1) Consider three different processors P1, P2, and P3 executing the same instruction set. P1 has a 3 GHz clock rate and a CPI of 1.5. P2 has a 2.5 GHz clock rate and a CPI of 1.0. P3 has a 4.0 GHz clock rate and has a CPI of 2.2.
a. Which processor has the highest performance expressed in instructions per second?
b. If the processors each execute a program in 10 seconds, find the number of cycles and the number of instructions.
c. We are trying to reduce the execution time by 30%, but this leads to an increase of 20% in the CPI. What clock rate should we have to get this time reduction?

Answer 1) -
a)
Formula utilized : Performance = clock rate/CPI (Cycles per instruction)
Performance of P1: 3GHz/1.5 = 2 x 10^9 Inst/sec
Performance of P2: 2.5GHz/1.0 = 2.5 x 10^9 Inst/sec
Performance of P3: 4GHz/2.2 = 1.8 x 10^9 Inst/sec

P2 has the highest performance expressed in instructions per second.

b)
Formula utilized : Cycles = Clock rate * time
Cycles of P1: 3 GHz x 10 s = 30 B cycles
Cycles of P2: 2.5GHz x 10 s = 25 B cycles
Cycles of P3: 4 GHz x 10s = 40 B cycles

Formula utilized : Number of Instructions = Cycles / CPI
Number of Instructions of P1: 30 B cycles/1.5 Cycles per Instruction = 20B
Number of Instructions of P2: 25 B cycles/1 Cycles per Instruction = 25B
Number of Instructions of P3: 40 B cycles/2.2 Cycles per Instruction = 18.18B

c)
Time reduced 30% from 10 sec to 7 sec.
CPI increased by 20% i.e. from (1.5 to 1.8, 1 to 1.2 and 2.2 to 2.6)

Formula utilized : Number of Instructions = ( Clock rate * time ) / CPI
Clock rate_new = Number of Instructions x CPI_new / time_new
Clock_rate_new P1 = 20 B x 1.8 / 7s = 5.14 GHz
Clock_rate_new P2 = 25 B x 1.2 / 7s = 4.28 GHz
Clock_rate_new P3 = 18.18 B x 2.6 / 7s = 6.75 GHz

2) Assume for arithmetic, load/store, and branch instructions, a processor has CPIs of 1, 12, and 5, respectively. Also assume that on a single processor a program requires the execution of 2.56E9 arithmetic instructions, 1.28E9 load/store instructions, and 256 million branch instructions.
Assume that each processor has a 2 GHz clock frequency. Assume that, as the program is parallelized to run over multiple cores, the number of arithmetic and load/store instructions per processor is divided by 0.7 × p (where p is the number of processors) but the number of branch instructions per processor remains the same

a. Find the total execution time (ET) for this program on 1, 2, 4, and 8 processors, and show the relative speedup of the 2, 4, and 8 processors result relative to the single processor result.
b. If the CPI of the arithmetic instructions were doubled, what would the impact be on the execution time of the program on 1, 2, 4, or 8 processors?
c. To what extent should the CPI of load/store instructions be reduced in order for a single processor to match the performance of four processors using the original CPI values?

Answer 2)
a)
Clock rate : 2 * 10^9 Hz
CPI : 1, 12 and 5
Number of Instructions : 2.56E9 / (0.7 * p), 1.28E9 / (0.7 * p) and 2.56E8
Formula utilized : Number of Instructions = ( Clock rate * time ) / CPI
Time = (Number of Instructions * CPI ) / Clock rate

Execution Time
P1 (1)
T1 = 2.56 * 10^9 * 1 / 2 * 10^9= 1.28
T2 = 1.29 * 10^9 * 12 / 2 * 10^9= 7.74
T3 = 2.56 * 10^8 * 5 / 2 * 10^9= 0.64
Total execution time = 9.66 sec

P2 (2)
T1 = 2.56 * 10^9 * 1 / 2 * 10^9 * (0.7 * 2) = 0.914
T2 = 1.29 * 10^9 * 12 / 2 * 10^9 * (0.7 * 2) = 5.52
T3 = 2.56 * 10^8 * 5 / 2 * 10^9 = 0.64
Total execution time = 7.074 sec

P3 (4)
T1 = 2.56 * 10^9 * 1 / 2 * 10^9 * (0.7 * 4) = 0.457
T2 = 1.29 * 10^9 * 12 / 2 * 10^9 * (0.7 * 4)= 2.76
T3 = 2.56 * 10^8 * 5 / 2 * 10^9 = 0.64
Total execution time = 3.857 sec

P4 (8)

T1 = 2.56 * 10^9 * 1 / 2 * 10^9 * (0.7 * 8)= 0.228
T2 = 1.29 * 10^9 * 12 / 2 * 10^9 * (0.7 * 8) = 1.38
T3 = 2.56 * 10^8 * 5 / 2 * 10^9 = 0.64
Total execution time = 2.248 sec

Speedup
P1/P2 = 9.66/7.074 = 1.365
P1/P3 = 9.66/3.857 = 2.504
P1/P4 = 9.66/2.248 = 4.297

b)
Arithmetic CPI doubled i.e. T1 would be two times the original execution time.
Formula utilized : ET_new = ET_old + T1
P1 ET_new = 9.66 + 1.28 = 10.94 sec
P2 ET_new = 7.074 + 0.914 = 8.048 sec
P3 ET_new = 3.857 + 0.457 = 4.314 sec
P4 ET_new = 2.248 + 0.228 = 2.476 sec

The execution time is increased for all the processors, P4 having the lowest increase.

Speedup
P1/P2 = 10.94/8.048 = 1.359
P1/P3 = 10.94/4.314 = 2.536
P1/P4 = 10.94/2.476 = 4.418

The speedup for P1 vs P4 shows a significant increase.

c)
Updated values of P1 having single processor -
P1 (1)
T1 = 2.56 * 10^9 * 1 / 2 * 10^9= 1.28
T2 = 1.29 * 10^9 * x / 2 * 10^9= 0.645x
T3 = 2.56 * 10^8 * 5 / 2 * 10^9= 0.64

(T1+T2+T3) of P1 = Total execution time of P3 having 4 parallel processors
1.28+0.64+ 0.645x = 3.857
1.92+0.645x = 3.857
0.645x=1.937
x=3.003 (new CPI of load/store instruction)

3)
Consider the three different processors P1, P2, and P3 executing the same instruction set. P1 has a clock cycle time of 0.33 ns and CPI of 1.5; P2 has a clock cycle time of 0.40 ns and CPI of 1.0; P3 has a clock cycle time of 0.3 ns and CPI of 2.8.
a. Which has the highest clock rate? What is it?
b. Which is the fastest computer? If the answer is different than above, explain why. Which is slowest?
c. How do the answers for a and b reflect the importance of benchmarks?

Answer 3)
a)
Formula utilized : Clock rate = 1 / clock cycle time
CR1= 1/0.33ns = 3GHz
CR2= 1/0.40ns = 2.5GHz
CR3= 1/0.3ns = 3.3GHz

P3 has the highest clock rate.

b)
Formula utilized : Performance = clock rate/CPI (Cycles per instruction)
p1=3/1.5=2.0 Inst/sec
p2=2.5/1.0=2.5 Inst/sec
p3=3.3/2.8=1.19 Inst/sec

P2 is the fastest as it takes the least amount of time to do the same work.
P3 is the slowest as it takes the least amount of time to do the same work. The clock rate is high but, due to high CPI the performance is degraded.

c) These answers actually help us gauge the real life problems in a more systematic way such that it can help us decide which among the 2 preferred options do we choose from or maybe to check if these parameters are not unrealistic while making a retail purchase scenario.

4)
You are designing a system for a real-time application in which specific deadlines must be met. Finishing the computation faster gains nothing. You find that your system can execute the necessary code, in the worst case, twice as fast as necessary.
a. How much energy do you save if you execute at the current speed and turn off the system when the computation is complete?
b. How much energy do you save if you set the voltage and frequency to be half as much?

Answer 4)
a) There is no advantage to completing the execution sooner. Processor logic transitions determine how much energy is used. There will be no energy savings because the amount of logic instructions and transitions in the CPU stays the same. Additionally, if the system is operated at the current speed and then turned off, no energy can be saved because the voltage and frequency requirements while operation are higher and nearly equivalent to twice the levels required.

b) Formula Used : P = F x ½ (CV^2) ; E = ½ load*(V^2)
Hence, half the voltage would decrease the energy by 1/4th times (25%) although, there won't be any effect of frequency on energy.

5)
Consider two different implementations of the same instruction set architecture. The instructions can be divided into four classes according to their CPI (classes A, B, C, and D). P1 with a clock rate of 2.0 GHz and CPIs of 1, 2, 2, and 1, and P2 with a clock rate of 4 GHz and CPIs of 2, 3, 4, and 4.
a. Given a program with a dynamic instruction count of 1.0E6 instructions divided into classes as follows: 10% class A, 20% class B, 50% class C, and 20% class D. Which is faster: P1 or P2 (in total execution time)?
b. What is the global CPI for each implementation?
c. Find the clock cycles required in both cases.
d. Which processor has the highest throughput performance (instructions per second) ?
e. Which processor do you think is more energy efficient? Why?

Answer 5)
a)

Formula utilized : Time = (Number of Instructions * CPI ) / Clock rate

| | A (10%) | B (20%) | C (50%) | D (20%) |
|---|---|---|---|---|
| P1 | 1 | 2 | 2 | 1 |
| P2 | 2 | 3 | 4 | 4 |

Time of P1 = 10^6 *(0.1 X 1 + 0.2 X 2 + 0.5 X 2 + 0.2 X 1)/2 * 10^9 = 10^-3 * 1.7/2 = 8.5*10^-4 sec
Time of P2 = 10^6 *(0.1 X 2 + 0.2 X 3 + 0.5 X 4 + 0.2 X 4)/4 * 10^9= 10^-3 * 3.6/4 = 9*10^-4 sec

P1 in this case is faster.

b)
Formula utilized : Global CPI = sum of CPI*class_share
P1 = 0.1 X 1 + 0.2 X 2 + 0.5 X 2 + 0.2 X 1 = 1.7
P2 = 0.1 X 2 + 0.2 X 3 + 0.5 X 4 + 0.2 X 4 = 3.6

c)
Formula utilized : Clock cycles required = CPI x IC
P1: 1.7 x 10^6 = 1.7M cycles
P2: 3.6 x 10^6 = 3.6M cycles

d)
Formula utilized : Performance = clock rate/CPI (Cycles per instruction)
p1 = 2/1.7 = 1.17 x 10^9 Inst/sec
p2 = 4/3.6 = 1.11 x 10^9 Inst/sec

e)
P1 is more efficient as it is able to process more instructions per second.

6)
a. What is the difference between CISC and RISC architectures? Give some examples wherein you think CISC architectures are better suited for than RISC architectures and vice versa.
b. Describe in your own words why you think RISC V would be a better alternative compared to ARM or x86 architectures?
c. What do you think are the challenges faced by RISC V architecture going forward?

Answer 6)
a)

| CISC (Complex Instruction Set Computer) | RISC (Reduced Instruction Set Computer) |
|---|---|

| | |
|---|---|
| Huge number of instructions | Small number of fixed length instructions |
| Instructions take varying amount of time | Instructions take fixed amount of time |
| Format of instructions is variable | Format of instructions is fixed |
| Hardware centric model | Software centric model |
| Fewer registers | More registers |
| Efficient RAM usage | High RAM consumption |
| Pipeline are difficult to implement | Pipelining can be implemented |

The CISC architecture is better when implemented in security systems or home automation because it consumes less RAM and complex instructions can be handled.

Whereas, RISC is used in high-end applications such as video processing, telecommunications, and image processing due to energy efficiency, more registers making it cheaper and the faster processing.

b)
Advantages of using RISC -

- RISC-V and ARM processors are based on RISC concepts in terms of computing architectures, while x86 processors from Intel and AMD employ CISC designs.
- x86 and ARM are based on proprietary IP, and the companies sell and/or license their products. RISC-V is an open specification and platform; it is not an open-source processor. Open-source RISC-V cores are available, but there are also commercially licensed cores.
- RISC-V is an open source hardware while x86 and ARM are closed source.
- There is no hassle of license fees
- RISC-V is cheaper than ARM
- RISC-V consumes less time
- RISC-V's inherent modularity and design freedom make it the ideal choice

c)
The challenges faced by RISC V architecture going forward might be as follows -

- The compilers may not be consistent. The entire ecosystem is fragmented.
- Few tools might be missing as it is relatively new
- The three layer security framework is very weak.
- The high RAM requirements might be difficult to set aside post a certain tenure.

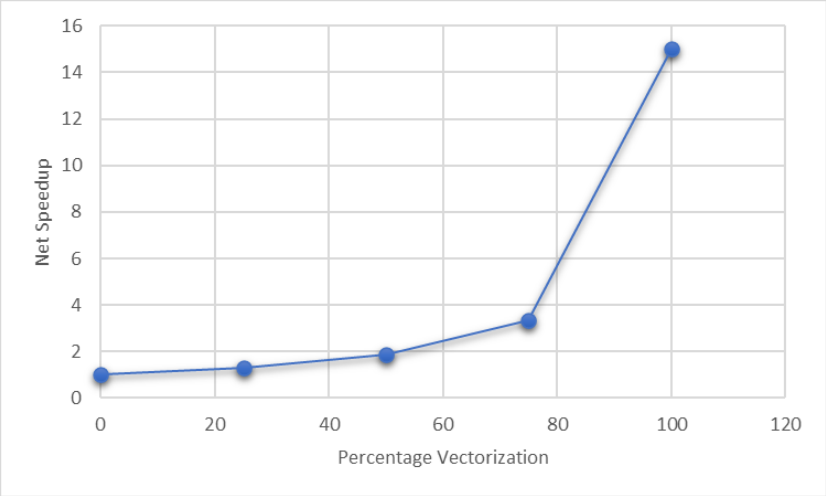
7)

In this exercise, assume that we are considering enhancing a machine by adding vector hardware to it. When a computation is run in vector mode on the vector hardware, it is 15 times faster than the normal mode of execution. We call the percentage of time that could be spent using vector mode the percentage of vectorization. Vectors are discussed in Chapter 4, but you don't need to know anything about how they work to answer this question!
a. Draw a graph that plots the speedup as a percentage of the computation performed in vector mode. Label the y-axis "Net speedup" and label the x-axis "Percent vectorization."
b. What percentage of vectorization is needed to achieve a speedup of 2?
c. What percentage of the computation run time is spent in vector mode if a speedup of 2 is achieved?
d. What percentage of vectorization is needed to achieve one-half the maximum speedup attainable from using vector mode?
e. Suppose you have measured the percentage of vectorization of the program to be 70%. The hardware design group estimates it can speed up the vector hardware even more with significant additional investment. You wonder whether the compiler crew could increase the percentage of vectorization, instead. What percentage of vectorization would the compiler team need to achieve in order to equal an additional 2× speedup in the vector unit (beyond the initial 15×)?

Answer 7)
a)
Formula utilized : Net Speedup = 1 / ( 1 - PV ) / (PV / faster times)

Computations for plotting the graph as Net Speedup being 1 initially =

Speedup of 25% vectorization = 1 / ( 0.75 + 0.25/15) = 1.304
Speedup of 50% vectorization = 1 / ( 0.5 + 0.5/15) = 1.875
Speedup of 75% vectorization = 1 / ( 0.25 + 0.75/15) = 3.33
Speedup of 100% vectorization = 1 / ( 1/15) = 15



b) 2 = 1 / ( 1 - PV) / (PV / 15).
Solving for PV gives us = 0.535 i.e. 53.5% vectorization
c)
Formula utilized : Computation run time = (PV / faster times) / ( 1 - PV ) / (PV / faster times)
From above, the value(x = speedup = 2) on Y corresponds to 53.5%.
Computation run time = (PV/15) / (PV/15) + (1-PV)
= 3.56 / (3.56 + 46.5)
= 0.0711
I.e. 7.11% computation run time

d)
Half of 15 would be 7.5
7.5 = 1 / ( 1 - PV) + (PV / 15)
Solving for PV gives us = 0.9285 i.e. 92.85% vectorization

e)
70% PV
= 1 /( 1 - PV) + (PV / 15)
= 1 /( 0.3) + (0.7/15)
= 2.88 speedup

Additional 2 * speedup would be 5.769 speedup
5.769 = 1 /( 1 - PV) + (PV / 15)
Solving for PV gives us = 0.885 i.e. 88.5% vectorization
Hence, by doubling the speed the increase would only be of 18.5%

8)
In a server farm such as that used by Amazon or eBay, a single failure does not cause the entire system to crash. Instead, it will reduce the number of requests that can be satisfied at any one time.
a. If a company has 10,000 computers, each with a MTTF of 35 days, and it experiences catastrophic failure only if 1/3 of the computers fail, what is the MTTF for the system?
b. If it costs an extra $1000, per computer, to double the MTTF, would this be a good business decision? Show your work.

Answer 8)
a)
Number of computers = 10,000
Failed computer % = ⅓ rd of 10,000 = 3,333
MTTF = 35 days
Time_failure = t
MTTF = (total number of computers * t)/ failed computers
35 = (10,000 * t) / 3333
11.67 days = t

b)
Increase in cost = 1,000(per computer) * 10,000(computers) = $10^7
MTTF_doubled = 6000 hours
We can go ahead with this decision as it will substantially increase the productivity of the computers.

9)
a. A program (or a program task) takes 150 million instructions to execute on a processor running at 2.7 GHz. Suppose that 70% of the instructions execute in 3 clock cycles, 20% execute in 4 clock cycles, and 10% execute in 5 clock cycles. What is the execution time for the program or task?
b. Suppose the processor in the previous question part is redesigned so that all instructions that were initially executed in 5 cycles and all instructions executed in 4 cycles now execute in 2 cycles. Due to changes in the circuitry, the clock rate also must be decreased from 2.7 GHz to 1.5 GHz. What is the overall percentage improvement?

Answer 9)
a)
Formula utilized : Time = (Number of Instructions * CPI ) / Clock rate

Time_old = 150 *10^6 * (0.7*3 + 0.2*4 + 0.1*5) /2.7 * 10^9
= 0.188 sec

b)
Time_new = 150 *10^6 * (0.7*3 + 0.2*2 + 0.1*2) /1.5 * 10^9
= 0.270 sec

Performance ratio = Time_new/Time_old = 0.277/0.188 = 1.47
There is 47% positive improvement.

10)
Availability is the most important consideration for designing servers, followed closely by scalability and throughput.
a. We have a single processor with a failure in time (FIT) of 100. What is the mean time to failure (MTTF) for this system?
b. If it takes one day to get the system running again, what is the availability of the system?
c. Imagine that the government, to cut costs, is going to build a supercomputer out of inexpensive computers rather than expensive, reliable computers. What is the MTTF for a system with 1000 processors? Assume that if one fails, they all fail.

Answer 10)
a)

Formula utilized : MTTF = 1/FIT
= 10^9/100 = 10^7
Generally failures are per billion hours of Failure in Time (FIT)

b)
Formula utilized : Availability of the system = MTTF/ (MTTF + MTTR)
Availability of the system = 10^7/(10^7 + 24) = 0.999 = 1 failure

c)
Understanding utilized :
Number of processors = 1/ chance of failure
More number of failures will occur when 1 processor have multiple

Formula utilized :
FIT for x processors = Number of processors* FIT /10^9
=(1000*100)/10^9
= 1*(10^-4)

MTTF = 1/FIT
= 1/(10^-4)
=10,000

11)
Server farms such as Google and Yahoo! provide enough compute capacity for the highest request rate of the day. Imagine that most of the time these servers operate at only 60% capacity. Assume further that the power does not scale linearly with the load; that is, when the servers are operating at 60% capacity, they consume 90% of maximum power. The servers could be turned off, but they would take too long to restart in response to more load. A new system has been proposed that allows for a quick restart but requires 20% of the maximum power while in this “barely alive” state.
a. How much power savings would be achieved by turning off 60% of the servers?
b. How much power savings would be achieved by placing 60% of the servers in the “barely alive” state?
c. How much power savings would be achieved by reducing the voltage by 20% and frequency by 40%?
d. How much power savings would be achieved by placing 30% of the servers in the “barely alive” state and 30% off?

Answer 11)
a)
Power consumption P,
Number of servers x
Total power consumed_old = 0.9Px
Total power consumed_new = (0.4*0.9)Px = 0.36Px

Power saved = (0.9x - 0.36x)/0.9x = 0.54x/0.9x = 60%

b)
Total power consumed_old = 0.9Px
Total power consumed_new is 30% servers with 20% power usage, 30% with 0 power usage and 40% servers with power 90% usage
= 0.4x * 0.90 P + 0.3x * 0.20P = 0.42Px
Power saved = (0.9x - 0.48x)/0.9x = 0.48x/0.9x = 53.3%

c)
Formula utilized : Power = (capacitance × voltage^2 × frequence)/2
P_old = 0.5*C(V^2)F = 0.5*C* (0.8V^2) * (0.6F)
P_new = 0.384 * 0.5*C(V^2)F
Power saved = 1-0.384 = 0.616 = 61.6%

d)
Total power consumed_old = 0.9Px
Total power consumed_new is 30% servers with 20% power usage and 40% servers with power 90% usage
= 0.4x * 0.90 P + 0.6x * 0.20P = 0.48Px
Power saved = (0.9x - 0.48x)/0.9x = 0.42x/0.9x = 46.6%

12)
Assume for a given processor the CPI of arithmetic instructions is 1, the CPI of load/store instructions is 10, and the CPI of branch instructions is 3. Assume a program has the following instruction breakdowns: 100 million arithmetic instructions, 20 million load/store instructions, 20 million branch instructions.
a. Suppose that new, more powerful arithmetic instructions are added to the instruction set. On average, through the use of these more powerful arithmetic instructions, we can reduce the number of arithmetic instructions needed to execute a program by 25%, while increasing the clock cycle time by only 10%. Is this a good design choice? Why?
b. Suppose that we find a way to double the performance of arithmetic instructions. What is the overall speedup of our machine? What if we find a way to improve the performance of arithmetic instructions by 10 times?

Answer 12)
a)
Formula utilized : Time = (Number of Instructions * CPI ) / Clock rate
Clock rate = 1 / clock cycle time
Time = (Number of Instructions * CPI ) * clock cycle time

Original Execution Time
P1 (1)
Clock cycle time = x

T1 = (1*100*10^6)/clock rate = 10 *(10^7)*x
T2 = (10*20*10^6)/clock rate = 20 *(10^7)*x
T3 = (3*20*10^6)/clock rate = 6 *(10^7)*x
Total execution time  = 3.6 * 10^8*x

Increased Clock Cycle time = 1.1 x
T1_new = (1*75*10^6)/clock rate = 7.5*(10^7)*1.1x
Total execution time_new  = (20+6+7.5)*(10^7)*1.1x = 33.5*1.1 x *(10^7)
=368.5 * 10^6

No, this is not a good choice because the total execution time is now reduced.

b)
Formula utilized : Time = (Number of Instructions * CPI ) * clock cycle time
T1_twice = (1*100*10^6)/clock rate * 2 = 5*10^7*x
Total execution time  = 3.1 * 10^8*x
T1_ten = (1*100*10^6)/clock rate * 10 = 1*10^7*x
Total execution time  = 2.7 * 10^8*x

Speedup = T_old/T_twice
= 3.6* 10^(8) / 3.1* 10^(8)
= 1.16

~16% increase in speed

Speedup = T_old/T_ten
= 3.6* 10^(8) / 2.7* 10^(8)
= 1.33

~33% increase in speed