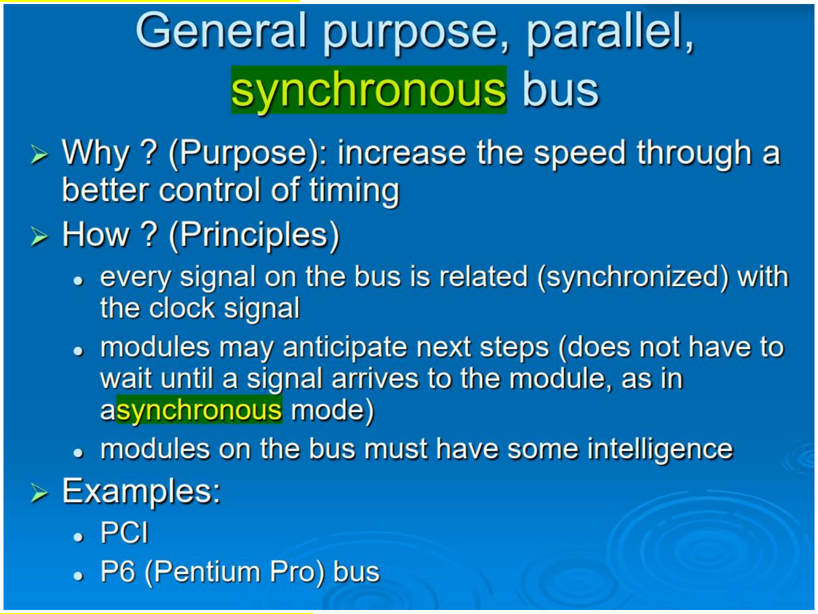
Reduce Latency: Another way to improve DRAM performance is to reduce the latency of the memory. This can be done by adjusting the timings of the memory or by using faster memory modules.

Enable XMP: XMP (Extreme Memory Profile) is a feature that can be enabled in the BIOS of the computer, which automatically optimizes the settings of the DRAM for improved performance.

Improve Cooling: DRAM can become hot during intensive operations, which can negatively impact its performance. By improving cooling, such as using a heatsink or a liquid cooling system, the temperature of the DRAM can be reduced, allowing it to operate more efficiently.

Upgrade Hardware: Upgrading the CPU, motherboard, or other hardware components in the computer can also improve DRAM performance by increasing the overall system speed and bandwidth.

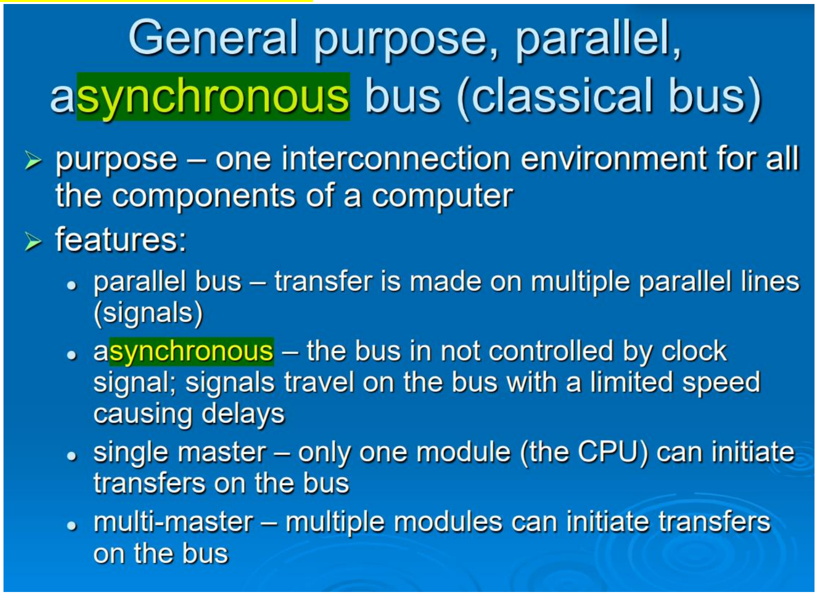
8. Interconnected systems
    a. purpose of interconnected systems
       Connect different components of a computer system (CPU, memory, interfaces, peripheral devices) – buses; or interconnect multiple computer systems – networks. Purpose: exchange of data and instruction codes, synchronization and coordination of actions, events signaling
    b. synchronous parallel bus



General purpose, parallel, synchronous bus

- Why ? (Purpose): increase the speed through a better control of timing
- How ? (Principles)
  - every signal on the bus is related (synchronized) with the clock signal
  - modules may anticipate next steps (does not have to wait until a signal arrives to the module, as in asynchronous mode)
  - modules on the bus must have some intelligence
- Examples:
  - PCI
  - P6 (Pentium Pro) bus

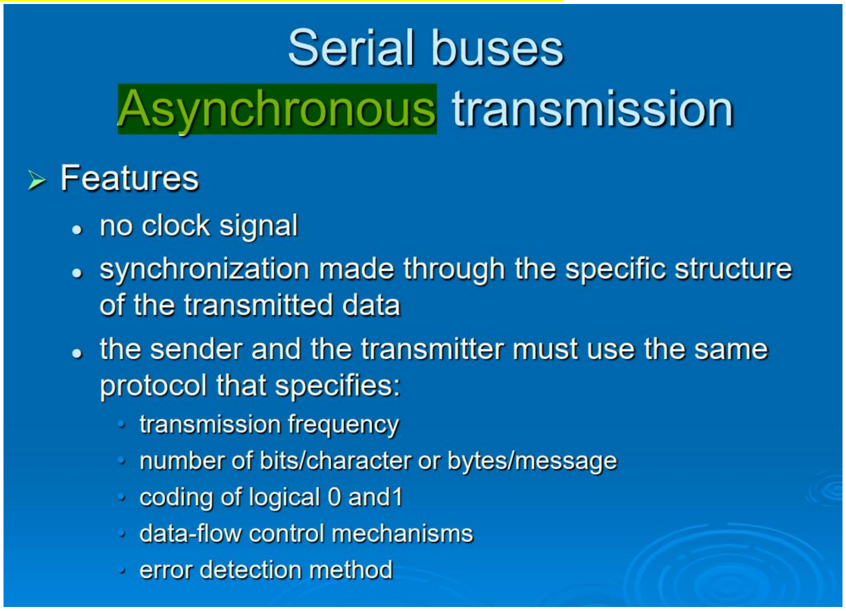
    c. asynchronous parallel bus



General purpose, parallel, asynchronous bus (classical bus)

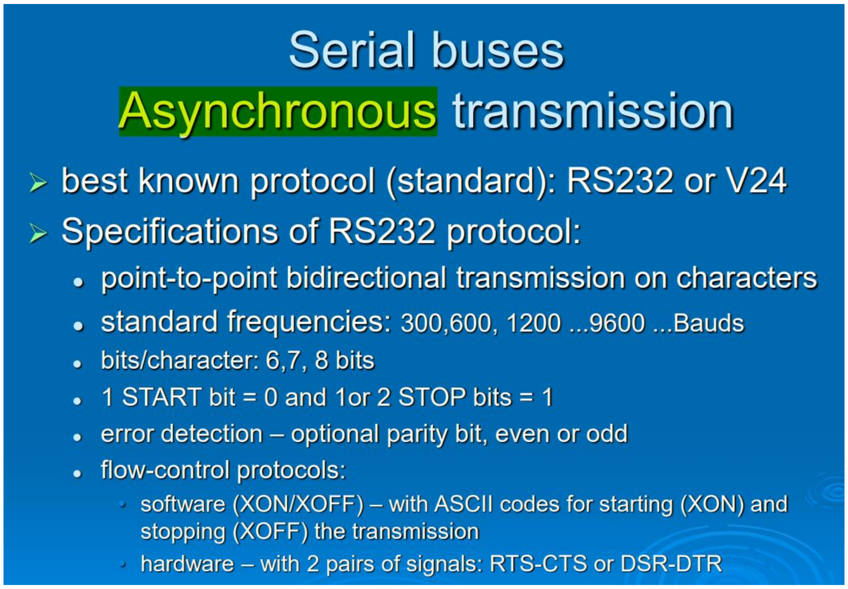
- purpose – one interconnection environment for all the components of a computer
- features:
  - parallel bus – transfer is made on multiple parallel lines (signals)
  - asynchronous – the bus in not controlled by clock signal; signals travel on the bus with a limited speed causing delays
  - single master – only one module (the CPU) can initiate transfers on the bus
  - multi-master – multiple modules can initiate transfers on the bus

d. asynchronous serial bus(RS424 sau cv de genu)



Serial buses
Asynchronous transmission

- Features
  - no clock signal
  - synchronization made through the specific structure of the transmitted data
  - the sender and the transmitter must use the same protocol that specifies:
    - transmission frequency
    - number of bits/character or bytes/message
    - coding of logical 0 and1
    - data-flow control mechanisms
    - error detection method



Serial buses
Asynchronous transmission

- best known protocol (standard): RS232 or V24
- Specifications of RS232 protocol:
  - point-to-point bidirectional transmission on characters
  - standard frequencies: 300,600, 1200 ...9600 ...Bauds
  - bits/character: 6,7, 8 bits
  - 1 START bit = 0 and 1or 2 STOP bits = 1
  - error detection – optional parity bit, even or odd
  - flow-control protocols:
    - software (XON/XOFF) – with ASCII codes for starting (XON) and stopping (XOFF) the transmission
    - hardware – with 2 pairs of signals: RTS-CTS or DSR-DTR

---

1. de aplicat adahmel (adahmel?)

$$\frac{1}{(1-q)+q/n}$$

2. Hyper threading (inclusiv procentul ala cu cat se imbunatateste timpul de executie (30%) se cerea sa fie mentionat)
   - parallel execution of instruction streams on a single CPU
   - when a tread is stalled because of some hazard cases another thread can be executed
   - solution: two threads executed in parallel on the same pipelined CPU, after every stage two buffers (registers) store the partial results of the two threads
   - 30% speedup
   - OS will detect 2 logical CPUs

3. ceva chestie dubioasa de imi dadea frecventa 2gHZ CPI=5, capacitatea memoriei 4gb si mai zicea ceva de 64 biti si se cerea nr de IPS (instr per second)

$$IPS = \frac{1}{CPI * T_{clk}} = \frac{f_{clk}}{CPI}$$

4. de aplicat media aritmetica si ce reprezinta rezultatul obtinut.
5. imi cerea capacitatea memoriei si aveam cate linii de adresa, de date si de control am.

To calculate the capacity of a memory when given the number of address, data, and control lines, you need to first determine the number of addressable locations, which is equal to 2^n where n is the number of address lines.

Next, you need to determine the number of bits that can be stored per addressable location, which is equal to the number of data lines.

Once you have determined the number of addressable locations and the number of bits that can be stored per addressable location, you can calculate the total capacity of the memory as follows:

Capacity = Number of Addressable Locations x Number of Bits per Addressable Location

For example, if a memory has 16 address lines and 8 data lines, the capacity would be:

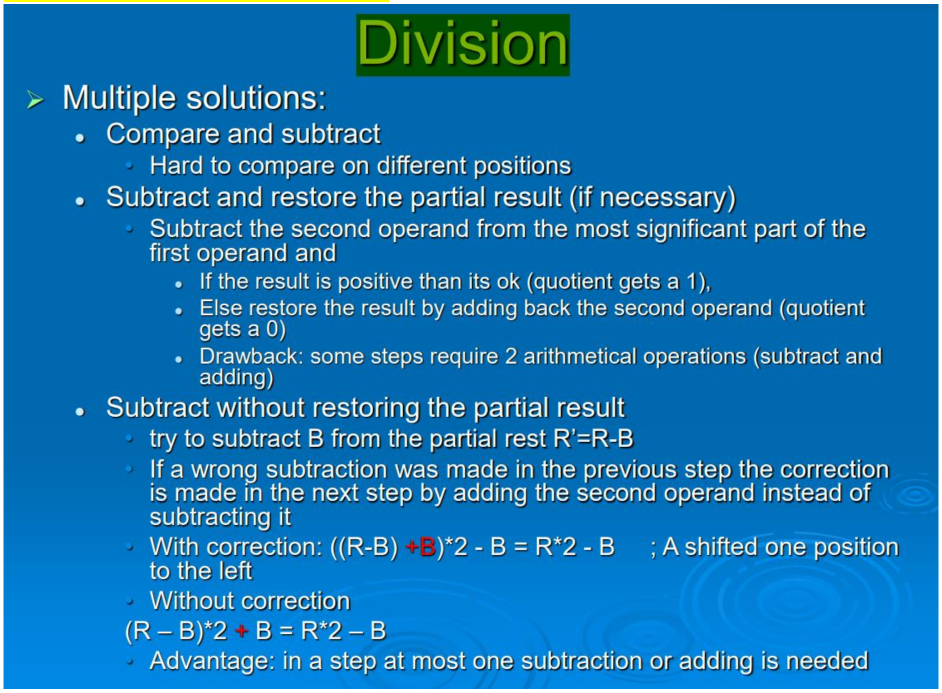
Number of Addressable Locations = 2^16 = 65,536

Number of Bits per Addressable Location = 8

Capacity = 65,536 x 8 = 524,288 bits

This is equivalent to 64 kilobytes (KB) of memory.

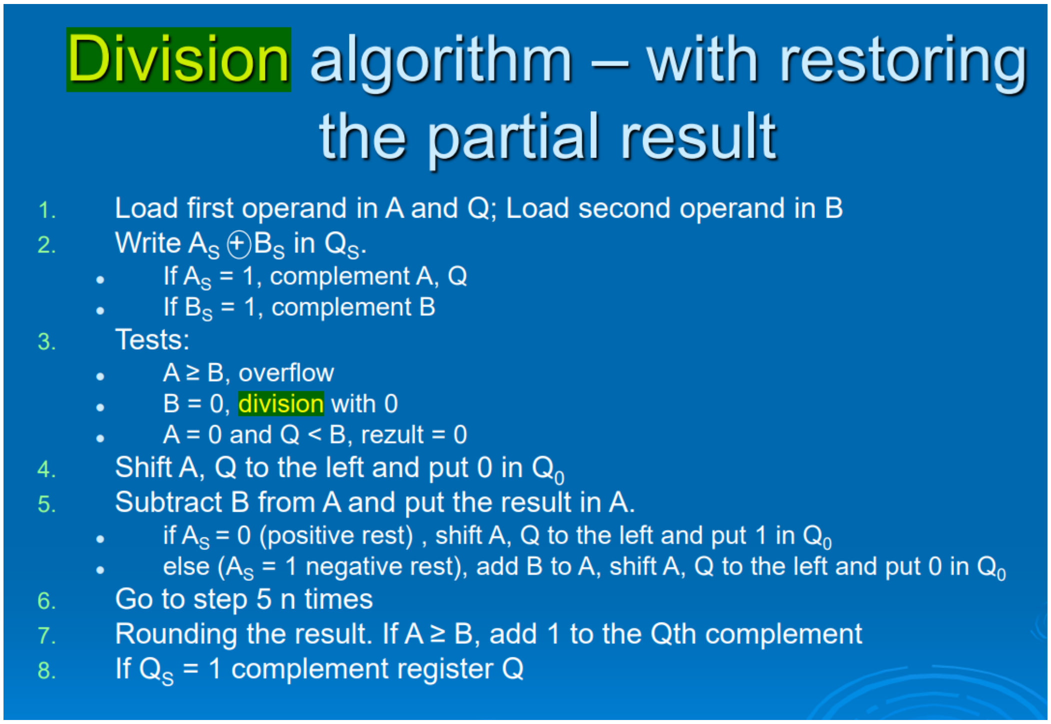
6. Impartirea intregi, tipuri si pasii pt aia cu with partial result restored + impartirea la floating point (+ schema aia din curs pt impartirea la intregi).



Division

- Multiple solutions:
  - Compare and subtract
    - Hard to compare on different positions
  - Subtract and restore the partial result (if necessary)
    - Subtract the second operand from the most significant part of the first operand and
      - If the result is positive than its ok (quotient gets a 1),
      - Else restore the result by adding back the second operand (quotient gets a 0)
      - Drawback: some steps require 2 arithmetical operations (subtract and adding)
  - Subtract without restoring the partial result
    - try to subtract B from the partial rest R'=R-B
    - If a wrong subtraction was made in the previous step the correction is made in the next step by adding the second operand instead of subtracting it
    - With correction: ((R-B) +B)*2 - B = R*2 - B ; A shifted one position to the left
    - Without correction

    (R – B)*2 + B = R*2 – B
    - Advantage: in a step at most one subtraction or adding is needed



Division algorithm – with restoring the partial result

1. Load first operand in A and Q; Load second operand in B
2. Write $A_S \oplus B_S$ in Q_S.
   - If $A_S = 1$, complement A, Q
   - If $B_S = 1$, complement B
3. Tests:
   - $A \geq B$, overflow
   - B = 0, division with 0
   - A = 0 and Q < B, rezult = 0
4. Shift A, Q to the left and put 0 in Q_0
5. Subtract B from A and put the result in A.
   - if $A_S = 0$ (positive rest) , shift A, Q to the left and put 1 in Q_0
   - else ($A_S = 1$ negative rest), add B to A, shift A, Q to the left and put 0 in Q_0
6. Go to step 5 n times
7. Rounding the result. If $A \geq B$, add 1 to the Qth complement
8. If $Q_S = 1$ complement register Q

7. Microprocesoare (def + semnalele de la bus + ex de procesoare de-a lungul istoriei + arhitectura P06 ce aduce nou)

A microprocessor is an integrated (VLSI) circuit that integrates a CPU (or more CPUs).

- address signals: specify memory locations, generated by the microprocessor, determine the maximum addressing space
- data signals: transfer instruction codes and data, determine width of data on the bus
- command signals: determine memory and interface read and write cycles
- control signals: help controlling the address and data amplifiers
- PSU signals: GND, Vcc

I8086, I8088, I80286, I80386, I80486, Pentium and variations

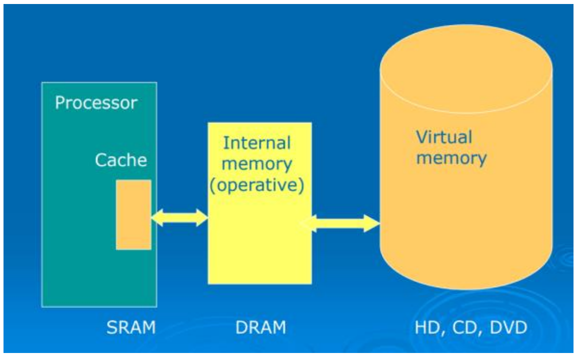
P6 architecture already described afaik

8. Ierarhia memoriei (la ce ajuta + principii care sustin asta (spatial locality, temporal locality, 90/10) + segmentation (gdt, ldt, ...) + pagination)

Hierarchy of “memoirs” = virtual, internal, cache

- we want: big capacity, high speed, affordable price
- we have: high speed, low cap (SRAM, ROM), medium speed, big cap (DRAM), low speed, high cap (HDD)

To achieve all 3 requirements we combine technologies in a hierarchical way.



- temporal locality: if a location is accessed at a given time it has a high probability of being accessed in the near future
- special locality: if a location is accessed than its neighbors have a high probability of being accessed in the near future
- 90/10: 90% of the time the processor executes 10% of the program

Segmentation: divide and protect memory zones from unauthorized access.

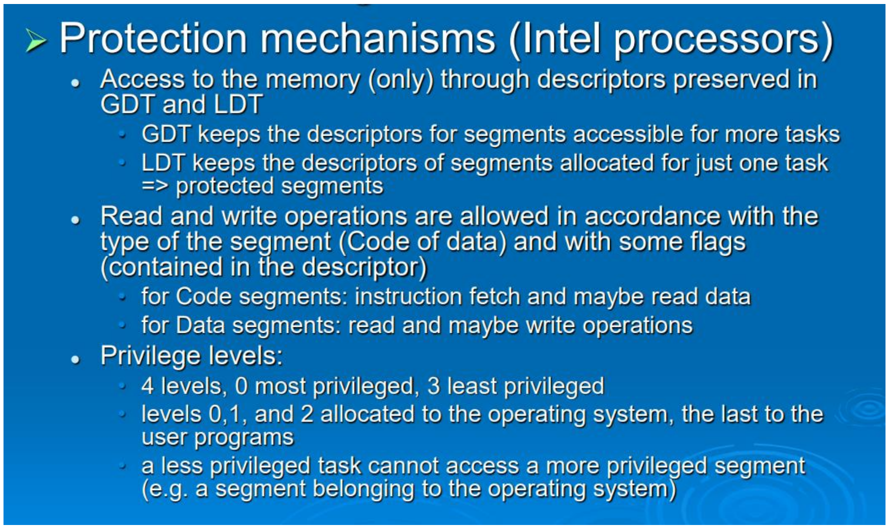
- divide into blocks (segments) – fixed/variable length, with/without overlapping
- address locations Physical_address = Segment_address + Offset_address
- attach attributes to segments in order to control the operations allowed in the segment and describe its content

Advantages: access is limited to a segment, memory zones may be separated, segments can be placed in different zones

Disadvantage: complex access mechanisms, longer access time

“pagination” Paging: increase the internal memory over the external one

- Internal and external memory is divided into blocks (pages) of fixed length
- bring into the internal memories only those pages that have a high probability of being used in the near future (90/10)
- associative implementation



➢ Protection mechanisms (Intel processors)

- Access to the memory (only) through descriptors preserved in GDT and LDT
  - GDT keeps the descriptors for segments accessible for more tasks
  - LDT keeps the descriptors of segments allocated for just one task => protected segments
- Read and write operations are allowed in accordance with the type of the segment (Code of data) and with some flags (contained in the descriptor)
  - for Code segments: instruction fetch and maybe read data
  - for Data segments: read and maybe write operations
- Privilege levels:
  - 4 levels, 0 most privileged, 3 least privileged
  - levels 0,1, and 2 allocated to the operating system, the last to the user programs
  - a less privileged task cannot access a more privileged segment (e.g. a segment belonging to the operating system)

In paging, memory is divided into fixed-size pages, and each page can be assigned a protection attribute such as read-only, read-write, or execute-only. This allows an operating system to control access to memory on a per-page basis, providing protection against unauthorized access or modification.

cred ca am facut subiectul asta deja + nu mai am chef sa il fac + L + ratio + restanta + media prea mica pentru bursa

1. Compute the speedup of a program if the computer has 4 processors and 80% of the program can be executed in parallel.
2. Moore's law and arguments if it is still valid today
3. Compute the weighted (normalized) average mean of the following benchmark with processor B as a reference and draw the conclusions from the computations.
   - Program 1 20s 10s
   - Program 2 30s 15s
   - Program 3 60s 30s
4. List 5 features that characterize the high performance computers
5. Define arithmetics with saturation and give examples (up and down).
6. Division
   - methods of division for integers and floating point
   - describe the algorithm of division with partial results
   - scheme for division algorithm
   - principle of floating point division
7. Memory
   - general characteristics of a memory
   - ways of classification (at least 3 criteria)
   - DRAM - principles, specifications, special requirements
   - diagram for a read operation in DRAM
   - methods of speedup for a DRAM (new technologies)
8. Interconnection Systems
   - why and where are they used?
   - describe a general purpose parallel asynchronous bus - principle, signals, advantages, drawbacks
   - describe a general purpose parallel synchronous bus - principle, signals, advantages, drawbacks
   - describe a serial bus