FINAL EXAM PREPARATION

EXCERCISES

CHAP 1:

What is abstraction in computer?

How to evaluate computer performance?

Answer:

- Abstraction:
    - Abstraction in computer is a technique for hardware and software to improve productivity.
    - It represents the design at different levels of representation.
    - Lower-level details are hidden to offer simpler model at higher levels.
- Performance:

$$performance = \frac{1}{CPU\ time}$$

$$CPU\ time = \#CPU\ clock\ cycles * clock\ cycle\ time = \frac{\#CPU\ clock\ cycles}{clock\ rate}$$

$$\#CPU\ clock\ cycles = \#instructions * CPI$$

Overall:

$$CPU\ time = \#instructions * CPI * clock\ cycle\ time = \frac{\#instructions * CPI}{clock\ rate}$$

CHAP 4:

Show the simple single cycle CPU design and explain how each type of instruction is executed (R-type, I-type, conditional branch).

How pipelining can improve CPU performance? What are the problems that may affect the real performance of CPU pipeline?

CPU design:

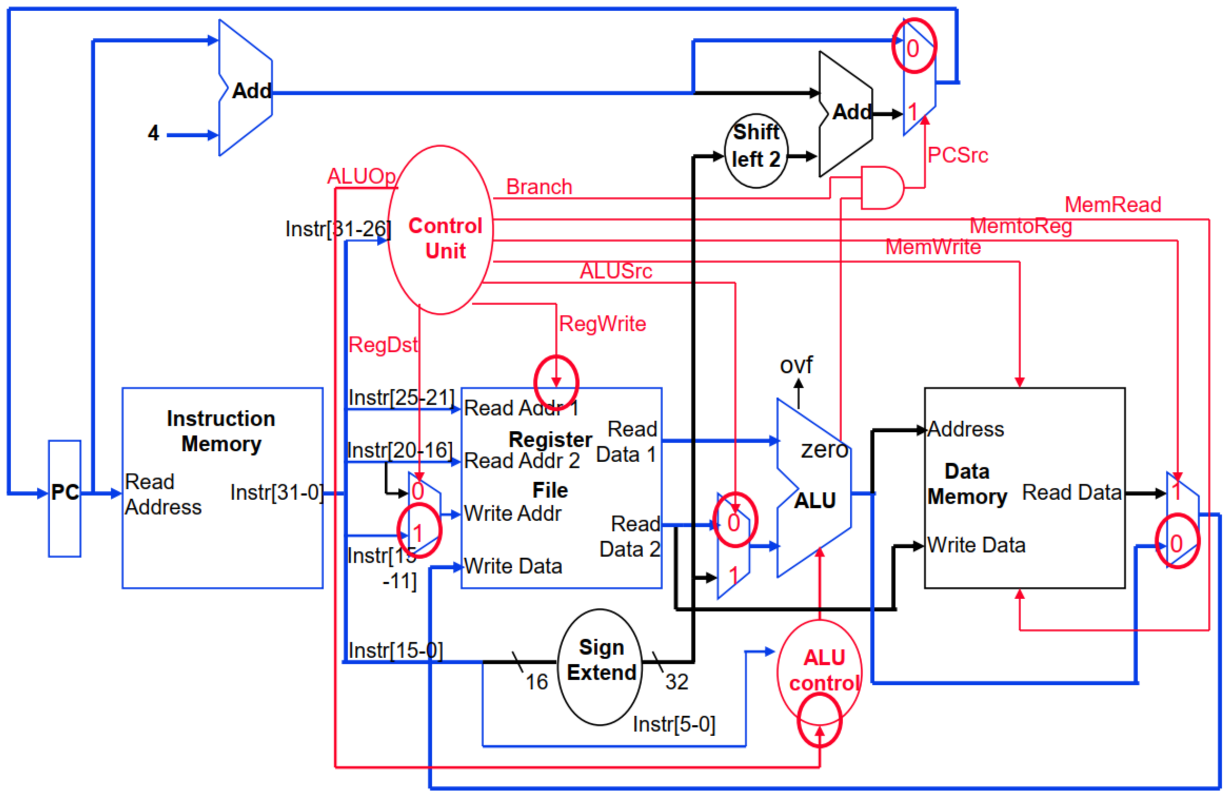


Figure 1. R-type (add)

- With the address pointed at by the PC:
  - The instruction is fetched the Instruction Memory.
  - PC is updated at the same time (PC = PC + 4).
- The instruction is decoded:
  - Instr[31-26]: Opcode is sent to the Control Unit, which generates all the red color signal to control others components. Control Unit also generates ALU control signal to the ALU.
  - Instr[25-21]: Register rs is sent to read port 1.
  - Instr[20-16]: Register rt is sent to read port 2 and multiplexor.
  - Instr[15-11]: Register rd is sent to multiplexor, RegDst == 1 → write address.
- In the Register File:
  - Read data 1, 2 is read from read address 1, 2
  - Forward read data 1 to the ALU.
  - Forward read data 2 to the multiplexor, ALUSrc == 0 → ALU.
- At the ALU:
  - Generate output, sent to multiplexor, MemtoReg == 0 → Register Fille.
- Finally:
  - RegWrite signal → write data to write address.

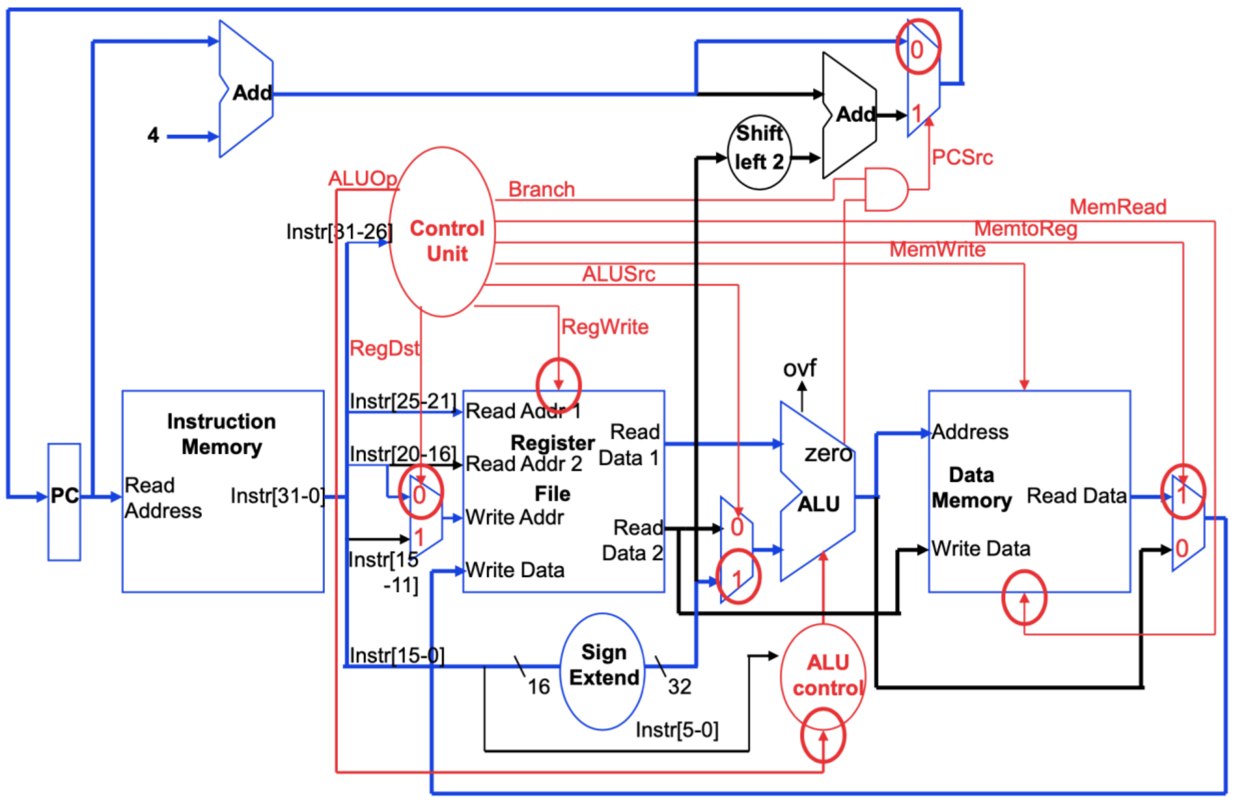


Figure 2. I-type (lw)

- With the address pointed at by the PC:
  - The instruction is fetched the Instruction Memory.
  - PC is updated at the same time (PC = PC + 4).
- The instruction is decoded:
  - Instr[31-26]: Opcode is sent to the Control Unit, which generates all the red color signal to control others components. Control Unit also generates ALU control signal to the ALU.
  - Instr[25-21]: Register rt is sent to read port 1.
  - Instr[20-16]: Register rs is sent to read port 2 and multiplexor, RegDst == 0 → write address.
  - Instr[15-0]: Send to sign extend to 32-bit, ALUSrc == 1 → ALU.
- In the register file:
  - Forward data read from read port 1 to ALU.
- At the ALU:
  - Generate ouput, send to address in Data Memory.
- At the Data Memory:
  - MemtoReg == 1 → Send read data back to Register File.
- Finally:
  - RegWrite signal → write data to write address.

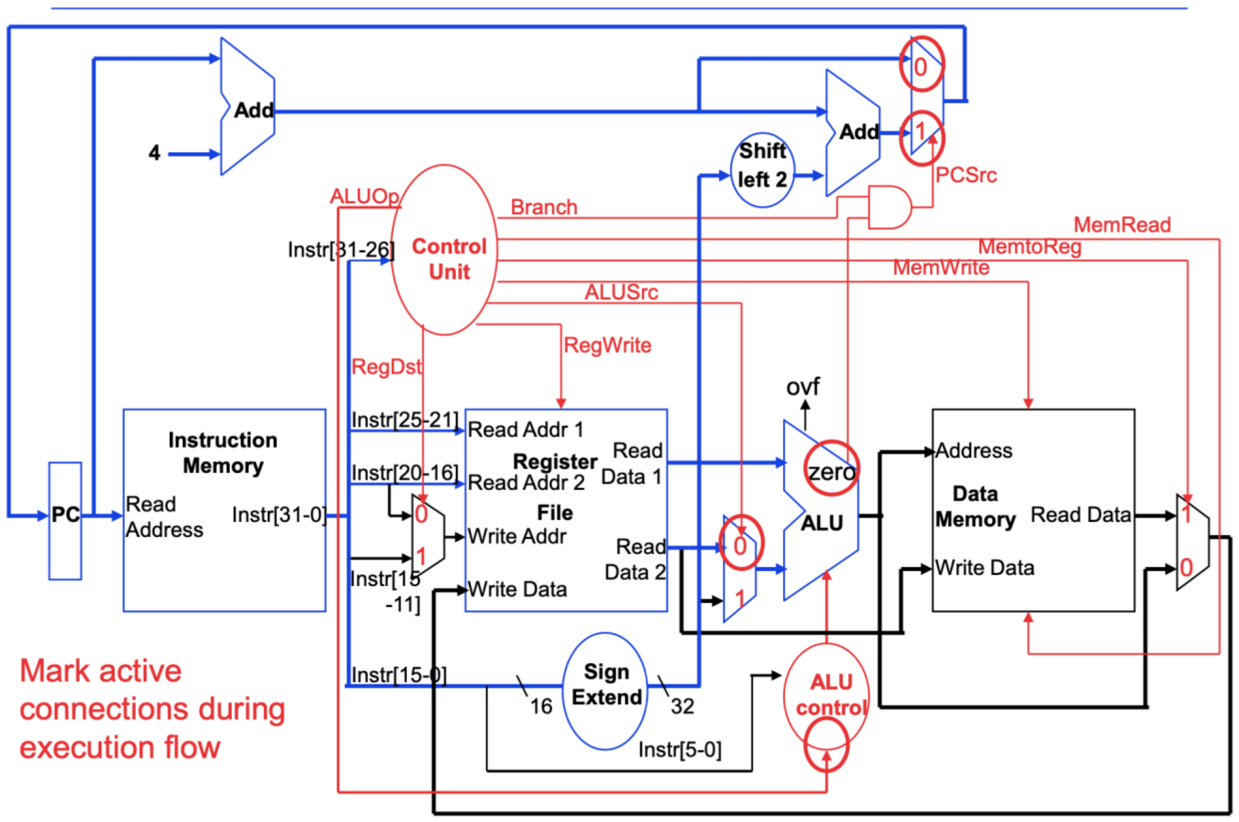


Figure 3. Branch (beq)

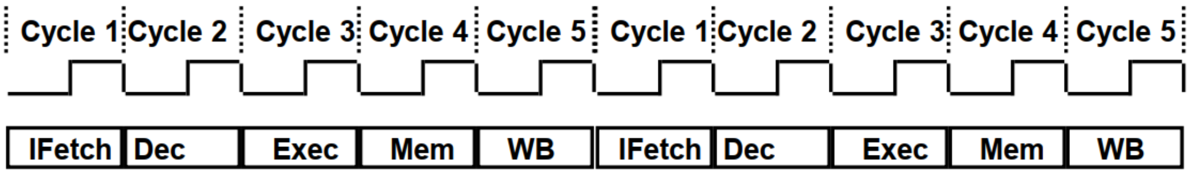
- With the address pointed at by the PC:
  - The instruction is fetched the Instruction Memory.
  - PC is updated at the same time (PC = PC + 4).
- The instruction is decoded:
  - Instr[31-26]: Opcode is sent to the Control Unit, which generates all the red color signal to control others components. Control Unit also generates ALU control signal to the ALU.
  - Instr[25-21]: Register rt is sent to read port 1.
  - Instr[20-16]: Register rs is sent to read port 2.
  - Instr[15-0]: Send to sign extend to 32-bit → Shift left 2 → PC add component.
- At the ALU:
  - Subtraction on read data 1, read data 2 as ALUOp of beq.
  - Generate zero flag, go through the and gate as branch signal == 1.
- Zero flag == 0 → branch is not taken. (PC = PC + 4)
- Zero flag == 1 → branch is taken. (PC = PC + 4 + offset << 2)

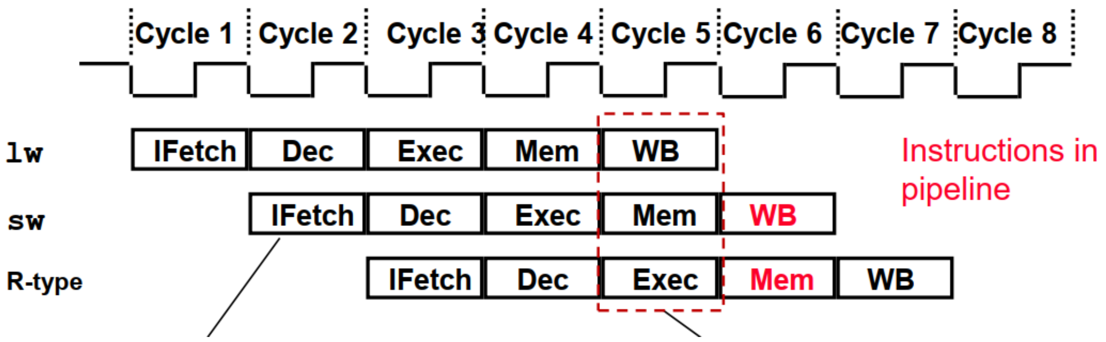
Pipelining:

- The basic idea of pipeline is to split the processor instruction intro series of small independent stages. Each stage is designed to perform a certain part of the instruction.
- At a very basic level, theses stages can be broken down into:
  - IFetch: Instruction fetch and update PC.
  - Dec: Register fetch and instruction decode.

- Exec: Execute R-type, calculate memory address
- Mem: Read/write the data from/to the Data Memory
- WB: Write the result data into the register file

- Instead of execute instruction sequence by sequence, pipeline start fetching and executing the next instruction before the current one has complete. Therefore, in the ideal situation, more than one instruction are executed at a time.





- In the above graph, when execute instructions sequentially, it takes total 10 cycles to complete 2 instructions. On the other hand, with pipeline approach, the cost is only 6 cycles. However, this approach might increase the execution time of 1 instruction but overall, it will improve the running time of the whole program (~ x4 when #instruction → oo).
- Note: **All the situation mentioned above is considered as ideal.**

Hazards:

- The problem that may affect the real performance of CPU pipeline is hazards. Hazards are situations that prevent starting the next instruction in the next cycle.
  - Structural hazards: attempt to use the same resource by two different instructions at the same time.
  - Data hazards: attempt to use data before it is ready. An instruction's source operands are produced by a prior instruction still in the pipeline.
  - Control hazards: attempt to make a decision about program control flow before the condition has been evaluated and the new PC target address calculated. Branch and jump instructions, exceptions.

CHAP 5:

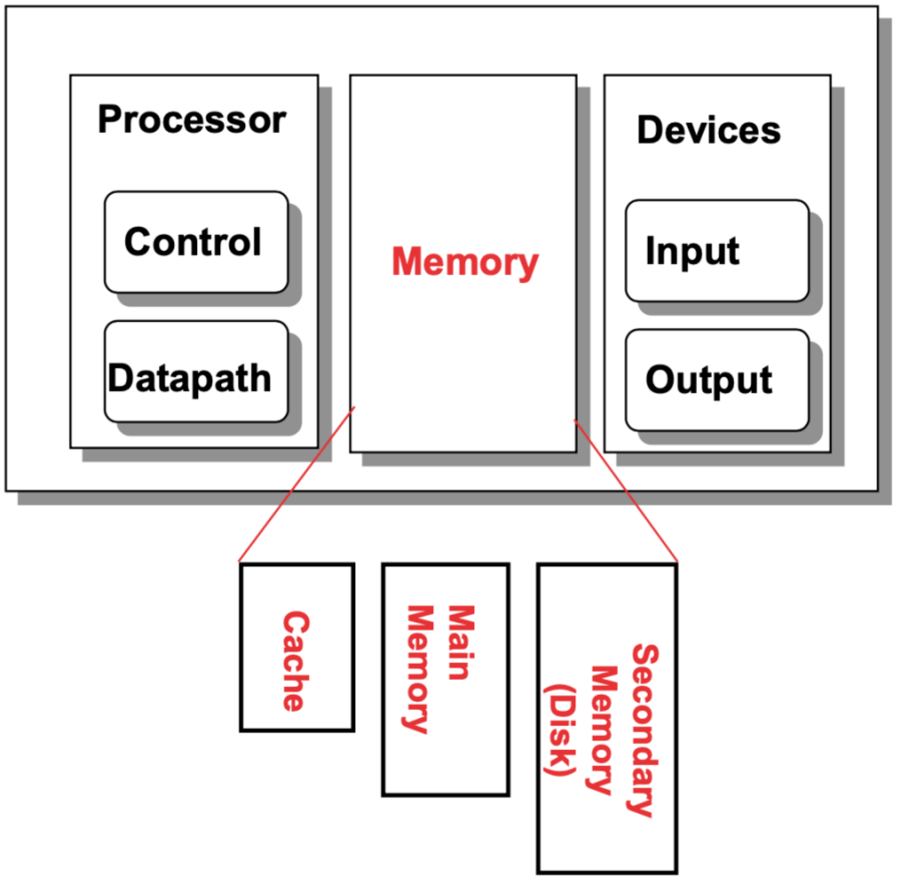
Why is computer's memory designed with hierarchical architecture? What are the main levels of computer's memory?

How cache memory can help improve computer's performance?

Hierarchical architecture:

- The main reason for having memory hierarchy is performance, which means that the access time can be reduced.
- The goal is to present the user with as much memory as is available in the cheapest technology, while providing access at the speed offered by the fastest memory.
- We take advantage of the principle of locality by implementing the memory of a computer as a memory hierarchy, it consists of multiple levels of memory with different sizes and speeds.
    - Temporal locality (in time): keep most recently accessed data.
    - Spatial locality (in space): move blocks consisting of contiguous words closer to the processor.

Main levels of computer's memory:



How cache memory can help improve computer's memory?

- Cache is a small amount of memory which is part of the CPU which is physically closer to the CPU than RAM is. The more cache there is, the more data can be stored closer to the CPU.
- Cache memory is beneficial because:
    - Cache memory holds frequently used instructions/data which the processor may require next and it is faster access memory than RAM, since it is on the same chip as the processor.
    - This reduces the need for frequent slower memory retrievals from main memory, which may otherwise keep the CPU waiting.

CHAP 6:

Most importance metrics of I/O system:

- Cost
- Dependability

I/O system performance measures:

- I/O bandwidth (throughput): amount of information that can be input/output and communicated across an interconnect between the processor/memory and I/O device per unit time.
- I/O response time (latency): the total elapsed time to accomplish an input or output operation.

Why is I/O system designed with hierarchical architecture?

- There are many types of I/O device with different parameter (bandwidth, respond time, MTTF, MTTR,…) and priority.
- Hence, in order to optimize performance and improve user experience, the I/O system is designed with hierarchical architecture.
- To be more specific, at the top of the hierarchy, there are devices having higher priority, smaller respond time and the priority decrease along with the increase of response time as we go down the hierarchy.

What are the three I/O communication modes?

- Polling: the processor periodically checks the status of an I/O device to determine its need for service.
- Interrupt driven I/O: The I/O device issues an interrupt to indicate that it needs attention. The processor detects and "serves" the interrupt by executing a handler (aka. Interrupt Service Routine).
- Direct Memory Access (DMA): the DMA controller has the ability to transfer large blocks of data directly to/from the memory without involving the processor.

PROGRAMMING

Write MIPS assembly programs:

1. Implement bubble sort algorithm
2. With the Digital Lab Sim and Keyboard and Display MMIO simulator in MARS:
    a. Control the display of two 7-seg LEDs.

b. Continuously read keys pressed on Keyboard simulator, detect if that is a numeric character, then display that number on the left 7-seg LED of the Digital Lab Sim.