ROHAN HULSURE
FEB 4, 2019

CSCI - 5593 - HOMEWORK - 1

---

1. Semester website individual assignment:

Hulsure , Rohan , rohan.hulsure@ucdenver.edu ,
https://github.com/cu-hulsurerohan/CSCI-5593-Computer-Architecture ,

2. 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

Solution:

```
LOAD R1, A
LOAD R2, B
ADD R3, R1, R2,
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.

Solution:

Based on the given machine we can have regular type format with following constructs:

1. Instruction format 32 bits
2. Registers - 3 registers (source, target, destination) of 5 bits each.

3. OPCODE - OPCODE will be of size 5 bits + 1 extra bit to specify address mode. Therefore, total 6 bits.
4. SHIFT - used in shift/rotate operations. This will have size of 5 bits which will specify an amount by which the source operand is shifted/rotated.
5. FUNCTION - An addition to opcode field. Used to specify control code or to specify operation.

6 Bits	5 Bits	5 Bits	5 Bits	5 Bits	6 Bits
OPCODE	**RS**	**RT**	**RD**	**SHIFT**	**FUNCTION**

32 Bits Instruction

Fig : Instruction Format

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.

Solution:

For load/store type operation we can have following constructs:

1. Instruction format 32 bits.
2. Registers - 2 Registers (Source, Target) of 5 bits each.
3. OPCODE - OPCODE will be of size 5 bits + 1 extra bit to specify address mode. Therefore, total 6 bits.
4. Displacement - 16 bits for displacement. This is the immediate value which is added to the base register, hence called displacement.

6 Bits	5 Bits	5 Bits	16 Bits
OPCODE	**RS**	**RT**	**Displacement**

32 Bits Instruction

Fig: Instruction Format

3. 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

Solution:

LOAD A
ADD B
STORE C

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

Solution:

Based on given machine for operations on Accumulator we can have instruction format with following constructs:

1. OPCODE - We have total 128 operations (2^7). OPCODE will require 7 bits.
2. INDEX - Size of 1 bit allocated to index register. Index register will have value 1 or 0. If value is 1 there is indexing.
3. IMM - Immediate Address will have size of 24 bits, each address representing location of word.

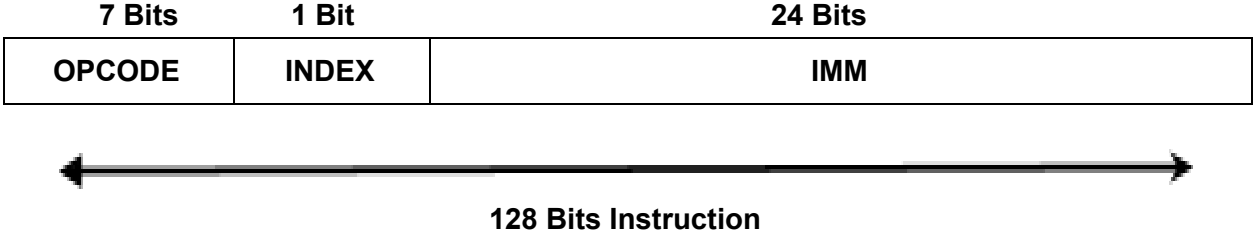

Fig: Instruction Format

4. 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:

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%

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

Solution:

Based on the above table we can see that both LOAD and STORE use 41%. Therefore with the formula given below we can find percentage of all memory access

$$\frac{41\%}{1+41\%} = \frac{41\%}{1.41} = 0.2907 * 100 = \mathbf{29.07}\ \%$$

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

Solution:

Based on the above table we can see that both LOAD and STORE use 41%. Assuming ⅔ of the data transfer are LOAD (27.33%). Therefore with the formula given below we can find percentage of all memory access that are read:

$$\frac{1+27.33\%}{1+41\ \%} = \frac{1.2733}{1.41} = 0.9030 * 100 = \mathbf{90.30}\ \%$$