Example Solutions for Homework Assignment 1

CS/ECE 6810: Computer Architecture
Sep 12th,2018

Performance Metrics, ISA, Basic Pipelining

100 points

Important Notes:

- Solutions turned in must be your own. Please, mention references (if any) at the end of each question. Please refrain from cheating.
- All solutions must be accompanied by the equations used/logic/intermediate steps. Writing only the final answer will receive **zero** credit.
- All units must be mentioned wherever required.
- Late submissions **(after 11:59 pm on 09/12/2018)** will not be accepted
- All submissions must be in PDF only. Scanned copies of handwritten solutions will not be accepted as a valid submission.
- Question 6 is a bonus question

1. **Performance Optimization.** The table below shows the frequencies and cycle counts for all types of instructions used by program A on a computer system P. We observed that 50% of the executed MULT instructions are followed by an ADD. Therefore, we propose a new processor with a fused-MULT-ADD (FMAD) instruction that executes a merged MULT and ADD in 5 cycles.

	LOAD	STORE	BRANCH	ADD	MULT
Frequency	15%	15%	10%	20%	40%
Cycles	1	1	2	4	5

   i Compute the instructions per cycle (IPC) for the new and old processors. **(10 points)**
   Solution: For the old processor, CPI = 0.15*1 + 0.15*1 + 0.1*2 + 0.2*4 + 0.4*5 = 3.3 cycles **(2 points)**
   IPC = 1/CPI = 1/3.3 = 0.303 instructions **(1.5 points)**

   For the new processor

	LOAD	STORE	BRANCH	ADD	MULT	FMAD
Frequency	18.75%	18.75%	12.5%	0%	25%	25%
Cycles	1	1	2	4	5	5

   (3 points)

   CPI = 0.1875*1 + 0.1875*1 + 0.125*2 + 0*4 + 0.25*5 + 0.25*5 = 3.125 cycles **(2 points)**

   IPC = 1/CPI = 1/3.125 = 0.32 instructions **(1.5 points)**

ii What is the speedup/slowdown gained through the proposed optimization? **(5 points)**
$CPI_{old} = 3.3$
$CPI_{new} = 3.125$
$IC_{new} = 0.8 * IC_{new}$**3 points)**
$CT_{new} = CT_{old}$
$ExecutionTime_{old} = CPI_{old} * CT_{old} * ICold = 3.125 * CT * IC_{old}$
$ExecutionTime_{new} = CPI_{new} * CT_{new} * ICnew = 3.125 * CT * 0.8 * IC_{old} * CT$
$Speedup = ExecutionTime_{old}/ExecutionTime_{new} = 1.32$ **(2 points)**

2. **Execution Time.** A computer system has three instruction classes as shown in the following table.

Class	CPI
A	1
B	2
C	3

We execute two programs (P1 and P2) on this computer system. The following table shows the number of executed instructions for different classes.

Program	A	B	C
P1	200	100	200
P2	400	100	100

i What is the CPI and IPC of P1 and P2? **(10 points)**
$\#instructionsofP1$=200 +100 + 200 = 500 **(1 points)**
$\#instructionsofP1$=400 +100 + 100 = 600 **(1 points)**
$\#RequiredCyclesforP1$=$200 \times 1 + 100 \times 2 + 200 \times 3 = 1000$ **(2 points)**
$\#RequiredCyclesforP2$=$400 \times 1 + 100 \times 2 + 100 \times 3$= 900 **(2 points)**
$CPI(P1)$= $\frac{\#RequiredCyclesforP1}{\#instructionsofP1} = \frac{1000}{500}$ =2 **(1 points)**
$IPC(P1) = \frac{1}{CPI(P1)} = 0.5$ **(1 points)**
$CPI(P2)$= $\frac{\#RequiredCyclesforP2}{\#instructionsofP2} = \frac{900}{600}$ =1.5 (1 points)
$IPC(P2) = \frac{1}{CPI(P2)} = 0.66$ **(1 points)**

ii Suppose that the computer operates at 1GHz clock frequency; compute the execution times of P1 and P2? **(10 points)**
$Exectime = CPI \times \#instruction \times ClockPeriod$
$Exectime(P1)$= $2 \times 500 \times \frac{1}{10^9} = 1\mu Second$ **(4 points (correct equation) +1 points (final result))**
$Exectime(P2)$= $1.5 \times 600 \times \frac{1}{10^9} = 0.9\mu Second$ **(4 points (correct equation) +1 points (final result))**

3. **Power and Energy.** A general purpose processor is operated at a 2GHz clock frequency with a 5V voltage supply. Suppose that the average load capacitance of all internal gates is 12pF. The static current driven from the supply is 5A and the logic gates exhibit an average activity of 0.8 switchings per cycle.

i Compute the static and dynamic power of the processor. **(10 points)** Solution: $P_{static} = V * I = 5 * 5 = 25W$**(2 points)**
Let activity factor be A
$P_{dynamic} = C * V^2 * f * A = 12 * 10^{-12} * (5)^2 * 2 * 10^9 * 0.8 = 0.48W$**(8 points)**

ii If the voltage is scaled down by 50% and the frequency is scaled up by 25%, calculate the percentage increase/decrease in the total system power. **(10 points)**
$V_{new} = 0.5 * V_{old} = 0.5 * 5 = 2.5V$**(2 points)**

$f_{new} = 1.25 * f_{old} = 1.25 * 2GHz = 2.5GHz$**(2 points)**

$P_{static} = 2.5 * 5 = 12.5W$ **(2 points)**

$P_{dynamic} = 12 * 10^{-12} * (2.5)^2 * 2.5 * 10^9 * 0.8 = 0.15W$**(2 points)**

Total Power $= 12.5 + 0.15 = 12.65W$
% Decrease in Power $= (25.48 - 12.65)/25.48 = 50.35\%$ **(2 points)**

iii If the frequency is scaled down by 50% of its initial value and the dynamic power along with the output current remains the same, calculate the percentage increase/decrease in the total system power. **(10 points)**

$P_{dynamic_{old}} = P_{dynamic_{new}}$
As capacitance and activity factor remains the same,
$V_{old}^2 * f_{old} = V_{new}^2 * f_{new}$
$V_{new} = \sqrt{2} * V_{old} = 7.07V$**(3 points)**

$f_{new} = 0.5 * f_{old} = 0.5 * 2GHz = 1GHz$**(1 points)**

$P_{static} = 7.07 * 5 = 35.35W$**(1 point)**

$P_{dynamic} = 0.48W$**(1 point)**

Total Power $= 35.35 + 0.48 = 35.83W$**(1 points)**

% Increase in Power $= (35.83 - 25.48)/25.48 = 40.63\%$ **(3 points)**

4. **Instruction Set Architecture.** Initial values of the architectural registers and memory are given below in the following tables. Compute the effective address and final the result for each of the following instructions. *All instructions are executed serially.* Register value changes are considered when moving from one instruction to another. List the final values of all the registers. **(20 points)**

R0	R1	R2	R3	R4	R5	R6
1000	50	2000	5000	0	0	0

Address	1000	2000	3000	4000	5000
Value	1000	4000	3000	5000	2000

```
LOAD R5, 3000(R0)
ADD R4, @(R5)
SUB R2, R4
LOAD R6, 1000(R0)
ADD R6, R2
SUB R5, R6
ADD R2, R5
ADD R2, R3
```

Solution: 1)LOAD R5, 3000(R0)
Effective Address = 3000(R0) = 3000 + R0 = 3000 + 1000 = 4000 **(1.5 points)**

Value: R5 = Mem[4000] = 5000**(1 point)**

2)ADD R4, @(R5)
Effective Address = NA **(1.5 points)**

Value: R4 = R4 + Mem[2000] = 0 + Mem[2000] = 4000**(1 point)**

3)SUB R2, R4
Effective Address = NA **(1.5 points)**

Value: R2 = R2 - R4 = 2000 - 4000 = -2000 **(1 point)**

4)LOAD R6, 1000(R0)
Effective Address = 1000 + R0 = 1000 + 1000 = 2000 **(1.5 points)**

Value: R6 = Mem[2000] = 4000 **(1 point)**

5)ADD R6, R2
Effective Address = NA **(1.5 points)**
Value: R6 = R6 + R2 = 4000 + (-2000) = 2000 **(1 point)**

6)SUB R5, R6
Effective Address = NA **(1.5 points)**

Value: R5 = R5 - R6 = 5000 - 2000 = 3000 **(1 point)**

7)ADD R2, R5
Effective Address = NA **(1.5 points)**

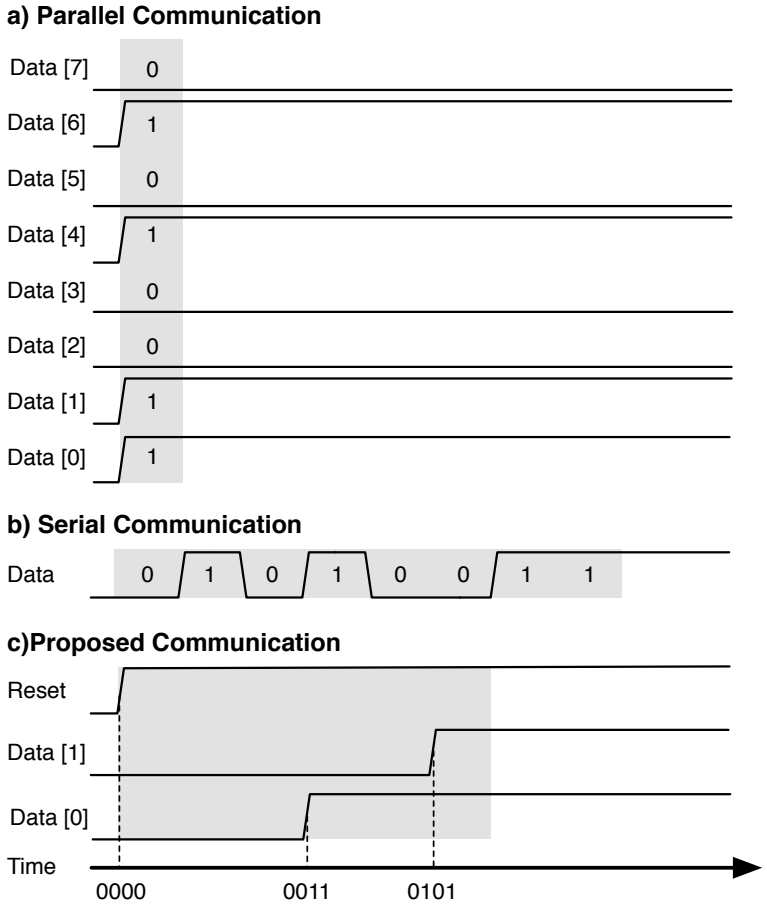
Value: R2 = R2 + R5 = -2000 + 3000 = 1000 **(1 point)**

8) ADD R2, R3
Effective Address: NA **(1.5 points)**

Value: R2 = R2 + R3 = 1000 + 5000 = 6000
(1 point)

5. **Pipelining.** We design a simple microprocessor core that implements five operations similar to those of the basic MIPS architecture. The delays of these operations are 36ns, 39ns, 23ns, 28ns and 64ns. Consider designing both non-pipelined and pipelined versions of the microprocessor. (Inserting a pipeline register between operations adds 1ns to the critical path delay.) What is the maximum achievable speedup of the pipelined version over non-pipelined architecture when executing a large number of instructions (n>>5)? **(15 points)**
   $SpeedUp = \frac{Exectime-NO-piplining}{Exectime-piplining} = \frac{ICold \times CPIold \times CTold}{ICnew \times CPInew \times CTnew}$
   IC remains same. **(1 point)**
   For computing Maximum achievable speed up, the ratio of CPI is 1. **(3 points)**
   $CTold$ = 36+39+23+28+64 +1= 191 ns **(3 points)**
   $CTnew$ = max (36,39,23,28,64)+1 = 65 ns **(6 points)**
   $Speedup = \frac{191}{65} = 2.93$ **(2 points)**
6. **Bonus Question.** Data transmission on a limited number of wires is crucial to power and performance of computer systems. Two popular communication techniques are called *parallel* and *serial*

shown in Figure 1 (a) and (b). Suppose that all wires are holding zeroes prior to transferring data. With the parallel communication, we are using eight physical wires to transfer eight bit values in one cycle. In the serial communication, we use only one physical wire to send 8 bit values in 8 cycles.



We propose a new encoding method that needs three wires–two for data and one for reset. Every data byte is divided into two four-bit *chunks*. Each chunk is transferred by toggling one of the two data wires (Data [0] and Data [1] in Figure 1). The reset wire is shared by all data wires to specify the start of the data transmission (before the transmission of one data byte). The number of clock cycles between the reset signal and a bit-flip on a data wire represents the chunk value. We use a counter that starts counting as soon as a transition appears on the reset wire. The contents of the counter on every data transition represents the chunk value.

Suppose that we have three applications A, B, and C transferring data on the memory interface. The following table shows the transferred data for each application. (Least significant bits, chunks, and bytes are transferred first.)

Application	Data Value
A	11001010 01110011 11000001
B	00011010 01110011 11001001
C	01010101 01010101 01010101

i Compute the total number of switchings on the wires for each application when using the serial, parallel, and the proposed techniques. **(5 points)**

Solution: This example solution assumes: (1) one transition per byte on the reset wire, (2) one transition per byte on each data wire, and (3) the next byte will start to transfer only if the previous byte is completely sent. (*Other possible scenarios based on your reasonable*

assumptions may be accepted as correct solutions.) We need to track the data patterns on wires to count the number of transitions.

Table 1: Number of Switchings (Transitions).

Application	Serial	Parallel	Proposed
A	11	12	9
B	12	13	9
C	24	4	9

ii Suppose that the only parameter that changes during data transmission using these techniques is the number of bit flips (i.e., the wire capacitance and voltage supply are fixed). Compute the relative dynamic power of these techniques to serial communication. **(5 points)**

Solution: For computing the number of required cycles, we know for serial each bits takes one cycle; for parallel each byte takes one cycle; for the proposed technique, the transmission time depends on the contents of each chunk. For every byte the maximum chunk value determine the transmission time.

Table 2: Number of Cycles

Application	Serial	Parallel	Proposed
A	24	3	12+7+12=31
B	24	3	10+7+12=29
C	24	3	5+5+5=15

Number of switchings (transitions) per cycle can be obtained through dividing the number of transitions by the number of cycles.

Table 3: Number of switchings per cycle (A)

Application	Serial	Parallel	Proposed
A	0.45	4.0	0.291
B	0.5	4.3	1.3
C	1	1.3	0.6

As C, V, and f are the same across all the communications schemes, $P_{dynamic}$ relative to $P_{dynamicofserial}$ can be simplified by A/A_{serial}. To summarizes the computed numbers across multiple applications we should use the geometric mean (GEOMEAN).

Table 4: Dynamic power normalized to serial communication

Application	Serial	Parallel	Proposed
A	1	8.7	0.63
B	1	8.6	0.62
C	1	1.3	0.6
GEOMEAN	1	4.6	0.61

iii Compute the energy-delay-product of these techniques normalized to the serial communication. **(5 points)**

Solution: Energy is obtained by $Energy = power \times Time$. We assume that static power is zero. $Time$ = Number of Cycles $\times$ Cycle Time. Also, Cycle Time= $1/f$. Therefore, $Energy = C \times V^2 \times f \times A\times$Number of Cycles $\times 1/f$. Recall that A =Number of Switchings / Number of Cycles. As a result, $Energy = C \times V^2\times$Number of Switchings.

Table 5: (Dynamic) Energy

Application	Serial Trans	Parallel	Proposd
A	$11CV^2$	$12CV^2$	$9CV^2$
B	$12CV^2$	$13CV^2$	$9CV^2$
C	$24CV^2$	$4CV^2$	$9CV^2$

Next, the Energy-Delay-Product can be used by multiplying the energy results by delay. After normalizing them to the serial communication and using GEOMEAN, we obtain the following Table.

Table 6: Energy-Delay product normalized to the serial communication

Application	Serial	Parallel	Proposed
A	1	0.13	1.05
B	1	0.13	0.90
C	1	0.02	0.23
GEOMEAN	1	0.07	0.61

iv Give three bytes data value which will lead to worst case execution time for the proposed method. **(5 points)**

Solution: For the proposed method, the execution time depends on the contents of chunks. The worst case happens when at least one of the chunks per each transferred byte is set to the maximum chunk value (i.e., 1111). An example data pattern with the worst case delay is "11111111 11111111 11111111"