**HW1**

**YU Cheng** *yc4845*

**1.** Assume a program requires the execution of 60 × 106 FP instructions, 120 × 106 INT instructions, 60 × 106 L/S instructions, and 16 × 106 branch instructions. The CPI for each type of instruction is 1, 1, 4, and 2, respectively. Assume that the processor has a 2 GHz clock rate.

**a.** By how much must we improve the CPI of FP instructions if we want the program to run two times faster?

To achieve a 2x speedup by improving the CPI of FP instructions alone, the CPI would need to be approximately −2.77, which is not feasible. This indicates that reducing only the FP instruction CPI cannot achieve the desired speedup, suggesting a limitation in only optimizing FP instructions.

**b.** By how much must we improve the CPI of L/S instructions if we want the program to run two times faster?

The CPI for L/S instructions would need to be reduced to approximately 0.23. This is a significant reduction, implying a need for highly optimized memory operations to achieve a 2x speedup.

**c.** By how much is the execution time of the program improved if the CPI of INT and FP instructions is reduced by 50% and the CPI of L/S and Branch is reduced by 25%?

If the CPI of INT and FP instructions is reduced by 50%, and the CPI of L/S and Branch instructions is reduced by 25%, the execution time of the program is improved by approximately 1.54 times. This indicates a 54% performance improvement with these optimizations.

**2.** When a program is adapted to run on multiple processors in a multiprocessor system, the execution time on each processor is comprised of computing time and the overhead time required for locked critical sections and/or to send data from one processor to another.

Assume a program requires t = 200 s of execution time on one processor. When running p processors, each processor requires t/p s, as well as an additional 10 s of overhead, irrespective of the number of processors. Compute the per-processor execution time for 2, 4, 8, 16, 32, 64 processors. For each case, list the corresponding speedup relative to a single processor and the ratio between actual speedup versus ideal speedup (speedup if there was no overhead).

Execution time on one processor (t) = 200 seconds

Overhead time per processor = 10 seconds

Number of processors (p) = 2, 4, 8, 16, 32, 64

**For p=2:**

Execution Time = 200/2+10=110 seconds

Speedup = 200/110≈1.82

Ideal Speedup = 2

Ratio = 1.82/2≈0.91

**For p=4:**

Execution Time = 200/4+10=60 seconds

Speedup = 200/60≈3.33

Ideal Speedup = 4

Ratio = 3.33/4≈0.83

**For p=8:**

Execution Time = 200/8+10=35seconds

Speedup = 200/35≈5.71

Ideal Speedup = 8

Ratio = 5.71/8≈0.71

**For p=16:**

Execution Time = 20016+10=22.5seconds

Speedup = 200/22.5≈8.89

Ideal Speedup = 16

Ratio = 8.89/16≈0.56

**For p=32:**

Execution Time = 200/32+10=16.25seconds

Speedup = 200/16.25≈12.31

Ideal Speedup = 32

Ratio = 12.31/32≈0.38

**For p=64p = 64p=64:**

Execution Time = 200/64+10=13.125seconds

Speedup = 200/13.125≈15.24

Ideal Speedup = 64

Ratio = 15.24/64≈0.24

| **Processors (p)** | **Execution Time (s)** | **Actual Speedup** | **Ideal Speedup** | **Speedup Ratio** |
| --- | --- | --- | --- | --- |
| 2 | 110 | 1.82 | 2 | 0.91 |
| 4 | 60 | 3.33 | 4 | 0.83 |
| 8 | 35 | 5.71 | 8 | 0.71 |
| 16 | 22.5 | 8.89 | 16 | 0.56 |
| 32 | 16.25 | 12.31 | 32 | 0.38 |
| 64 | 13.125 | 15.24 | 64 | 0.24 |

**3.** 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?**

Assuming a uniform distribution of load across all servers and assuming the load does not change in each server when turning off any fraction of the servers, 60% of servers turned off, reduces power by 60%

1. **How much power savings would be achieved by placing 60% of the servers in the “barely alive” state?**

40% of the servers will work normally which cost 40% power. And 60% servers working in the “barely alive” state cost 20% of the maximum power, which is 60%×20%/90% = 0.133%. Thus, the total power is reduced to 40% + 13.3% = 53.3% of nominal power consumption.

1. **How much power savings would be achieved by reducing the voltage by 20% and frequency by 40%?**

For the quadratic dependence of power on voltage and the linear relationship between power and frequency, after the reducing, the new power will be . Thus, the power saving is 1-38.4%= 61.6%.

1. **How much power savings would be achieved by placing 30% of the servers in the “barely alive” state and 30% off?**

40% of the servers are working normally, which cost 40% power. And 30% of the servers are closed, 30% of the servers are working in the “barely alive” state, which cost 30%×20%/90% = 6.67%. The total power is reduced to 40% + 6.67% = 46.67% of nominal power consumption.

**4.** 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.

a.

Reciprocal of MTTF is failure rate - FIT, which is computer/day. For only 1/3 of the computer fail will the system crash. MTTF of the system should be / = 35/3 = 11.67 days.

b.

It costs an extra $1000 per computer, The total extra cost on computer is 1000×10000 = $10 million. And the MTTF of the system is doubled, which is 23.34 days. This is a good business decision because if we spend $1000 on a computer, we can increase the working time of the computers in a company.

**5.** 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 3?

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?

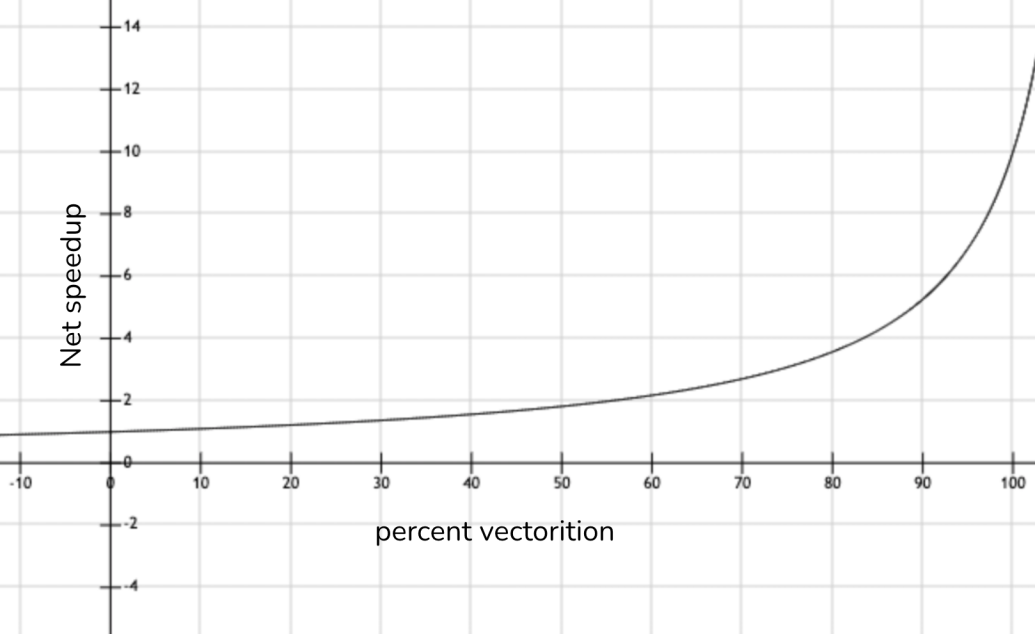
1. Suppose you have measured the percentage of *vectorization of the program to be 70%*. The hardware design group estimates it can speed up the vector hardware even more with significant additional investment. You wonder whether the compiler crew *could increase the percentage of vectorization, instead*. What percentage of vectorization would the compiler team need to achieve in order to equal **an addition 2× speedup in the vector unit** (beyond the initial 15×)?

a.

Assume T0 is the time taken for the non-vectorized code to run. And assume ‘x’ equals the percentage of the code vectorized. This part of code takes 1/10 the time to run as the non-vectorized code. And (100-x)% of the code is not vectorized.

The total time to run code with x % vectorized = (100 – x)% T0 + (x/10)%T0. Therefore, the speedup is 100%×T0 /[(100 – x)% T0 + (x/10)%T0]

Plot of the equation: y = 100/[(100-x)+x/10]

1. 

b.

According to the result in a, to achieve the speedup of 2, the percentage of vectorization*(x)* is:

2 = 100 /[(100 – x) + (x/10)]

x = 55.56

c.

(x/10) / [ (x/10)+ (100-x) ] = 5.56/(5.56 + 44.44) = 11.12

d.

The maximum speedup should be 10. According to the result in a, to achieve one-half the maximum speedup attainable:

1. = 100 /[(100 – x) + (x/10)]

x = 88.89

e.

With a 70% vectorization, speedup would be 100 /[(100 – 70) + (70/10)] = 2.7.

If we want to double the speedup(2.7×2 = 5.4), we can either increasing the speedup of the vector hardware to more than 10 or increasing the speedup with a higher percentage vectorization of the code.

By increasing the speedup with a higher percentage vectorization of the code. The new percentage(x) should be:

5.4 = 100 /[(100 – x) + (x/10)]

x = 90.53

**6.** Assume for arithmetic, load/store, and branch instructions, a processor has CPIs of 2, 10, 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 128 million branch instructions. Assume that each processor has a 2GHz 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 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.** To what 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?

| **Processors (p)** | **Execution Time (s)** | **Speedup** |
| --- | --- | --- |
| 1 | 17.28 | 1 |
| 2 | 6.72 | 2.57 |
| 4 | 3.52 | 4.91 |
| 8 | 1.92 | 9 |

b.

The execution time of a single processor must be the same as the execution time of 4 processors using the original CPI values.

From part a, the execution time for 4 processors was 3.52 seconds.

New total cycles=2.56×109×2+1.28×109×New CPI+128×106×5

=5.12×109+1.28×109×New CPI+640×106 =5.76×109+1.28×109×New CPI

Required cycles for 3.52 seconds:

3.52×2×109=7.04×109

7.04×109=5.76×109+1.28×109×New CPI

New CPI=1

The load/store CPI would need to be reduced from 10 to 1 for the single processor to match the performance of 4 processors using the original CPI values.

**7.** Suppose we developed a simpler processor that has 75% of the capacitive load of a more complex processor. Further, assume that it can adjust voltage so that it can reduce voltage by 20% compared to the complex processor, but this results in a 25% increase in frequency.

**a.** How much energy do we save through this change? **b.** What is the impact on the dynamic power?

a.

Esimple ∝(0.75×Ccomplex) ×(0.8×Vcomplex)2

Energy Saving=(Ecomplex−Esimple)/Ecomplex

=(Ccomplex×Vcomplex2−0.48×Ccomplex×Vcomplex2)/(Ccomplex×Vcomplex2)

=1−0.48=0.52

**52%** of the energy is saved through this change

b.

Pcomplex∝Ccomplex×Vcomplex2×fcomplex

Psimple∝(0.75×Ccomplex)×(0.8×Vcomplex)2×(1.25×fcomplex)

Change in Power=Psimple/Pcomplex=0.6

the dynamic power is reduced by **40%**.

**8.** One challenge for architects is that the design created today will require several years of implementation, verification, and testing before appearing on the market. This means that the architect must project what the technology will be like several years in advance. Sometimes, this is difficult to do.

**a**. According to the trend in device scaling historically observed by Moore’s Law, the number of transistors on a chip in 2025 should be how many times the number in 2015? [assume # Transistors double every 2 years]  
**b**. The increase in performance once mirrored this trend. Had performance continued to climb at the same rate as in the 1990s, approximately what performance would chips have over the VAX- 11/780 in 2025? [assume performance increases 52% every year]

**c**. What has limited the rate of growth of the clock rate, and what are architects doing with the extra transistors now to increase performance?

a.

Given:

* Time span from 2015 to 2025 is 10 years.
* Transistors double every 2 years.

(2025−2015)/2=5 doublings

**Increase in the number of transistors**:

25=32

So, according to Moore's Law, the number of transistors on a chip in 2025 should be **32 times** the number in 2015.

b.

Given:

Performance increases by 52% every year

Time span from 2015 to 2025 is 10 years.

Annual growth rate is 1.52

Total performance increase over 10 years is 1.5210

1.5210≈57.67

c.

1) Power and Thermal Constraints, Diminishing Returns

2) Multicore Architectures, Larger Caches, Energy Efficiency Improvements

**9.** General-purpose processes are optimized for general-purpose computing. That is, they are optimized for behavior that is generally found across a large number of applications. However, once the domain is restricted somewhat, the behavior that is found across a large number of the target applications may be different from general-purpose applications. One such application is deep learning or neural networks. Deep learning can be applied to many different applications, but the fundamental building block of inference—using the learned information to make decisions— is the same across them all. Inference operations are largely parallel, so they are currently performed on graphics processing units, which are specialized more toward this type of computation, and not to inference in particular. In a quest for more performance per watt, Google has created a custom chip using tensor processing units to accelerate inference operations in deep learning.1 This approach can be used for speech recognition and image recognition, for example. This problem explores the trade-offs between this process, a general-purpose processor (Haswell E5-2699 v3) and a GPU (NVIDIA K80), in terms of performance and cooling. If heat is not removed from the computer efficiently, the fans will blow hot air back onto the computer, not cold air. Note: The differences are more than processor—on-chip memory and DRAM also come into play. Therefore statistics are at a system level, not a chip level.

**a.** If Google’s data center spends 70% of its time on workload A and 30% of its time on workload B when running GPUs, what is the speedup of the TPU system over the GPU system?

**b.** Google’s data center spends 70% of its time on workload A and 30% of its time on workload B when running GPUs, what percentage of Max IPS does it achieve for each of the three systems?

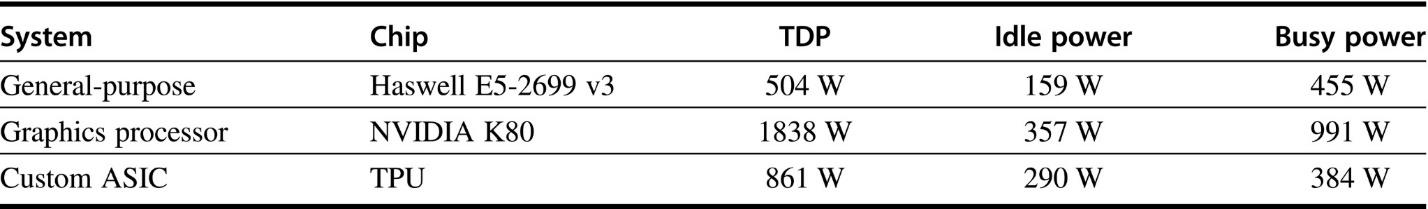
**c.** Building on (b), assuming that the power scales linearly from idle to busy power as IPS grows from 0% to 100%, what is the performance per watt of the TPU system over the GPU system?

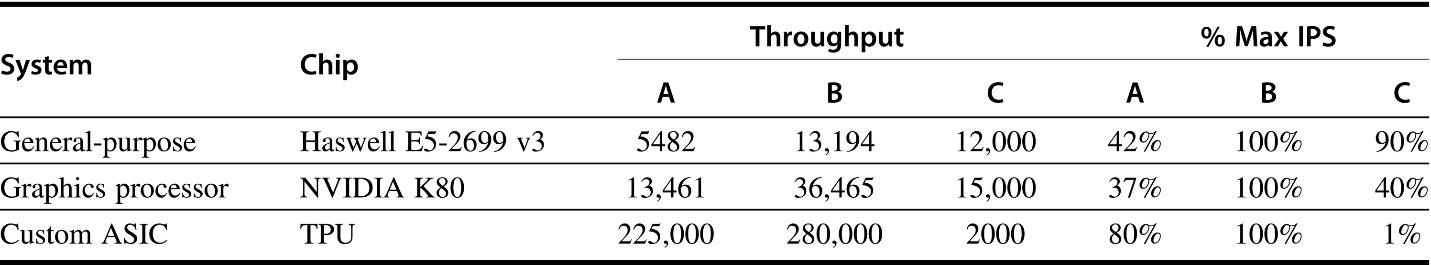
**d.** If another data center spends 40% of its time on workload A, 10% of its time on workload B, and 50% of its time on workload C, what are the speedups of the GPU and TPU systems over the general- purpose system?

**e.** A cooling door for a rack cost $4000 and dissipates 14 kW (into the room; additional cost is required to get it out of the room). How many Haswell-, NVIDIA-, or Tensor-based servers can you cool with one cooling door, assuming TDP in Figures 1.27 and 1.28?

**f.** Typical server farms can dissipate a maximum of 200 W per square foot. Given that a server rack requires 11 square feet (including front and back clearance), how many servers from part (e) can be placed on a single rack, and how many cooling doors are required?

**Figure 1.27** Hardware characteristics for general-purpose processor, graphical processing unit-based or custom ASIC-based system, including measured power





**Figure 1.28** Performance characteristics for general-purpose processor, graphical processing unit-based or custom ASIC-based system on two neural-net workloads

a.

Speedup of TPU over GPU for workload A = 225000/13461 = 16.7

Speedup of TPU over GPU for workload B = 280000/36465 = 7.7

For spending 70% of its time on workload A and 30% of its time on workload B when running GPUs:

0.7 x Ex Time (A+B) of GPU = Ex Time (A) of TPU x 16.7

0.3 x Ex Time (A+B) of GPU = Ex Time (B) of TPU x 7.7

Ex Time (A+B) of TPU = [ 0.7 x Ex Time (A+B) of GPU /16.7 + 0.3 x Ex Time (A+B) of GPU/7.7 ]

= (0.7/16.7 + 0.3/7.7) Ex Time (A+B) GPU

Therefore, the speedup of TPU over GPU = Ex Time (A+B) GPU / Ex Time (A+B) of TPU

= 1/(0.7/16.7 + 0.3/7.7) = 12.36

b.

IPS = Cycles per Second/Cycles per Instruction.

Haswell E5-2699 v3:

According to Figure1.28, The general purpose CPU can accomplish only 42% of its maximum IPS in workload A and 100% of its maximum IPS in workload B. Maximum IPS it can possibly achieve = 42% x 70% + 100% x 30% = 0.594

NVIDIA K80:

According to Figure1.28, The GPU can accomplish 37% of its maximum IPS in workload A and 100% of its maximum IPS in workload B. Maximum IPS it can possibly achieve = 37% x 70% + 100% x 30% = 0.559

Custom ASIC:  
According to Figure1.28, The TPU can accomplish 80% of its maximum IPS in workload A and 100% of its maximum IPS in workload B. Maximum IPS it can possibly achieve = 80% x 70% + 100% x 30% = 0.86

c.

Because of the linear growth, power = Idle power + [Busy power –Idle power] x [% of max IPS]

GPU: 357 W + (991 W-357 W) x0.559= 711.4 W

TPU: 290 W + (384 W-290 W) x0.86= 370.8 W

The performance per watt of the TPU system over the GPU system is:

(% max IPS\_TPU/Power\_TPU) / (% of max IPS\_GPU/Power\_GPU)

=(0.86/370.8)/(0.559/711.4)

=2.95

d.

Same as process in a:

GPU:

Speedup of GPU over CPU for workload A = 13461/5482 = 2.455

Speedup of GPU over CPU for workload B = 36465/13194 = 2.76

Speedup of GPU over CPU for workload C = 15000/12000 = 1.25

For spending 40% of its time on workload A and 10% of its time on workload B and 50% of its time on workload C, when running GPUs:

0.4 x Ex Time (A+B+C) of GPU = Ex Time (A) of CPU x 2.455

0.1 x Ex Time (A+B+C) of GPU = Ex Time (B) of CPU x 2.76

0.5 x Ex Time (A+B+C) of GPU = Ex Time (C) of CPU x 1.25

Ex Time (A+B+C) of CPU = [ 0.4 x Ex Time (A+B+C) of GPU /2.455 + 0.1 x Ex Time (A+B+C) of GPU/2.76 + 0.5 x Ex Time (A+B+C) of GPU/1.25 ]

Therefore, the speedup of GPU over CPU = Ex Time (A+B+C) CPU / Ex Time (A+B+C) of GPU

= 1/(0.4/2.455 + 0.1/2.76 + 0.5/1.25) = 1.669

TPU:

Speedup of TPU over CPU for workload A = 225000/5482 = 41.04

Speedup of TPU over CPU for workload B = 280000/13194 = 21.22

Speedup of TPU over CPU for workload C = 2000/12000 = 0.167

For spending 40% of its time on workload A and 10% of its time on workload B and 50% of its time on workload C, when running TPUs:

0.4 x Ex Time (A+B+C) of TPU = Ex Time (A) of CPU x 41.04

0.1 x Ex Time (A+B+C) of TPU = Ex Time (B) of CPU x 21.22

0.5 x Ex Time (A+B+C) of TPU = Ex Time (C) of CPU x 0.167

Ex Time (A+B+C) of CPU = [ 0.4 x Ex Time (A+B+C) of TPU /41.04 + 0.1 x Ex Time (A+B+C) of TPU/21.22 + 0.5 x Ex Time (A+B+C) of TPU/0.167 ]

Therefore, the speedup of TPU over CPU = Ex Time (A+B+C) CPU / Ex Time (A+B+C) of TPU

= 1/(0.4/41.04 + 0.1/21.22 + 0.5/0.167) = 0.332

e.

The number of CPUs that can be cooled with a cooling door = 14kW/504W = 27

The number of GPUs that can be cooled with a cooling door = 14kW/1838W = 7

The number of TPUs that can be cooled with a cooling door = 14kW/861W = 16

f.

Maximum power per rack in a server farm: 200 W/ft2 x 11 ft2= 2200 W

Maximum number of servers per rack & number of cooling doors for this maximum:

CPU: 2200W / 504W = 4.37; 4 servers

GPU: 2200W / 1838W = 1.20; 1 server

CPU: 2200W / 861W = 2.56; 2 servers

According to the results in (e), one cooling door is enough to dissipate these servers.

**10.** Consider the following two processors. P1 has a clock rate of 4GHz, average CPI of 0.9, and requires the execution of 5.0E9 instructions. P2 has a clock rate of 3GHz, an average CPI of 0.75, and requires the execution of 1.0E9 instructions.

**a**. One usual fallacy is to consider the computer with the largest clock rate as having the highest performance. Check if this is true for P1 and P2.

**b**. Another fallacy is to consider that the processor executing the largest number of instructions will need a larger CPU time. Considering that processor P1 is executing a sequence of 1.0E9 instructions and that the CPI of processors P1 and P2 do not change, determine the number of instructions that P2 can execute in the same time that P1 needs to execute 1.0E9 instructions.

**c.** A common fallacy is to use MIPS (millions of instructions per second) to compare the performance of two different processors, and consider that the processor with the largest MIPS has the largest performance. Check if this is true for P1 and P2.

a.

P1 has a higher clock rate (4 GHz) than P2 (3 GHz). However, P1 takes **1.125 seconds** to execute its instructions, while P2 takes only **0.25 seconds**. The processor with the higher clock rate (P1) does not necessarily have the highest performance. In this case, P2 has a shorter execution time, indicating better performance.

b.

CPU TimeP1=(1.0×109×0.9)/(4×109)= 0.225 seconds

Let the number of instructions be N.

CPU TimeP2=(N×0.75)/(3×109)

Set this equal to P1's CPU time for 1.0E9 instructions:

(N×0.75)/(3×109)=0.225

**So** N=9.0×108 instructions

So, P2 can execute **900 million instructions** in the same time that P1 executes **1 billion instructions**.

**11.** A program runs in 100 seconds on a single-core processor. A new processor improves the performance of a specific task within the program by a factor of 5, but this task only accounts for 30% of the total execution time. Calculate the speedup achieved on the new processor using Amdahl's Law.

Speedup=1/ ((1−0.30) +0.30/5)

Calculate the denominator:

1−0.30=0.70

0.30/5=0.06

Denominator=0.70+0.06=0.76

Speedup=1/0.76≈1.316