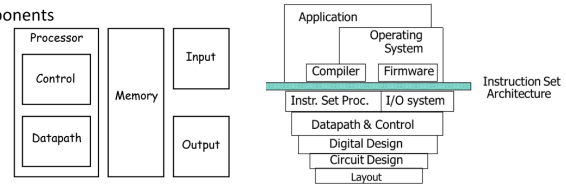
Draw the basic components of a computer and Layer of a computer.
Show the relationship among Instruction Set, Software and
Hardware that defines computer architecture.
Two computers better ?

components



3. itsa ratio between two systems. inverse relation between performance and execution time.

$$\frac{\text{Relative Performance}}{\text{Performance}_Y} = \frac{\text{Performance}_X}{\text{Performance}_Y} = \frac{\text{Execution Time}_Y}{\text{Execution Time}_X} = n$$

4. performance of computers are measured by analyzing the processing unit's task execution times. the faster the result is, the better the performance.

Q. Briefly explain all types of instruction format in MIPS.

Ans:

An R-Type instruction contains 6 fields: a 6 bit function code (*funct*), a 5 bit shift amount (*shamt*), three 5 bit register addresses (*rd*, *rt*, *rs*), and a 6 bit operation code (*opcode*) which is always zero.

An I-Type instruction contains 4 fields: a 16 bit immediate field (*immed.* or *address*), two 5 bit register addresses (*rt*, *rs*) and a 6 bit operation code (*opcode*).

A J-Type instruction contains 2 fields: a 26 bit jump destination (*target*) and a 6 bit operation code (*opcode*).

Name	Fields						Comments
Field size	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	All MIPS instructions 32 bits
R-format	op	rs	rt	rd	shamt	funct	Arithmetic instruction format
I-format	op	rs	rt	address/immediate			Transfer, branch, imm. format
J-format	op	target address					Jump instruction format

MIPS Arithmetic design Principle 1 : simplicity favors regularity. all MIPS instructions must have 3 operands. For this characteristics, ~~phase~~ phase execution ~~struc~~ cycle structure is easy to design and implement.

C code : s_0 s_1 s_2 s_3
$A = B + C + D$ ← temporary register
$A = E + D \Rightarrow E = B + C$
$F = E - A$

MIPS code : add $t_0, \$s_1, \$s_2$ } 1st we need to assign all variables in registers.
(arithmetic) add $\$s_0, \$t_0, \$s_3$ } $A=s_0, B=s_1, C=s_2, D=s_3$
$E=s_4$ and $F=s_5$
($t_0 = B+C$)

This question considers the basic MIPS, 5-stage pipeline (IF, ID, EXE, MEM, WB).
Assume that you have the following sequence of instructions:
lw $s2, 0($s1) (instr1)
add $s3, $s4, $s2 (instr2)
Sub $s6, $s2, $s3. (instr3)
Show the implementation through 5 stages and explain the implementation for both pipelined and non-pipelined design. (explain if there is any pipeline hazards)

For pipeline:

Program instruction order							
Inst_1 lw $s2, 0($s1)	IF	ID	EXE	MEM	WB		
Inst_2 add $s3, $s4, $s2		IF	ID	EXE	MEM	WB	
Inst_3 Sub $s6, $s2, $s3			IF	ID	EXE	MEM	WB
Cycle time	1	2	3	4	5	6	7

There will be a data hazard in instruction-3. Because the register $S3 is fetched before it was updated in the instruction-2. So we will not get the updated value at instruction-3. So we have to use bubble to overcome this problem.

Program instruction order								
Inst_1 lw $s2, 0($s1)	IF	ID	EXE	MEM	WB			
Inst_2 add $s3, $s4, $s2		IF	ID	EXE	MEM	WB		
			Bubble	Bubble	Bubble	Bubble	Bubble	
Inst_3 Sub $s6, $s2, $s3				IF	ID	EXE	MEM	WB
Cycle time	1	2	3	4	5	6	7	8

For non-pipeline:

Program instruction order															
Inst_1 lw $s2, 0($s1)	IF	ID	EXE	MEM	WB										
Inst_2 add $s3, $s4, $s2						IF	ID	EXE	MEM	WB					
Inst_3 Sub $s6, $s2, $s3											IF	ID	EXE	MEM	WB
Cycle time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15