Find the word or phrase from the list below that best matches the description in the following questions. Use the numbers to the left of the words in the answer. Each answer should be used only once.

|  |  |
| --- | --- |
| 1. virtual worlds | 14. operating system |
| 2. desktop computers | 15. compiler |
| 3. servers | 16. Bit |
| 4. low-end servers | 17. Instruction |
| 5. supercomputers | 18. assembly language |
| 6. terabyte | 19. machine language |
| 7. petabyte | 20. C |
| 8. datacenters | 21. Assembler |
| 9. embedded computers | 22. high-level language |
| 10. multicore processors | 23. system software |
| 11. VHDL | 24. application software |
| 12. RAM | 25. COBOL |
| 13. CPU | 26. FORTRAN |

1.1.1 Computer used to run large problems and usually accessed via a network  ==> 3) servers   
1.1.2 10^15 or 2^50 bytes (i.e., 10 to the 15 or 2 to the 50) ==> 7) petabyte   
1.1.3 Computer composed of hundreds to thousands of processors and terabytes    
of memory ==> 5) supercomputers   
1.1.4 Today's science fiction application that probably will be available in near    
future ==> 1) virtual worlds   
1.1.5 A kind of memory called random access memory==> 12) RAM   
1.1.6 Part of a computer called central processor unit ==> 13) CPU   
1.1.7 Thousands of processors forming a large cluster ==> 8) datacenters   
1.1.8 A microprocessor containing several processors in the same chip==> 10) multi-core processors   
1.1.9 Desktop computer without screen or keyboard usually accessed via a network==> 4) low-end servers   
1.1.10 Currently the largest class of computer (i.e., there are more   
of these types of computers than any others) that runs one application   
or one set of related applications ==> 9) embedded computers   
1.1.11 Special language used to describe hardware components ==> 11) VHDL   
1.1.12 Personal computer delivering good performance to single users at low    
cost ==> 2) desktop computers   
1.1.13 Program that translates statements in high-level language to assembly    
language==> 15) compiler    
1.1.14 Program that translates symbolic instructions to binary instructions==> 21) assembler   
1.1.15 High-level language for business data processing ==> 25) cobol   
1.1.16 Binary language that the processor can understand==> 19) machine language   
1.1.17 Commands that the processors understand ==> 17) instruction   
1.1.18 High-level language for scientific computation ==> 26) fortran   
1.1.19 Symbolic representation of machine instructions ==> 18) assembly language   
1.1.20 Interface between user's program and hardware providing a variety of    
services and supervision functions ==> 14) operating system   
1.1.21 Software/programs developed by the users ==> 24) application software   
1.1.22 Binary digit (value 0 or 1) ==> 16) bit   
1.1.23 Software layer between the application software and the hardware that    
includes the operating system and the compilers ==> 23) system software   
1.1.24 High-level language used to write application and system software==> 20) C   
1.1.25 Portable language composed of words and algebraic expressions that    
must be translated into assembly language before run in a computer ==> 22) high-level language   
1.1.26 10^12 or 2^40 bytes==> 6) terabyte

Exercise 1.2

1.2.1 For a color display using 8 bits for each of the primary colors (red, green, blue) per pixel, what should be the minimum size in bytes of the frame buffer to store a frame?

8 bits × 3 colors = 24 bits/pixel => 4 bytes/pixel.

1280 × 800 pixels = 1,024,000 pixels.

1,024,000 pixels × 4 bytes/pixel = 4,096,000 bytes (approxitly 4 Mbytes).

1.2. 2 How many frames could it store, assuming the memory contains no other information?

2 GB = 2000 Mbytes

Number of frames = 2000 Mbytes/4 Mbytes = 500 frames

1.2.3 If a 256 Kbytes file is sent through the Ethernet connection, how long it would take?

1 gigabit network ==> 1 gigabit/per second = 125 Mbytes/second.

File size: 256 Kbytes = 0.256 Mbytes.

Time= Mbytes/network speed: 0.256/125 = 2.048 ms.

Exercise 1.3

1.3.1 Which processor has the highest performance expressed in instructions per second?

P2 has the highest performance

P1 (Performance = instructions/sec) = 2 × 10^9/1.5 = 1.33 × 10^9

P2 (Performance = instructions/sec) = 1.5 × 10^9/1.0 = 1.5 × 10^9

P3 (Performance = instructions/sec) = 3 × 10^9/2.5 = 1.2 × 10^9

1.3.2 If the processors each execute a program in 10 seconds, find the number of cycles and the number of instructions.

Number of cycles = time × clock rate

Cycles (P1) = 10 × (2 × 10^9) = 20 × 10^9 s

Cycles (P2) = 10 × (1.5 × 10^9) = 15 × 10^9 s

Cycles (P3) = 10 × (3 × 10^9) = 30 × 10^9 s

Time = (Number of instructions × CPI)/clock rate; Number of instructions = Number of cycles/CPI

Instructions (P1) = 20 × 10^9/1.5 = 13.33 × 10^9

Instructions (P2) = 15 × 10^9/1 = 15 × 10^9

Instructions (P3) = 30 × 10^9/2.5 = 12 × 10^9

1.3.3 We are trying to reduce the 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?

Timenew = Timeold × 0.7 = 7 s

CPInew = CPIold × 1.2 => CPI (P1) = 1.8, CPI (P2) = 1.2, CPI (P3) = 3

Time = Number of instructions × CPI/clock rate

Time (P1) = 13.33 × 10^9 × 1.8/7 = 3.42 GHz

Time (P2) = 15 × 10^9 × 1.2/7 = 2.57 GHz

Time (P3) = 12 × 10^9 × 3/7 = 5.14 GHz

1.3.4 Find the IPC (instructions per cycle) for each processor.

IPCnew = 1/CPIold = Number of instructions/ (time × clock rate)

IPC (P1) = (20 x 10^9)/(7 x 3) = 1.42

IPC (P2) = (30 x 10^9)/(10 x 2.5) = 2

IPC (P3) = (90 x 10^9)/(9 x 4) = 3.33

1.3.5 Find the clock rate for P2 that reduces its execution time to that of P1.

Timenew/Timeold = 7/10 = 0.7.

Clock ratenew = Clock rateold/0.7 = 1.5 GHz/0.7 = 2.14 GHz

1.3.6 Find the number of instructions for P2 that reduces its execution time to that of P3.

Timenew/Timeold = 9/10 = 0.9.

Instructionsnew = Instructionsold × 0.9 = (30 × 10^9)× 0.9 = 27 × 10^9

Exercise 1.4

1.4.1 Given a program with 106 instructions divided into classes as follows: 10% class A, 20% class B, 50% class C and 20% class D, which implementation is faster?

P2 has a faster implementation time

Class A: 10^5 instr.

Class B: 2 × 10^5 instr.

Class C: 5 × 10^5 instr.

Class D: 2 × 10^5 instr.

Time = Number of instructions × CPI/clock rate

P1: Time class A = 0.66 × 10^−4

Time class B = 2.66 × 10^−4

Time class C = 10 × 10^−4

Time class D = 5.33 × 10^−4

Total time P1 = 18.65 × 10^−4

P2: Time class A = 10^−4

Time class B = 2 × 10^−4

Time class C = 5 × 10^−4

Time class D = 3 × 10^−4

Total time P2 = 11 × 10^−4

1.4.2 What is the global (i.e., average) CPI for each implementation?

CPI = time × clock rate/ number of instructions

CPI (P1) = (18.65 × 10^−4) × (1.5 × 10^9)/10^6= 2.79

CPI (P2) = (11 × 10^−4) × (2 × 10^9)/10^6 = 2.2

1.4.3 Find the clock cycles required in both cases.

Clock cycles (P1) = (10^5 × 1) + ((2 × 10^5) × 2) + ((5 × 10^5) × 3) + ((2 × 10^5) × 4) = 28 × 10^5

Clock cycles (P2) = (10^5 × 2) + ((2 × 10^5) × 2) + ((5 × 10^5) × 2) + ((2 × 10^5) × 3) = 22 × 10^5

1.4.4 Assuming that arithmetic instructions take 1 cycle, load and store 5 cycles and branch 2 cycles, what is the execution time of the program in a 2 GHz processor?

(500 × 1) + (50 × 5) + (100 × 5) + (50 × 2) × (0.5 × 10^–9) = 675 ns

1.4.5 Find the CPI for the program.

CPI = time × clock rate/ Number of instructions

CPI = ((675 × 10^–9) × (2 × 10^9))/700 = 1.92

1.4.6 If the number of load instructions can be reduced by one-half, what is the speed-up and the CPI?

Time = ((500 × 1) + (50 × 5) + (50 × 5) + (50 × 2)) × (0.5 × 10^–9) = 550 ns

Speed-up = 675 ns/550 ns = 1.22

CPI = ((550 × 10^–9) × (2 × 10^9))/700 = 1.57

Exercise 1.5

1.5.1 Assume that peak performance is defined as the fastest rate that a computer can execute any instruction sequence. What are the peak performances of P1 and P2 expressed in instructions per second?

1.5.2 If the number of instructions executed in a certain program is divided equally among the classes of instructions except for class A, which occurs twice as often as each of the others. Which computer is faster? How much faster is it?

1.5.3 If the number of instructions executed in a certain program is divided equally among the classes of instructions except for class E, which occurs twice as often as each of the others? Which computer is faster? How much faster is it?

1.5.4 Assuming that computes take 1 cycle, loads and store instructions take 10 cycles, and branches take 3 cycles, find the execution time of the program on a 3 GHz MIPS processor.

1.5.6 Assuming that computes take 1 cycle, loads and store instructions take 2 cycles, and branches take 3 cycles, what is the speed-up of a program if the number of compute instruction can be reduced by one-half?

Exercise 1.6

1.6.1 For the same program, two different compilers are used. The table above shows the execution time of the compiled program. Find the average CPI for the program given that the processor has a clock cycle time of 1 nS.

1.6.2 Assume the average CPI found in 1.6.1, but that the compiled program runs on two difference processors. If the execution times on the two processors are the same, how much faster is the clock of the processor running compiler A’s code versus the clock of the processor running compiler B’s code?

1.6.3 A new compiler is developed that uses only 600 million instructions and has an average CPI of 1.1. What is the speed-up of using this new compiler versus using Compiler A or B on the original processor of 1.6.1?

1.6.4 Assume that peak performance is defined as the fastest rate that a computer can execute any instruction sequence. What is the peak performance of P1 and P2 expressed in instructions per second?

1.6.5 If the number of instructions executed in a certain program is divided equally among the classes of instructions in Problem 2.36.4 except for class A, which occurs twice as often as each of the others, how much faster is P2 than P1?

1.6.6 At what frequency does P2 have the same performance as P1 for the instruction mix given in 1.6.5?

Exercise 1.7

1.7.1 What is the geometric mean of the ratios between consecutive generations for both clock rate and power? (The geometric mean is described in Section 1.7.)

Geometric mean clock rate ratio = (1.28 × 1.56 × 2.64 × 3.03 × 10.00 × 1.80 × 0.74)1/7 = 2.15

Geometric mean power ratio = (1.24 × 1.20 × 2.06 × 2.88 × 2.59 × 1.37 × 0.92)1/7 = 1.62

1.7.2 What is the largest relative change in clock rate and power between generations?

Largest clock rate ratio = 2000 MHz/200 MHz = 10

Largest power ratio = 29.1 W/10.1 W = 2.88

1.7.3 How much larger is the clock rate and power of the last generation with respect to the first generation?

Clock rate: 2.667 × 10^9/12.5 × 106 = 212.8

Power: 95 W/3.3 W = 28.78

1.7.4 Find the average capacitive loads, assuming a negligible static power consumption.

Capacitive = Power/Voltage^2 × clock rate

80286: C = 0.0105 × 10^−6

80386: C = 0.01025 × 10^−6

80486: C = 0.00784 × 10^−6

Pentium: C = 0.00612 × 10^−6

Pentium Pro: C = 0.0133 × 10^−6

Pentium 4 Willamette: C = 0.0122 × 10^−6

Pentium 4 Prescott: C = 0.00183 × 10^−6

Core 2: C = 0.0294 × 10^−6

1.7.5 Find the largest relative change in voltage between generations.

Pentium Pro/Pentium 4 Willamette = 3.3/1.75 = 1.78

1.7.6 Find the geon1etric mean of the voltage ratios in the generations since the Pentium.

Pentium Pro / Pentium => 3.3/5 = 0.66

Pentium 4 Willamette/ Pentium Pro => 1.75/3.3 = 0.53

Pentium 4 Prescott / Pentium 4 Willamette => 1.25/1.75 = 0.71

Core 2 / Pentium 4 Prescott => 1.1/1.25 = 0.88

Geometric mean = 0.68

Exercise 1.8

1.8.1 How much has the capacitive load been reduced between versions if the dynamic power has been reduced by 10%?

Power1 = V^2× clock rate × Capacitive; Power2 = 0.9 Power1

C2/C1 = 0.9 × 5^2 × 0.5 × 10^9/3.3^2 × 1 × 10^9 = 1.03

1.8.2 By how much has the dynan1ic power been reduced if the capacitive load does not change?

Power2/Power1 = V22 × clock rate2/V12 × clock rate1

Power2/Power1 = 0.87 => Reduction of 13%

1.8.3 Assuming that the capacitive load of version 2 is 80% the capacitive load of version 1, find the voltage for version 2 if the dynamic power of version 2 is reduced by 40% from version 1.

Power2 = V22 × 1 × 10^9 × 0.8 × C1 = 0.6 × Power1

Power1 = 52 × 0.5 × 10^9 × C1

V22 × 1 × 10^9 × 0.8 × C1 = 0.6 × 52 × 0.5 × 10^9 × C1

V2 = ((0.6 × 52 × 0.5 × 10^9)/(1 × 10^9 × 0.8))1/2 = 3.06 V

1.8.4 By what factor does the dynamic power scales?

Powernew = 1 × Cold × V2old/(2−1/4)2× clock rate × 21/2 = Powerold

According to this the power scales by 1.

1.8.5 Find the scaling of the capacitance per unit area.

1/2−1/2 = 21/2

1.8.6 Using data from Exercise 1.7, find the voltage and clock rate of the Core 2 processor for the next process generation.

Voltage = 1.1 × 1/2−1/4 = 0.92 V

Clock rate = 2.667 × 21/2 = 3.771 GHz

Exercise 1.9

1.9.1 Find the percentage of the total dissipated power comprised by static power.

1.9.2 If the total dissipated power is reduced by 10% while maintaining the static to total power rate of problem 1.9.1, how much should the voltage be reduced to maintain the same leakage current?

1.9.3 Determine the ratio of static power to dynamic power for each technology.

1.9.4 Determine the static power for each version at 0.8 V, assuming a static to dynamic power ratio of 0.6.

Powerst/Powerdyn = 0.6 => Powerst = 0.6 × Powerdyn

1. Powerst = 0.6 × 35 W = 21 W
2. Powerst = 0.6 × 30 W = 18 W

1.9.5 Determine the static power and dynamic power dissipation assuming the rates obtained in problem 1.9.1.

Llk = Voltage / Powerst

a) Ilk = 21/0.8 = 26.25 A

b) Ilk = 18/0.8 = 22.5 A

Exercise 1.10

1.10.1 The table above shows the number of instructions required per processor to complete a program on a multiprocessor with 1, 2, 4, or 8 processors. What is the total number of instructions executed per processor? What is the aggregate number of instructions executed across all processors?

Instructions per processor = (ArithInst x ArithCpi)+( loadInst x loadCpi)+( branchInst x branchCpi)

Total Instructions = Instructions per processor \* processors

|  |  |  |  |
| --- | --- | --- | --- |
|  | Processors | Instructions per processor | Total Instructions |
| a | 1 | 4096 | 4096 |
| 2 | 2048 | 4096 |
| 4 | 1024 | 4096 |
| 8 | 512 | 4096 |
| b | 1 | 4096 | 4096 |
| 2 | 2278 | 4556 |
| 4 | 1464 | 5856 |
| 8 | 1132 | 9056 |

1.10.2 Given the CPI values on the right of the table above, find the total execution time for this program on I, 2, 4, and 8 processors. Assume that each processor has a 2 GHz clock frequency.

1.10.3 If the CPI of arithmetic instructions was doubled, what would the impact be on the execution time of the program on 1,2,4, or 8 processors?

1.10.4 Assuming a 3 GHz clock frequency, what is the execution time of the program using 1, 2, 4, or 8 cores.

1.10.5 Assuming that the power consumption of a processor core can be described by the following equation:

Power = (5.0n1A)/(MHz)\* Voltage^2

Where the operation voltage of the processor is described by the following equation:

Voltage = 1/5\*Frequency+0.4

With the frequency measured in GHz. So, at 5 GHz, the voltage would be 1.4V. Find the power consumption ofthe program executing on 1, 2, 4, and 8 cores assuming that each core is operating at a 3 GHz clock frequency. Likewise, find the power consumption of the program executing on I, 2, 4, or 8 cores assuming that each core is operating at 500 MHz.

Exercise 1.11

1.11.1 Find the yield.

Wafer area = π × (d/2)2

a. Wafer area = π × 7.52 = 176.7 cm2

b. Wafer area = π × 12.52 = 490.9 cm2

Die area = wafer area/dies per wafer

a. Die area = 176.7/90 = 1.96 cm2

b. Die area = 490.9/140 = 3.51 cm2

Yield = 1/ (1 + (defect per area × die area)/2)2

a. Yield = 0.97

b. Yield = 0.92

1.11.2 Find the cost per die.

Cost per die = cost per wafer/ (dies per wafer × yield)

a. Cost per die = 0.12

b. Cost per die = 0.16

1.11.3 If the number of dies per wafer is increased by 100/0 and the defects per area unit increases by 150/0, find the die area and yield.

a. Dies per wafer = 1.1 × 90 = 99

Defects per area = 1.15 × 0.018 = 0.021 defects/cm2

Die area = wafer area/Dies per wafer = 176.7/99 = 1.78 cm2

Yield = 0.97

b. Dies per wafer = 1.1 × 140 = 154

Defects per area = 1.15 × 0.024 = 0.028 defects/cm2

Die area = wafer area/Dies per wafer = 490.9/154 = 3.19 cm2

Yield = 0.93

1.11.4 Find the defects per area unit for each technology given a die area of 200 mm2

Yield = 1/(1 + (defect per area × die area)/2)2

Defect per area = (2/die area) (y−1/2− 1)

Replacing values for T1 and T2 we get

T1: defects per area = 0.00085 defects/mm2 = 0.085 defects/cm2

T2: defects per area = 0.00060 defects/mm2 = 0.060 defects/cm2

T3: defects per area = 0.00043 defects/mm2 = 0.043 defects/cm2

T4: defects per area = 0.00026 defects/mm2 = 0.026 defects/cm2

Exercise 1.12

1.12.1 Find the CPI if the clock cycle time is 0.333 ns.

CPI = clock rate × CPU time/instr. count

Clock rate = 1/cycle time = 3 GHz

a. CPI (pearl) = 3 × 10^9 × 500/2118 × 10^9 = 0.7

b. CPI (mcf) = 3 × 10^9 × 1200/336 × 10^9 = 10.7

1.12.2 Find the SPEC ratio.

SPECratio = ref. time/execution time.

a. SPECratio (pearl) = 9770/500 = 19.54

b. SPECratio (mcf) = 9120/1200 = 7.6

1.12.3 For these two benchmarks, find the geometric mean.

(19.54 × 7.6)1/2 = 12.19

1.12.4 Find the increase in CPU time if the number of instruction of the benchmark is increased by 100/0 without affecting the CPI.

CPU time = number of instructions × CPI/clock rate

If CPI and clock rate do not change, then the CPU time will increase equal to the number of instructions, which would be by 10%.

1.12.5 Find the increase in CPU time if the number of instruction of the benchmark is increased by 10% and the CPI is increased by 5%.

CPU time (before) = number of instructions × CPI/clock rate

CPU time (after) = 1.1 × number of instructions × 1.05 × CPI/clock rate

CPU times (after)/CPU time (before) = 1.1 × 1.05 = 1.155.

CPU time is increased by 15.5%

1.12.6 Find the change in the SPECratio for the change described in 1.12.5.

SPECratio = reference time/CPU time

SPECratio (after)/SPECratio (before) = CPU time (before)/CPU time (after) = 1/1.1555 = 0.86

The SPECratio is decreased by 14%.

Exercise 1.13

1.13.1 Find the new CPI.

CPI = (CPU × clock rate)/Number of instr.

a. CPI = 450 × 4 × 10^9/ (0.85 × 2118 × 10^9) = 0.99

b. CPI = 1150 × 4 × 10^9/ (0.85 × 336 × 10^9) = 16.10

1.13.2 In general, these CPI values are larger than those obtained in previous exercises for the same benchmarks. This is due mainly to the clock rate used in both cases, 3 GHz and 4 GHz. Determine whether the increase in the CPI is similar to that of the clock rate. If they are dissimilar, why?

Clock rate ratio = 4 GHz/3 GHz = 1.33.

a. CPI at 4 GHz = 0.99, CPI at 3 GHz = 0.7, ratio = 1.41

b. CPI at 4 GHz = 16.1, CPI at 3 GHz = 10.7, ratio = 1.50

They are different because the CPU time has been reduced by a lower percentage, even though the number of instructions has been reduced by 15%,

1.13.3 How much has the CPU time been reduced?

a. 450/500 = 0.90

CPU time reduction = 10%

b. 1150/1200 = 0.958

CPU time reduction = 4.2%

1.13.4 If the execution time is reduced by an additional 10% without affecting the CPI and with a clock rate of 4 GHz, determine the number of instructions.

Number of instructions = CPU × clock rate/CPI.

a. Number of instructions = 820 × 0.9 × 4 × 10^9/0.96 = 3075 × 10^9

b. Number of instructions = 580 × 0.9 × 4 × 10^9/2.94 = 710 × 10^9

1.13.5 Determine the clock rate required to give a further 10% reduction in CPU time while maintaining the nun1ber of instructions and CPI unchanged.

Clock rate = Number of instructions × CPI/CPU time.

Clock ratenew = Number of instructions × CPI/0.9 × CPU time = 1/0.9 clock rateold = 3.33 GHz.

1.13.6 Determine the clock rate if the CPI is reduced by 15% and the CPU time by 20% while the number of instructions is unchanged.

Clock rate = Number of instructions × CPI/CPU time.

Clock ratenew = Number of instructions × 0.85 × CPI/0.80 CPU time = 0.85/0.80 clock rateold = 3.18 GHz.

Exercise 1.14

1.14.1 One usual fallacy is to consider the computer with the largest clock rate as having the large perfonl1ance. Check if this is true for PI and P2.

Number of instructions = 106

Tcpu (P1) = 106 × 1.25/4 × 10^9 = 0.315 × 10–3 s

Tcpu (P2) = 106 × 0.75/3 × 10^9 = 0.25 × 10–3 s

Clock rate (P1) > clock rate (P2), but performance (P1) < performance (P2)

1.14.2 Another fallacy is to consider that the processor executing the largest number of instruction will need a larger CPU time. Considering that processor PI is executing a sequence of 106 instructions and that the CPI of processors PI and P2 do not change, detenl1ine the number of instructions that P2 can execute in the same time that PI needs to execute 106 instructions.

P1: 106 instructions, Tcpu (P1) = 0.315 × 10–3 s

P2: Tcpu (P2) = N × 0.75/3 × 109

N = 1.26 × 106

1.14.3 A common fallacy is to use MIPS (millions of instructions per second) to con1pare the performance of two different processors, and consider that the processor with the largest MIPS has the largest perforn1ance. Check if this is true for PI and P2.

MIPS = Clock rate × 10−6 /CPI

MIPS (P1) = 4 × 10^9 × 10–6/1.25 = 3200

MIPS (P2) = 3 × 10^9 × 10–6/0.75 = 4000

MIPS (P1) < MIPS (P2), performance (P1) < performance (P2) in this case

Another common performance figure is MFLOPS (millions of floating-point operations per second), defined as

MFLOPS = Number of FP operations/(execution time x 10^6)

1.14.4 Find the MFLOPS figures for the programs.

a. FP op = 106 × 0.4 = 4 × 105, clock cylesfp = CPI × Number of FP instr. = 4 × 105

Tfp = 4 × 105 × 0.33 × 10–9 = 1.32 × 10–4 then MFLOPS = 3.03 × 103

b. FP op = 3 × 106 × 0.4 = 1.2 × 106, clock cylesfp = CPI × Number of FP instr. = 0.70 × 1.2 × 106

Tfp = 0.84 × 106 × 0.33 × 10–9 = 2.77 × 10–4 then MFLOPS = 4.33 × 103

1.14.5 Find the MIPS figures for the programs.

5 CPU clock cycles = FP cycles + CPI (L/S) × Number of instr. (L/S) + CPI (Branch) ×

Number of instr. (Branch)

a. 5 × 105 L/S instr., 4 × 105 FP instr. and 105 Branch instr.

CPU clock cycles = 4 × 105 + 0.75 × 5 × 105 + 1.5 × 105 = 9.25 × 105

Tcpu = 9.25 × 105 × 0.33 × 10–9 = 3.05 × 10–4

MIPS = 106/ (3.05 × 10–4 × 106) = 3.2 × 103

b. 1.2 × 106 L/S instr., 1.2 × 106 FP instr. and 0.6 × 106 Branch instr.

CPU clock cycles = 0.84 × 106 + 1.25 × 1.2 × 106 + 1.25 × 0.6 × 106 = 3.09 × 106

Tcpu = 3.09 × 106 × 0.33 × 10–9 = 1.01 × 10–3

MIPS = 3 × 106/ (1.01 × 10–3 × 106) = 2.97 × 103

1.14.6 Find the performance for the programs and con1pare with MIPS and MFLOPS.

a. Performance = 1/Tcpu = 3.2 × 103

b. Performance = 1/Tcpu = 9.9 × 102

The second program has the higher performance, but the ﬁrst program has the higher MIPS ﬁgure.

Exercise 1.15

1.15.1 By how much is the total time reduced if the time for FP operations is reduced by 20%?

a. Tfp = 35 × 0.8 = 28 s; Tp1 = 28 + 85 + 50 + 30 = 193 s; Reduction of 3.5%

b. Tfp = 50 × 0.8 = 40 s; Tp4 = 40 + 80 + 50 + 30 = 200 s; Reduction of 4.7%

1.15.3 Can the total time be reduced by 20% by reducing only the time for branch instructions?

a. Tp1 = 200 × 0.8 = 160 s; Tfp + Tint + Tl/s = 170 s. No the total time cannot be reduced by 20%

b. Tp4 = 210 × 0.8 = 168 s; Tfp + Tint + Tl/s = 180 s. No the total time cannot be reduced by 20%

Assume that each processor has a 2 GHz clock rate.

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

Clock cyles = CPIfp × Number of FP instr. + CPIint × Number of INT instr. + CPIl/s × Number of L/S instr. + CPIbranch × Number of branch instr.

Tcpu = clock cycles/clock rate = clock cycles/2 × 109

a. 1 processor=> clock cycles = 8192; Tcpu = 4.096 s

b. 8 processors=> clock cycles = 1024; Tcpu = 0.512 s

To half the number of clock cycles by improving the CPI of FP instructions:

CPIimproved fp × Number of FP instr. + CPIint × Number of INT instr. + CPIl/s × Number of L/S instr. +

CPIbranch × Number of branch instr. = clock cycles/2

CPIimproved fp = (clock cycles/2 − (CPIint × Number of INT instr. + CPIl/s × Number of L/S instr. +

CPIbranch × Number of branch instr.))/Number of FP instr.

a. 1 processor: CPIimproved fp = (4096 – 7632)/560 => not possible

b. 8 processors: CPIimproved fp = (512 – 944)/80 => not possible

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

Using clock cycle from 1.15.4:

To half the number of clock cycles improving the CPI of L/S instructions:

CPIfp × Number of FP instr. + CPIint × Number of INT instr. + CPIimproved l/s × Number of L/S instr. +

CPIbranch × Number of branch instr. = clock cycles/2

CPIimproved l/s = (clock cycles/2 − (CPIfp × Number of FP instr. + CPIint × Number of INT instr. +

CPIbranch × Number of branch instr.))/Number of L/S instr.

a. 1 processor: CPIimproved l/s = (4096 – 3072)/1280 = 0.8

b. 8 processors: CPIimproved l/s = (512 – 384)/160 = 0.8

1.15.6 By how much is the execution time of the program improved if the CPI of INT and FP instruction is reduced by 40% and the CPI of L/S and branch is reduced by 30%?

Clock cyles = CPIfp× Number of FP instr. + CPIint× Number of INT instr. + CPIl/s× Number of L/S instr. +

CPIbranch × Number of branch instr.

Tcpu = clock cycles/clock rate = clock cycles/2 × 109

CPIint = 0.6 × 1 = 0.6; CPIfp = 0.6 × 1 = 0.6; CPIl/s = 0.7 × 4 = 2.8; CPIbranch = 0.7 × 2 = 1.4

a. 1 processor => Tcpu (before improved) = 4.096 s; Tcpu (after improved) = 2.739 s

b. 8 processors =>Tcpu (before improved) = 0.512 s; Tcpu (after improved) = 0.342 s