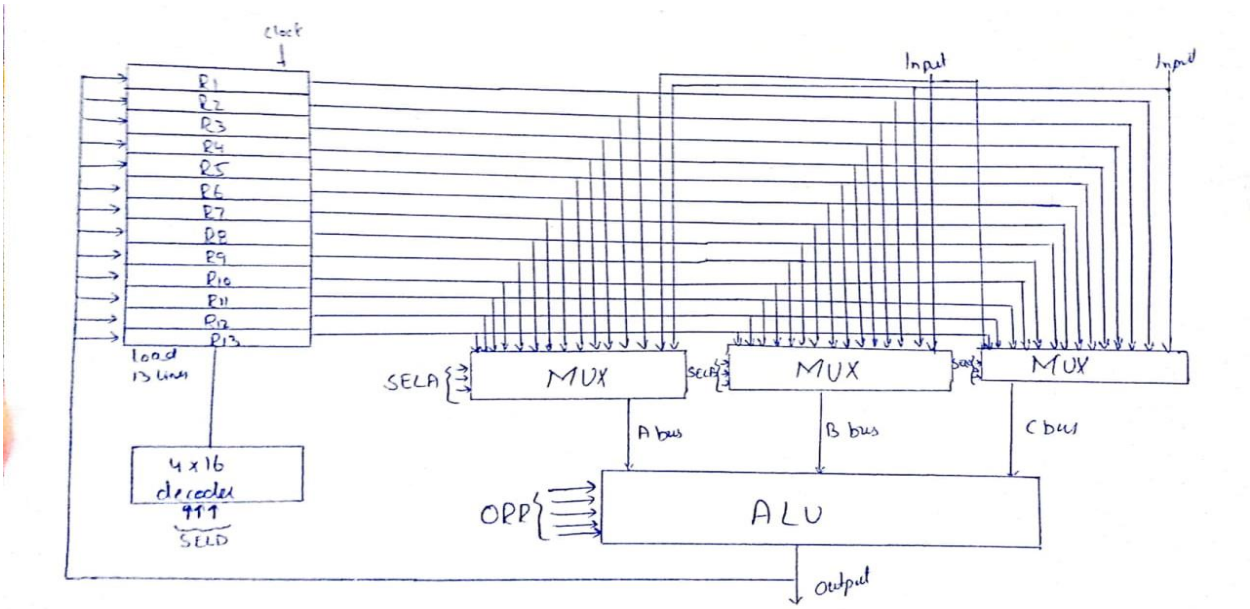
Q1 (a)



Q1 (b)

Virtual memory uses both computer hardware and software to work. When an application is in use, data from that program is stored in a physical address using RAM. More specifically, virtual memory will map that address to RAM using a memory management unit (MMU). The OS will make and manage memory mappings by using page tables and other data structures. The MMU, which acts as an address translation hardware, will automatically translate the addresses.

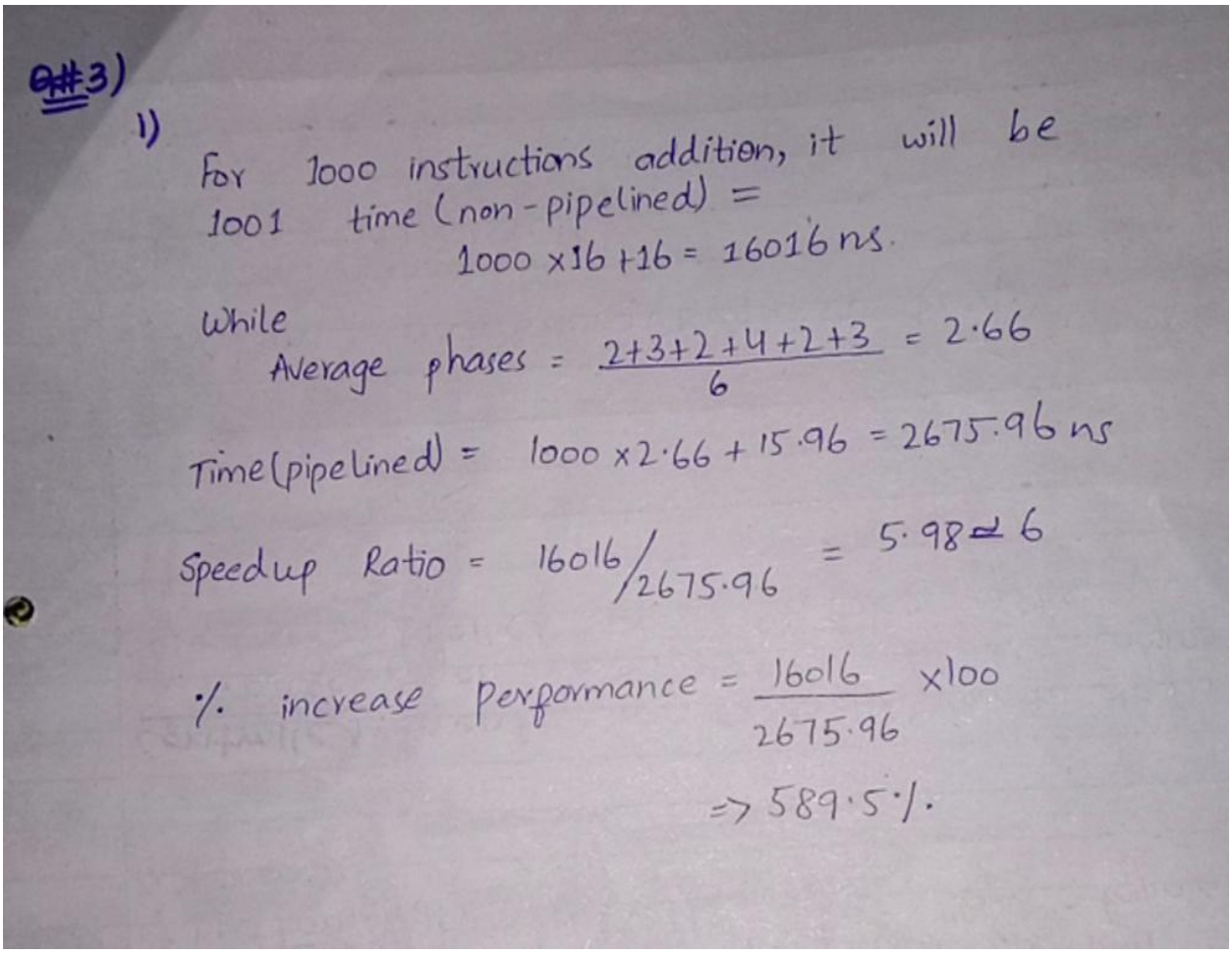
Q2 (a)

RISC stands for reduced instruction set computer. These types of microprocessor architecture utilizes a small, highly optimized set of instructions. RISC processors are mostly used in embedded systems because these processors are lower in silicon cost and power consumptions. RISC emphasis on software single clock. Usually have large code size but low cycles/seconds. Using the RISC approach, the core ARM processor requires only 35,000 transistors, compared to the millions in many conventional processor chips, resulting in lower power usage and making it very attractive in smaller devices. ARM has also a very good system of instruction pipelining which heavily increases the speed of the system.

Q2 (b)

	Opcode	Memory_Reg	ALU_Src	Reg_det
lw	100011	1	1	0
sw	101011	x	1	x
beq	000100	x	0	x
addi	000000	0	0	1
add	001000	0	0	1

Q3 (a)



Q3 (b)

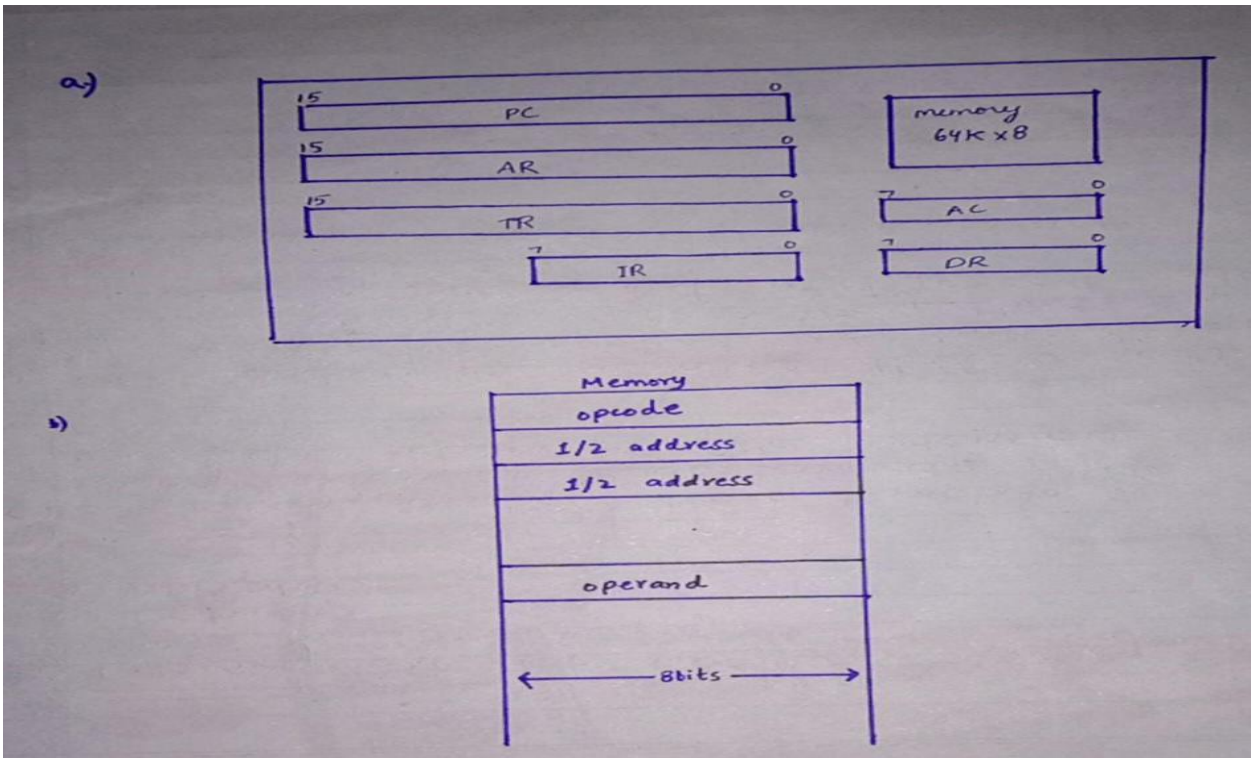
RISC:

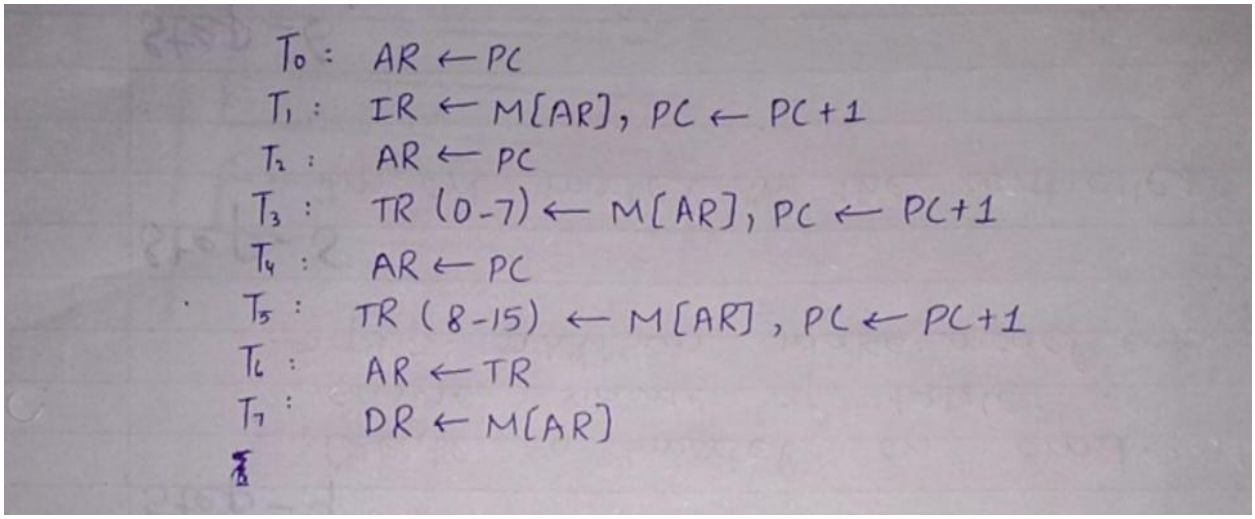
- No memory unit.
- RISC has simple instruction taking about one clock cycle

CISC:

- Has a memory unit.
- CISC has complex instruction that take multiple clock for execution.

Q4 (a, b, c)





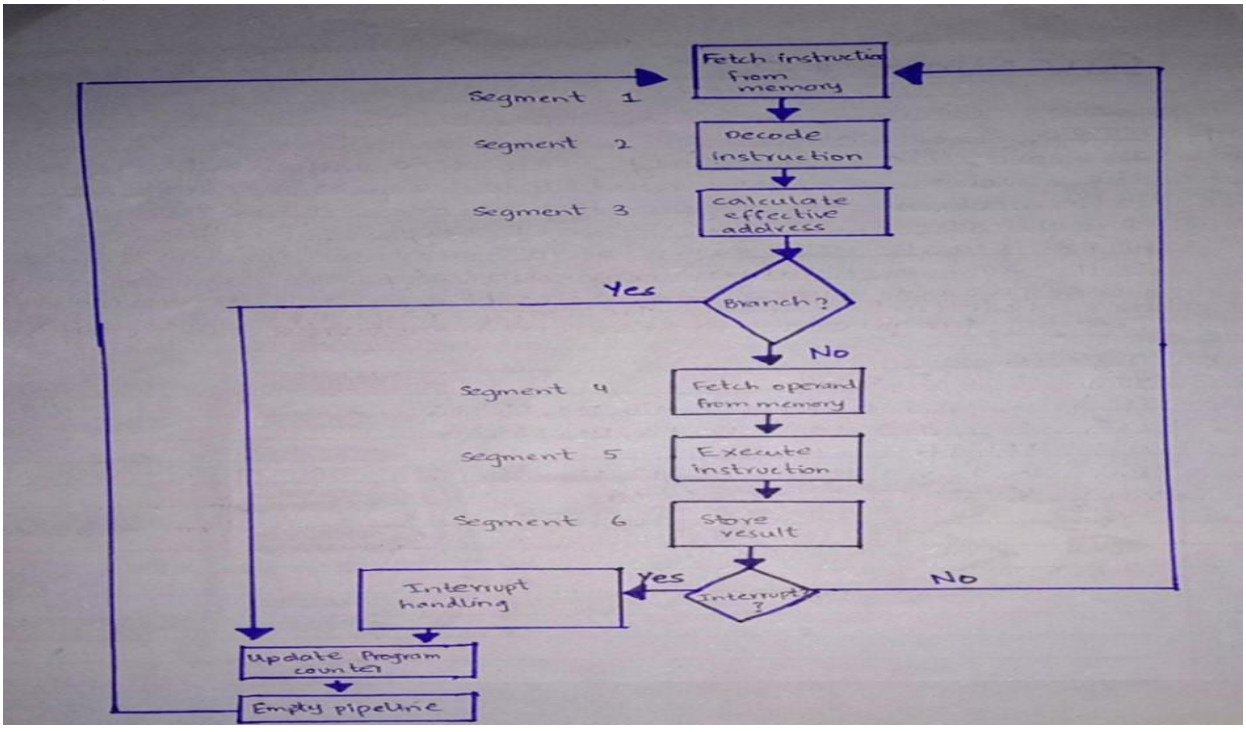
Q4 (2)

Modern operating systems provide a general-purpose mechanism for processing data larger than available main memory called *virtual memory*. Transparent to the program, *swapping* moves parts of the data back and forth from disk as needed. Usually, the virtual address space is divided into units called pages, the corresponding equal-size units in physical memory are called page frames. A *page table* maps the virtual addresses on the page frames and keeps track of their status (loaded/absent). When a *page fault* occurs (i.e., a program tries to use an unmapped page), the CPU is interrupted; the operating system picks a rarely picked page frame and writes its contents back to the disk. It then fetches the referenced page into the page frame just freed, changes the map, and restarts the trapped instruction. In modern computers memory management is implemented on hardware with a page size commonly fixed at 4,096 bytes.

Q5 (a)

Each CPU core is basically a separate processor on its own. John's argument doesn't look valid in this case. The performance of computer will not increase by 100% by doing this. Modern processor have multiple cores on a single chip, effectively just doing that. Each CPU core shares a few resources with other processors (such as memory controller and last-level cache). By adding a new CPU to increase the performance is not good to me. He should add a ram, upgrade the drivers. So to me, this argument is not valid.

Q5 (2)



The instruction cycle can be processed with six segments which are as follows:

• Fetch the instruction from memory: Fetch instruction is use for fetching the instruction from the memory.

• Decode the instruction: It is use for decoding the instruction.

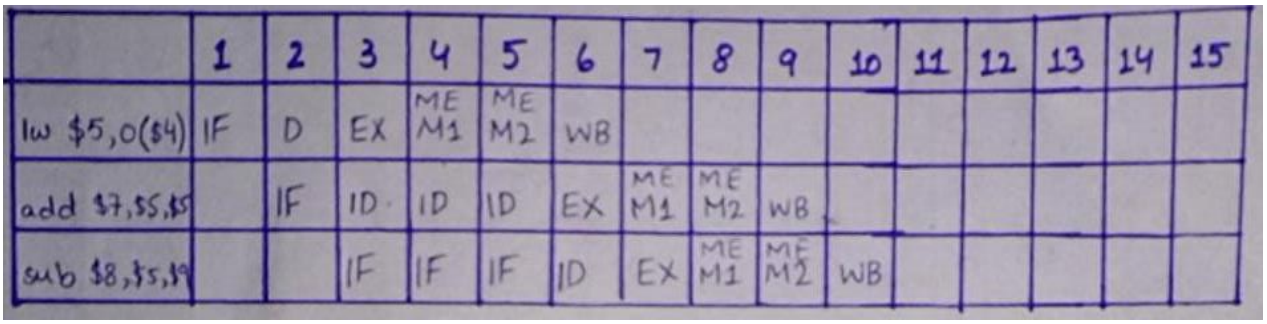
• Calculate the effective address: It is use for calculating the effective address of the instruction

• Fetch the operands from memory: It is use for fetching the operands which is present in the memory.

• Execute the instruction: It is use for executing the instruction from the memory.

• Store the result in proper place: After the execution of the instruction, storing the instruction in the proper place after execution

Q6 (a)



Q6 (b)

An instruction is stored at location 300 with its address field at location

301', the address field has value400. A processor register RI contains the number 200. Evaluate the effective address matching the following addressing modes to their respective effective addresses.

(a) Direct

(b) Immediate

(c) Relative

(d) Register indirect

(e) Index (R1 is index)

(1) 702

(2)2000

(3) 400

(4) 600

(5) 301

(A) a-3 b-5 c-1 d-2 e-4

(B) a-3 b-4 c-2 d-1 e-5

(C) a-5 b-3 c-2 d-1 e-4

(D) a-4 b-3 c-1 d-5 e-2

In the Direct Addressing Mode, the address part of the instruction is equal to the effective address. Here, the operand resides in the memory. Register address is given directly by the address field of the instruction. Hence, Effective Address = 400.

In the Immediate Addressing Mode, when the instruction is assembled, the operand comes immediately after the opcode and as such, data is a part of the instruction itself. Hence, Effective Address = 301.

In the Relative Addressing Mode, the Program Counter (PC) is the implicitly referenced register. So, the effective address is generated by adding the next instruction address to the the address field. Hence, Effective Address = 302 + 400 = 702.

In the Register Indirect Addressing Mode, the operand (which is the contents of a register) is specified by giving the name of the register (in this case, the processor register) in the instruction. Hence, Effective Address = 200. In the Indexed Register Addressing Mode, the effective address is the sum of the contents of 2 registers. Since, RI here acts as the index register and the address field contains the value 400, hence Effective Address = 200 + 400 = 600

Q6 (c)

Cache Memory is a special very high-speed memory. It is used to speed up and synchronizing with high-speed CPU. Cache memory is costlier than main memory or disk memory but economical than CPU registers. Cache memory is an extremely fast memory type that acts as a buffer between RAM and the CPU. It holds frequently requested data and instructions so that they are immediately available to the CPU when needed.

Cache Mapping:

1) Direct Mapping:
   The simplest technique, known as direct mapping, maps each block of main memory into only one possible cache line.

2) Associate Mapping:
   In this type of mapping, the associative memory is used to store content and addresses of the memory word. Any block can go into any line of the cache

3) Set Associate Mapping:
   This form of mapping is an enhanced form of direct mapping where the drawbacks of direct mapping are removed.

In this case, the cache consists of a number of sets, each of which consists of a number of lines. The relationships are

$m = v * k$

$i = j \bmod v$

Where,

i=cache set number

j=main memory block number

v=number of sets

m=number of lines in the cache number of sets

k=number of lines in each set.