The University of Alabama in Huntsville
ECE Department
CPE 431 01, CPE 531 01/01R Fall 2022

Technology Trends and Performance Due August 30, 2022

You must show your work to get full credit. The number in parentheses is the point value of the problem.

1.0(5), 2.0(20), 3.0.0(5), 3.0.1(10), 4.0.0(5), 4.0.1(10), 4.0.1(10), 5.0(5), 6.0.0(10), 6.0.1(5), 7.0(5)

***Answers provided in bold by each problem in case its hard to see in picture (except #7 since it can be seen easily in picture). Work provided below problems starting on Page 3.**

1.0 The seven great ideas in computer architecture are similar to ideas from other fields. Match the seven ideas from computer architecture, "Use Abstraction to Simplify Design", "Make the Common Case Fast", "Performance via Parallelism", "Performance via Pipelining", "Performance via Prediction", "Hierarchy of Memories", and "Dependability via Redundancy" to the following ideas from other fields:

a. Assembly lines in automobile manufacturing **Performance via Pipelining**
b. Suspension bridge cables **Dependability via Redundancy**
c. Aircraft and marine navigation systems that incorporate wind information **Performance via Prediction**
d. Express elevators in buildings **Common Case Fast**
e. Library reserve desk **Hierarchy of Memories**
f. Increasing the gate area on a CMOS transistor to decrease its switching time **Performance via Parallelism**
g. Building self-driving cars whose control systems partially rely on existing sensor systems already installed into the base vehicle, such as lane departure systems and smart cruise control systems **Use Abstraction to simplify design**

2.0 Consider two different implementations of the same instruction set architecture. The instructions can be divided into four classes of instructions according to their CPI (class A, B, C, and D). P1 with a clock rate of 3.3 GHz and CPIS of 1, 2, 5, and 3, and P2 with a clock rate of 3.1 GHz and CPIS of 1, 4, 2, and 3.

Given a program with a dynamic instruction count of 1.0E6 instructions divided into classes as follows: 15% class A, 25% class B, 40% class C, and 20% class D:

a. What is the global CPI for each implementation? **P1: 3.25, P2: 2.55**
b. Find the clock cycles required in both cases. **P1: 3250000, P2: 2550000**
c. Which implementation is faster and by how much? **P1 is faster by 1.275**

3.0 Assume for arithmetic, load/store, and branch instructions, a processor has CPIs of 1, 8, and 4, 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 3 GHz clock frequency.

Assume that, as the program is parallelized to run over multiple cores, the number of arithmetic instructions per processor is divided by 0.75 x p (where p is the number of processors) but the number of branch and load/store instructions per processor remains the same.

3.0.0 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 processor result relative to the single processor result. **See work provided**

3.0.1 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 value? **CPI should be 6.66 to match the performance of 4 processors.**

4.0 Section 1.11 cites as a pitfall the utilization of a subset of the performance equation as a performance metric. To illustrate this, consider the following two processors. P1 has a clock rate of 3.4 GHz, average CPI of 1.7, and requires the execution of 1.8 E11 instructions. P2 has a clock rate of 2.9 GHz, an average CPI of 0.8, and requires the execution of 0.7 E11 instructions.

4.0.0 One usual fallacy is to consider the computer with the largest clock rate as having the largest performance. Check if this is true for P1 and P2. **The fallacy checks out because P1 has the higher CR but a slower ET than P2 by 0.215.**

4.0.1 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. **1.8125 E9 instructions**

4.0.2 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. **P2 has a higher MIPS but we know that P1 has a faster ET. So the fallacy checks out.**

5.0 Another pitfall cited in Section 1.11 is expecting to improve the overall performance of a computer by improving only one aspect of the computer. Consider a computer running a program that requires 320 s, with 70 s spent executing FP instructions, 85 s executing L/S instructions, and 40 s spent executing branch instructions. Can the total time be reduced by 20 % by reducing only the time for FP instructions? **No you can't reduce total time by 20% only reducing FP instructions as that only improves the total time by 4.4%**

6.0 Assume a program requires the execution of 50 E6 FP instructions, 110 E6 INT instructions, 80 E6 L/S instructions, and 16 E6 branch instructions. The CPI for each type of instruction is 10, 2, 3, and 5, respectively. Assume that the processor has a 6 GHz clock rate.

6.0.0 By how much must we improve the CPI of L/S instructions if we want the program to run two times faster? **Need to improve CPI by 5 times**

6.0.1 By how much is the execution time improved if the CPI of INT and FP instructions is reduced by 45% and the CPI of L/S and Branch is reduced by 35%? **1.72 time faster**

7.0 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 = 250 s of execution time on one processor. When run on p processors, each processor requires t/p s, as well as an additional 5 s of overhead, irrespective of the number of processors. Compute the per-processor execution time for 2, 4, 8, 16, 32, 64, and 128 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). **See work provided**

Work

Sorry I write so light so some things may be hard to see.

1 a Performance via Pipeling
b Performance via Parallelism
c Performance via Prediction
d Common Case Fast
e Hierachy of Memories
f Dependability via Redundancy
g Use Abstraction to simplify design

Swap

2. ref pg 37 A B C D

$CR_1 = 3.3 \frac{cycles}{time}$, CPIs : 1, 2, 5, 3

$CR_2 = 3.3 \frac{cycles}{time}$, CPIs : 1, 4, 2, 3

$IC = 1 \times 10^6$

A = 150000 C = 400000

B = 250000 D = 200000

cycles = CPI x IC

b.

	A	B	C	D	
P1	(1)(150000)	2(250000)	5(400000)	3(200000) =	3250000
P2	(1)(150000)	4(250000)	2(400,0000)	3(200000) =	2550000

a. $\frac{cycles}{IC}$

$CPI_1 = \frac{3250000}{1 \times 10^6} = 3.25$

$CPI_2 = \frac{2550000}{1 \times 10^6} = 2.55$

c $\frac{PA_1}{PA_2} = \frac{ET_2}{ET_1} = \frac{\frac{cc}{CR}}{\frac{cc}{CR}}$ $\frac{3250000 \text{ cycles}}{3.3 \text{ cycles/time}}$ $= 1.275$

$\frac{2550000}{3.3}$

P1 is the faster implementation by 1.275

3 cycles/IC arith lw/sw branch

CPIs 1 8 4

Instructions 2.56×10^9 (arith) 1.28×10^9 (lw/sw) 256 m

instructions/processor $0.75 \times p$

CR $= 3 \times 10^9$ cycles/time

$ET = \frac{CC}{CR} \Rightarrow CC = IC \times CPI$

$ET = \frac{(IC_A \times CPI_A) + (IC_{l/s} \times CPI_{l/s}) + (IC_B \times CPI_B)}{CR}$

$ET = \frac{(2.56 \times 10^9 * 1) + (1.28 \times 10^9 * 8) + (256 \times 10^6 * 4)}{3 \times 10^9}$ $\quad IC\left(\frac{\text{cycles}}{s}\right) \cdot \frac{s}{\text{cycles}}$

$= 4.608$ 1 processor

$ET = \frac{2.56 \times 10^9 * 1}{0.75(2)} + \frac{1.28 \times 10^9 * 8}{3 \times 10^9} + \frac{256 \times 10^6 * 4}{3 \times 10^9} =$

$= 4.324$ 2 processor

Speedup $= \frac{1p}{2p} = \frac{4.608}{4.324} = 1.066$

$ET = \frac{2.56 \times 10^9 * 1}{0.75(4)} + \frac{1.28 \times 10^9 * 8}{3 \times 10^9} + 256 \times 10^6 * 4 = 4.039$ 4 processor

Speedup $\frac{4.608}{4.039} = 1.14$

$ET = \frac{2.56 \times 10^9 * 1}{0.75(8)} + \frac{1.28 \times 10^9 * 8}{3 \times 10^9} + 256 \times 10^6 * 4 = 3.897$ 8 p

Speedup $= \frac{4.608}{3.897} = 1.18$

3.

1 CPI = 1 lw/sw = 8 CPI

ET = 4.039 4 processors

$$ET = \frac{(2.56\times10^9 * 1) + (1.28\times10^9 * CPI_{lw/sw}) + 256\times10^6 * 4}{3\times10^9}$$

$$4.039 = \frac{2.56\times10^9 + 1.28\times10^9 CPI + 1.024\times10^9}{3\times10^9}$$

$$4.039 = \frac{3.584\times10^9 + 1.28\times10^9 CPI}{3\times10^9}$$

$$1.2117\times10^{10} = 3.584\times10^9 + 1.28\times10^9 CPI \Rightarrow \frac{8.533\times10^9}{1.28\times10^9} = CPI$$

CPI = 6.6664

$$\frac{6.6664}{8} = 0.83$$

4 P1 : 3.4×10^9 cycles/s CR , CPI = 1.7 IC = 1.8×10^{11}

P2 : 2.9×10^9 cycles/s CR , CPI = 0.8 IC = 0.7×10^{11}

$$\frac{P_1}{P_2} = \frac{ET_2}{ET_1} = \frac{(0.7\times10^{11})(0.8)/(2.9\times10^9)}{(1.8\times10^{11})(1.7)/(3.4\times10^9)} = \frac{19.31}{90} = 0.215$$

The fallacy is true since P1 has the higher CR but slower ET time and thus is 0.215 slower than P2

1.1 $ET_1 = \frac{(1\times10^9)(1.7)}{3.4\times10^9} = 0.5$

$ET_2 = ET_1 = \frac{IC(0.8)}{2.9\times10^9} \Rightarrow \frac{0.5(2.9\times10^9)}{0.8} = 1.8125\times10^9$ not much more

1.2 $MIPS = \frac{IC}{ET\times10^6} = \frac{CR}{CPI\times10^6}$

$P1 = \frac{(3.4\times10^9)}{1.7\times10^6}$ $P2 = \frac{2.9\times10^9}{0.8\times10^6}$

2000 3625

P2 has higher MIPS but we know that P1 has a higher ET. So the fallacy is true.

5 320s = Total Time

70s : FP IC , 85s L/S IC , 40s Branch

195 = time spent on IC i.e 125 elsewhere

20% of total Time = 256

20% of FP IC = 56

reduction IC time : 181 /w + 125 elsewhere = 306

256 < 306 $\frac{306}{320} = 0.95625$

$1 - 0.95625 = 0.044 \approx 4.4\%$

So no you can't reduce Total time by 20% only doing FP IC

6

	IC	CPI	Reduction
FP	50×10^6	10	27.5×10^6
INT	110×10^6	2	60.5×10^6
L/S	80×10^6	3	52×10^6
Branch	16×10^6	5	10.4×10^6

$6 \times 10^9 = CR$

$ET = (50 \times 10^6 * 10) + (110 \times 10^6 * 2) + (80 \times 10^6 * 3) + (16 \times 10^6 * 5) / 6 \times 10^9$

$500 \times 10^6 \quad 220 \times 10^6 \quad 240 \times 10^6 \quad 80 \times 10^6 \quad / 6 \times 10^9$

$= 0.17\bar{3}$

$\frac{P_0}{P_N} = \frac{ET_N}{E_0} = \frac{ET_N}{0.173} = 2$

$0.34\bar{6} = \frac{800 \times 10^6 + 80 \times 10^6 * CPI}{6 \times 10^9} \Rightarrow 2.08 \times 10^9 = 800 \times 10^6 + 80 \times 10^6 CPI$

$1.28 \times 10^9 = 80 \times 10^6 CPI \qquad \frac{16}{3} \approx 5$

$CPI = 16$ Need to improve by 5 times

6.1 $ET_0 = 0.17\bar{3}$

$ET_N = (27.5 \times 10^6 * 10) + (60.5 \times 10^6 * 2) + (52 \times 10^6 * 3) + (10.4 \times 10^6 * 5) / CR$

$275 \times 10^6 + 121 \times 10^6 + 156 \times 10^6 + 52 \times 10^6 / 6 \times 10^9$

$= 0.100\bar{6}$

$\frac{ET_N}{ET_0} \quad \frac{0.100\bar{6}}{0.17\bar{3}} = 0.58 \Rightarrow \frac{1}{0.58} = 1.72$ times faster

7 $t = 250$ s ET 1 processor

$\frac{t}{p}$ + 5s of overhead

	ET	ET_{Ideal}	ET per processor	Actual: $\frac{ET_1}{ET_N}$	Ideal	ratio: $\frac{actual}{ideal}$
1	250	250				
2	130	125	65	1.9	2	0.95
4	67.5	62.5	16.875	3.70	4	0.925
8	36.25	31.25	4.5	6.897	8	0.862
16	20.625	15.625	1.289	12.12	16	0.7575
32	12.8125	7.8125	0.4	19.5	32	0.61
64	8.90625	3.90625	0.139	28.07	64	0.439
128	6.953125	1.953125	0.05	35.955	128	0.281