Kerry Gip

2/22/22

Review Homework

Due in One Week (Submit solutions to all questions to Canvas under assignments)

This is a review assignment to refresh your memory regarding instruction sets, formats, and memory accesses. While we are reviewing much of this in class, it is useful for you to do this exercise to become more comfortable with the base concepts as we move to advanced topics. Your lecture notes and Appendix A of your textbook are good sources of study.

Note: Submit to Canvas. All typed work must be in MS Word.

1. Consider a load-store type machine with the following specifications:

- 2^{32} x bytes of memory
- 32-bit fixed format instructions
- 32 32-bit general purpose registers (GPR)
- 3-address register-to-register arithmetic instructions
- Single address mode for load/store: base + displacement
- Capable of performing a total of 32 arithmetic operations

For simplicity assume that the machine only performs arithmetic operations plus data transfer operations (i.e. load and store).

a) Write the equivalent machine level language corresponding to a C

statement of C = A + B

Load Ra ← A

Load Rb ← B

Add R3 ← Ra + Rb

Store R3 ← C

b) Give an instruction format for the arithmetic operations. To do this draw a diagram of the instruction format with each field clearly specified.

For each field indicate its size, the reason for selected size, and a description of what purpose the field serves.

31 27 25 21 20 16 15 11 10 6 5 0

Opcode	Rn	Rm	Rd	Shift/Shamt	Function
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

Opcode -
- 6 bits for 2^5 bits that would equal 32-bit fixed format instruction operation plus an additional bit for load/store

Rn - Source Register 1 with base address
- 5 bits for 2^5 bits that would equal 32-bit

Rm - Source Register 2 with target address
- 5 bits for 2^5 bits that would equal 32-bit

Rd - Destination register
- 5 bits for 2^5 bits that would equal 32-bit

Shift/Shamt - accounts for the displacement
- 5 bits

Func - Used for functions in the opcode
- 6 bits

c) Give an instruction format for the load/store operations. To do this draw a diagram of the instruction format with each field clearly specified. For each field indicate its size, the reason for selected size, and a description of what purpose the field serves.

opcode	Rn	Rm	Immediate
6 bits	5 bits	5 bits	16 bits

I-format instruction of load/store

Opcode -
- 6 bits is what's leftover bits needed to equal 32 bits. Rn and Rm have to equal each other since they come from the source register

Rn - Source Register 1 with base address
- 5 bits for 2^5 bits that would equal 32-bit

Rm - Source Register 2 with target address
- 5 bits for 2^5 bits that would equal 32-bit

Immediate - Immediate address or offset
- the load immediate instruction loads 16 bits into half of a 32-bit register. The immediate address does not have access to memory

2. To see how different ISA decisions will impact the machine design, consider designing an accumulator machine with the following specifications:

- 2^{24} words of memory [words are 32-bit wide]
- Fixed format instructions
- A 32-bit accumulator register (AC)

- An index register X
- Index address mode: address field + X when indexing is indicated in the instruction
- Capable of performing a total of 128 operations

a) Write the equivalent machine level language corresponding to a C

statement of C = A + B

Accumulator: implicit that one of the operands is accumulator in the CPU A + B =

C

M = Machine AC = Accumulator

Load M[A] -> AC

Add M[B] #AC and M[B]

Store C, M[C] #the contents of AC

b) Give an instruction format for this computer. To do this draw a diagram of the instruction format with each field clearly specified.

opcode	index addressing bit/mode	index(X)
7 bits	1 bit	24 bit

opcode : 7 bits because $2^7 = 128$
index = 24 bits because 2^{24} words of memory
index addressing bit = 1 bits because 24+7 = 31, so 1 bit is leftover to fit in the 32 bit accumulator register

For each field indicate its size, the reason for selected size, and a description of what purpose the field.

3. When designing memory systems, it becomes useful to know the frequency of reads versus writes as well as the frequency of accesses for instructions versus data. Using the average instruction-mix information for MIPS for the program spice (as given below), find the following:

a. The percentage of all memory accesses that are for data (vs. instructions).

- LW and SW are the only ones that have data memory
- $\frac{Data}{Data + Instructions} = \frac{.41 * IC}{.41 * IC + IC} = \frac{.41\,IC}{1.41\,IC} = .29 = 29\%$

b. The percentage of all memory accesses that are reads (vs. writes). Assume that two-thirds of data transfers are loads.

- $\frac{Reads}{Data + Instructions} = \frac{IC + \frac{2}{3} * .41 * IC}{1.41 * IC} = \frac{1.27\,IC}{1.41\,IC} = 90\%$

Instruction Class	MIPS Examples	HLL Correspondence	Frequency (spice)

Arithmetic	add, sub, addi	operations in assignment statements	50%
Data transfer	lw, sw	references to data structures	41%
Conditional branch	beq, bne, slt, slti	If statements and loops	8%
Jump	j, jr, jal	procedure calls/returns, case/switch statements	1%

4. Literature Report Assignment:

For this assignment you need to do a literature search on advanced computer architectures topic. Find a peer reviewed publication on a topic of your interest. Read the article and write a one paragraph summary of the objectives, methods, findings of the research.

We will have several of this type of assignments. It is important to distinguish refereed research publications (reviewed by peer researchers) from non-refereed publications (white papers). To get access to refereed publications such as ACM, IEEE, etc, you can login to your library account at UCD. Through your membership you have access to most publications, even inter-loan library.

These exercises will help you identify research topics for your in-depth study research project. It will also help you on finding a project for experimental implementation for your term project.

The title of the paper is called “Performance Enhancement of Multicore Architecture“ by Awadalla, Medhat & Konsowa, Hanan.

This paper talks about how to create new algorithms for thread selection in fetch stage which is needed to enhance the performance of multicore architecture. Fetch is part of a fetch/execute cycle where it reads an instruction and then executes it. The group made 3 new fetch policies. By creating these new algorithms and the fetch policies, they were able to increase the execution time and number of instructions per second. They go onto describing different architectures, simulations, multithreading, benchmarks and other algorithms that they use to increase efficiency and power.
The methods include updating the multicore simulator and utilizing the 3 fetch policies. The result is that the new algorithms and fetch policies avoid resource starving by allowing a dynamic energy distribution to the threads without compromising performance. [1]

[1] Awadalla, Medhat & Konsowa, Hanan. (2015). “Performance Enhancement of Multicore Architecture”. *International journal of electrical and computer engineering*. Vol. 5, No. 4, 10.11591/ijece.v5i4.pp669-684. Accessed : 2/2/2022. [Online]. Available: http://ijece.iaescore.com/index.php/IJECE/article/view/5661/4628

Periodically, we will discuss your research topics in class to explore common interest.

What to submit:

- Put the submission date of the paper on the top right corner. Don't forget your name!
- Your typed paragraph (MS Word), written in your own words interpreting the paper, no more than one page, font size 12.
- A complete reference to the paper, please refer to the Reference Guide, IEEE Style at https://ieeeauthorcenter.ieee.org/wp-content/uploads/IEEE-Reference-Guide.pdf
- to be sure you present your reference correctly.
- Give the URL address to the paper if you have it.
- Post a copy of the article you reviewed as part of research in the Shared Space/Literature section in the Modules of our course in Canvas. If your article(s) can be accessed online, please post the title, one sentence description, correct citation (in the IEEE format), and its URL. The purpose is for everyone to have access, be able to review these articles throughout the semester, and hopefully for students to form teams based on research interests.