Review Test Submission: Homework 1 (due 9/11)

User	Yuchen Wang
Course	CSE661/CIS655 - Advanced Computer Architecture - F20
Test	Homework 1 (due 9/11)
Started	9/3/20 11:59 PM
Submitted	9/11/20 10:22 PM
Due Date	9/11/20 11:59 PM
Status	Completed
Attempt Score	100 out of 100 points
Time Elapsed	190 hours, 23 minutes
Results Displayed	All Answers, Submitted Answers, Correct Answers, Feedback, Incorrectly Answered Questions

Question 1

12 out of 12 points

[Max 1 page] Read and briefly summarize the paper titled “A New Golden Age for Computer Architecture” and answer the following questions:

a. What are the reasons of switching from CISC to RISC?
b. In which situation VLIW will fail and in which will still be available?
c. What are the main current challenges for processor architecture?
d. List the approaches improving program performance in hardware technology.

Selected Answer:

- What are the reasons of switching from CISC to RISC?
  - RISC instructions were simplified so there was no need for a microcoded interpreter. RISC instructions could be executed directly by the hardware
  - The fast memory, formerly used for the microcode interpreter of a CISC ISA, was repurposed to be a cache of RISC instruction.
  - Register allocators based on Gregory Chaitin's graph-coloring scheme made it much easier for compliers to efficiently use registers, which benefited these register-register ISAs, such as RISC.
  - Moore's Law meant there have been enough transistors since the 1980s to include a full 32-bit datapath, along with instruction and data caches, in a single chip
  - More complicated CISC ISA executed about 75% of the number instruction per program as RISC, but in a similar technology CISC executed about five to ix more clock cycles per instruction, making RISC microprocessors approximately 4 times faster.
  - Current mobile-device designers valued die area and energy efficiency as much as performance, disadvantaging CISC ISAs.
  - Nowadays, arrival of IoT vastly increased both the number of processors and the required trade-offs in die size, prower, cost and performance, increasing the importance of design time and cost, further disadvantaging CISC ISAs.
- In which situation VLIW will fail and in which will still be available?
  - VLIW processors will fail/work poorly on general-purpose code
  - VLIW processors will still be available for an explicitly parallel program, and small programs, and simpler branches and omit caches, including digital-signal processing.
- What are the main current challenges for processor architecture?
  - End of Moore's Law: Transistor density is not doubling every two year anymore. CMOS technology approaches fundamental limits.
  - End of Dennard Scaling: Began to slow significantly in 2007 and faded to almost nothing by 2012
  - Need to find more efficient ways to exploit parallelism and do energy-efficiency computation due to the end of Dennard Scaling
  - With the end of Dennard Scaling, increasing the number of cores on a chip also raise problem on the limit of TDP
  - Computer security problems/concerns.
- List the approaches improving program performance in hardware technology.
  - By improving the performance of modern high-level languages that are typically interpreted
  - By building do-main-specific architectures that greatly improve performance and efficiency
  - ILP and Multicore techniques
  - Domain-specific architectures

Correct Answer: [None]

Response Feedback: [None Given]

Question 2

8 out of 8 points

Determine ranges (min and max) for transistor counts, feature size, clock frequencies in the 70's, 80's, 90's, 00's, 10's, and today. Show your work in a tabular forma

	70s	80s	90s	00s	10s	today

Transistor Count						
Feature Size						
Clock Frequency						

Selected Answer:

	70s	80s	90s	00s	10s	today
Transistor Count	2,250 - 68,000	11,500 - 1,180,235	35,000 - 27,400,000	21,000,000 - 1,900,000,000	250,000,000 - 39,540,000,000	4,800,000,000 - 54,000,000,000
Feature Size	10 µm - 3 µm	3 µm - 800 nm	600 nm - 180 nm	180 nm - 32 nm	32 nm - 7 nm	7 nm - 5 nm
Clock Frequency	0.108 MHz - 14 MHz	6 MHz - 50 MHz	33 MHz - 833 MHz	400 MHz - 5000 MHz	800 MHz - 4000 MHz	2300 MHz - 3800 MH

Correct Answer: [None]

Response Feedback: [None Given]

Question 3

10 out of 10 points

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. According to the trend in device scaling observed by Moore's law, the number of transistors on a chip in 2020 should be how many times the number in 2000? Does the prediction come true?

Selected Answer: As the Moore's Law claims 'the number of transistors in a dense integrated circuit doubles about every two years', we expect the number of transistors on a chip in 2020 be **2^((2020 - 2000)/2) = 2^10 = 1024** times the number in 2000.
This prediction seems doesn't come true as it underestimates the reality, as **54,000,000,000/27,400,000 = 1970.8**

Correct Answer: [None]

Response Feedback: [None Given]

Question 4

10 out of 10 points

When parallelizing an application, the ideal speedup is speeding up by the number of processors. This is limited by two things: percentage of the application that can be parallelized and the cost of communication. Amdahl's law takes into account the former but not the latter. What is the speedup with N processors if 90% of the application is parallelizable, ignoring the cost of communication? What will be the speedup for a system with 100 processors? What will be the speedup for a system with 1000 processors? What will be the speedup for a system with 10000 processors? What is the limit of overall speedup?(infinite number of processors) Round to **2 decimal places.**

Selected Answer: As 90% of the application is parallelizable, and Speedup(enhanced) is equal to the number of processor(N), thus overall speedup = 1/((1-p)+(p/s)) = 1/((1-0.9)+(0.9/N)) = 1/(0.1+0.9/N)

- N processors:**Overall speedup = 1/(0.1+0.9/N)**
- 100 processors: **Overall speedup = 1/(0.1+0.9/100) = 9.17**
- 1000 processors: **Overall speedup = 1/(0.1+0.9/1000) = 9.91**
- 10000 processors: **Overall speedup = 1/(0.1+0.9/10000) = 9.99**
- infinite number of processors(limit of overall speedup): **Overall speedup = 1/(0.1+0.9/infinite) = 10**

Correct Answer:

Response Feedback: [None Given]

Question 5

10 out of 10 points

A single-core processor, runs four applications with the same execution time on this system. Assume the application characteristics as listed. If we switch to a 16-core processor, what is the overall speedup if B is run parallelized but everything else is run serially.

Application	A	B	C	D
% Parallelizable	50	80	60	90

Selected Answer: Overall speedup = 1/((1-p)+(p/s)).
As A, B, C and D were ran with the same execution time on this system with a single-core processor, and now we switch to a 16-core processor with B run parallelized but everything else is run serially, and also as the rate of parallelizable of application B is 80%. Hence, we have: **Overall speedup = 1/((1-(1/4) * 0.8)+((1/4) * 0.8/16)) = 1/(0.8 + 0.2/16) = 1.23**

Correct Answer: [None]

Response Feedback: [None Given]

Question 6

20 out of 20 points

You have the following characteristics, as shown in the table below, on your company's processor for a certain benchmark, which runs at 600 MHz:

Instruction Type	Frequency (%)	Cycles
Arithmetic and logical	30	1
Load and Store	25	2
Branches	40	3
Floating Point (FP)	5	10

You are asked to consider a cheaper, lower-performance version of this processor, by removing some of the FP hardware to reduce the die size. The wafer has a diameter of 20 cm, costs $2,000, and has a defect rate of 1/(cm2). This wafer has a 70% yield. The current chip has a die size of 10 mm2. The new chip becomes 8 mm2, and FP instructions will now take 20 cycles to execute.

a. What are the old and new CPI (Cycles Per Instructions) and MIPS (Million Instructions Per Second) ratings running this benchmark?
b. What are the old and new die yields, assuming the factor N being 4 in the die yield formula on page 34 in the textbook? What are the old and new costs per (working) processor? Please comment on the overall effect of the proposed hardware change on the cost and the performance of the processor.
c. What would be the theoretical limit of the best possible overall speedup that we could ever get by only improving the FP unit, and what would be the CPI and MIPS ratings of this new processor?

Selected Answer:

- What are the old and new CPI (Cycles Per Instructions) and MIPS (Million Instructions Per Second) ratings running this benchmark?
  - CPI = CPU clock cycles for a program/Instruction count
    - Old CPI =0.3*1+0.25*2+0.4*3+0.05*10/0.3+0.25+0.4+0.05=2.5
    - New CPI =0.3*1 + 0.25*2 + 0.4*3 + 0.05*20/0.3+0.25+0.4+0.05 = 3
  - MIPS = IC/(CPU time in Second *1,000,000)= 1/(CPI * CCT *1,000,000) = Clock rate/(CPI *1,000,000)
    - Old MIPS = 600,000,000/(2.5 * 1,000,000) =240
    - New MIPS = 600,000,000/(3 * 1,000,000) = 200
- What are the old and new die yields, assuming the factor N being 4 in the die yield formula on page 34 in the textbook? What are the old and new costs per (working) processor? Please comment on the overall effect of the proposed hardware change on the cost and the performance of the processor.
  - Die yield = Wafer yield *1/(1+ Defects per unit area * Die area)^N
    - Old die yield =0.7*1/(1+1*0.1)^4=0.48
    - New die yield = 0.7 * 1/(1+ 1 * 0.08)^4 = 0.51
  - Costs per processor = Cost of wafer/(Dies per wafer * Die yield) **OR IT COULD BE** = (Cost of die + cost of testing die + cost of packaging and final test)/Final test yield;
    - Old Die per wafer = (π * (Wafer diameter/2)^2)/Die area - (π * Wafer diameter) /(2 * Die area)^(1/2) = (π * (20/2)^2)/0.1 - (π * 20) /(2 * 0.1)^(1/2) = 3141.6 - 140.5 = 3001
    - New Die per wafer = (π * (Wafer diameter/2)^2)/Die area - (π * Wafer diameter) /(2 * Die area)^(1/2) = (π * (20/2)^2)/0.08 - (π * 20) /(2 * 0.08)^(1/2) = 3926.9 - 157.1 = 3769
    - With formula: Cost of wafer/(Dies per wafer * Die yield)
      - Old costs per processor = 2000/(3001 * 0.48) = $1.39
      - New costs per processor = 2000/(3769 * 0.51) = $1.04
    - With formula: (Cost of die + cost of testing die + cost of packaging and final test)/Final test yield; 'we assume cost of testing die'(A), 'cost of packaging and final test'(B) and 'Final test yield'(C)
      - Old costs per processor = (1.39 + A1)+ B1)/C1
      - New costs per processor = (1.04 + A2 + B2)/C2
  - The overall effect of the proposed hardware change on the cost and the performance
    - Cost is dramatically decrease and that is good
    - Performance is getting worse as CPI increases and MIPS decreases.
- What would be the theoretical limit of the best possible overall speedup that we could ever get by only improving the FP unit, and what would be the CPI and MIPS ratings of this new processor?
  - I assume to improve the FP unit to the theoretical limit of the process, as FP instruction only take 1 cpu clock cycle to process
    - CPI(new) = 0.3*1+0.25*2+0.4*3+0.05*1/0.3+0.25+0.4+0.05 = 2.05
    - MIPS(new) = 600,000,000/(2.05 * 1,000,000) = 292.68
    - Overall speedup = Execution time(old) / Execution time(new) = IC(old) * CPI(old) * CCT(old) / IC(new) * CPI(new) * CCT(new) = CPI(old) / CPI(new) = 3/2.05 = 1.46

Correct Answer: [None]

Response Feedback: [None Given]

Question 7

10 out of 10 points

Your company produces a mobile device. To extend the battery life in the newer version of the device, you are asked to elaborate on the idea to simply reduce the processor clock speed by 20% and make no other changes. Stating your assumptions, describe whether this is a good idea or a bad idea, and why? Make sure to address both power and energy.

Selected Answer: It is clear that reducing the processor clock speed would do some work on Power(dynamic), as Power(dynamic) is direct proportional to processor clock speed/frequency.

- **Power(dynamic) ∝ (1/2) * Capacitive load * Voltage^2 * Frequency switched**

However, it will make no differences on energy consumption if the user want to do a same task they have done on the previous device on the new device, because the new device will be slow and take more time than old devices, canceling the effect of power reduction, as

- **Energy(dynamic) ∝ (1/2) * Capacitive load * Voltage^2**

Hence, the battery life will last longer than the old devices, but the amount of things users can achieve on the new devices are the same as the old devices: For a fixed task, slowing clock rate reduces power, but not energy.
And I think it is not a good idea, as I don't think users want devices performing slower than old ones.

Correct Answer: [None]

Response Feedback: [None Given]

Question 8

5 out of 5 points

In this introduction part of the coding assignment, you will need to get ready for the next question. Please make sure to be ready to proceed with the assignment and mark YES if you are ready.

How to install Ubuntu:

If you don't already have access to a linux server on your computer, you can either get **Ubuntu** from Windows Store or visit https://www.windowscentral.com/how-install-bash-shell-command-line-windows-10 to install **Bash on Ubuntu** on Windows 10. Make sure your Ubuntu is up to date by running necessary upgrades and updates (sudo apt update; sudo apt upgrade; sudo do-release-upgrade / -d).

How to use a unix shell

```
man grep                    displays documentation about a command
mkdir CSE661                creates a new directory
cd CSE661                   changes the directory
pwd                         displays the full path of the current directory
cp asm.tar.gz CSE661        copies a file into a directory
mv asm.tar.gz CSE661        moves a file into a directory
gunzip asm.tar.gz           unzips asm.tar.gz into asm.tar
unzip test.zip              unzips a zipped file
tar xvf asm.tar             unpackes decoder.tar
ls                          lists all directories and files in the current directory
ls -a                       lists all the files in a directory, including the normally hidden files
ls -l                       lists all the files with more long format
ls -a asm                   lists all folders and files in a directory
touch test1                 creates a file
rm test1                    removes a file
rm *.txt                    removes all files with .txt extension
rm -r temp                  removes temp even if it is not empty
rmdir temp                  removes temp directory (if it is empty)
cat test1.txt               displays contents of test1.txt
quota -v                    displays users disk usage and limits
ps                          list processes with their pids
kill <pid>                  terminates a process with <pid>
emacs test1.asm             opens a text editor, emacs
vi test                     opens a popular text editor, vi
gedit                       opens a graphics text editor
lpr test.asm                prints a file with the default printer
```

How to compile and run assembly codes:

To compile the file

```
$ nasm -f elf64 filename.asm
```

To link the file

```
$ ld -s -o filename filename.o
```

To run the file

```
$ ./filename
```

OR, you can simply do all in one line:

```
nasm -f elf64 filename.asm && ld filename.o && ./a.out
```

Selected Answer: Yes
Answers: Yes
No

Question 9

15 out of 15 points

Please create the following two files, and assemble, compile & execute them. Revise the code with several scenarios and get their screen shots (showing your user name as well) and upload them here.

callsum.c:

```
/* calls a sum function written in assembly language. */

#include <stdio.h>
#include <inttypes.h>
```

```
double sum(double[], uint64_t);
int main() {
 double test[] = { 22.8, -16.2, 23.1, -39.2, 40.5, 13.5, 999.9 };
   printf("%30.5f\n", sum(test, 1));
   printf("%30.5f\n", sum(test, 2));
   printf("%30.5f\n", sum(test, 7));
   printf("%30.5f\n", sum(test, 0));
return 0;
}
```

sum.asm:

```
; returns a sum of an array
    global  sum
    section .text
sum:
    xorpd   xmm0, xmm0    ; initialize the sum to 0
    cmp     rsi, 0        ; special case for length = 0
    je      done
next:
    addsd   xmm0, [rdi]   ; add in the current array element
    add     rdi, 8        ; move to next array element
    dec     rsi           ; count down
    jnz     next          ; if not done counting, continue
done:
    ret                   ; return value already in xmm0
```

To run the code:

```
nasm -felf64 sum.asm && gcc sum.o callsum.c && ./a.out
```

Selected Answer: HW.png
Response Feedback: [None Given]
Thursday, October 8, 2020 7:21:05 AM EDT

← OK