DRISHTI GOEL

Feb 01, 2019

ASSIGNMENT 1

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

A. Write the equivalent machine level language corresponding to a C statement of C = A + B

LOAD r1, A

LOAD r2, B

ADD r3, r1, r2

STORE C, r3

1. 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.

ADD r3, r1, r2



 op  rs  rt  rd  Shamt  funct

op – operation of the instruction(operation code)--- 6 bits

* partially specifies what instruction it is. rs - source register -1 --- 5 bits

rt - source register -2 --- 5 bits rd - destination register --- 5 bits

shamt – This field contains the amount a shift instruction will shift by. --- 5 bits funct – function code (extends opcode) – 6 bits

* + combined with opcode, this number exactly specifies the instruction

Based on the Machine given above, we have 32 instructions and for the OPCODE we need 5 bits, but 3 other bits we need to determine which addressing mode we use.

Then I added 5 bits for each registers (Source, Target and Destination register). Since it is a 32-bit fixed format instruction, I have added the SHIFT and FUNCTION fields. The SHIFT field is used to save the amount of bits that are used to shift. This field is used in shift instruction. And the FUNCTION field has been added because 6 bits are used to specify the operation (this one is an addition to the opcode field)

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.

LOAD R1,A or STORE C,R3



 OP  rs  rt  Address/Immediate(IMM) 



op - operation of the instruction – 6 bits

rs – first register source operand – 5 bits

rt – second register source operand – 5 bits

Address/Immediate(IMM) – This value is usually used as the offset value in various instructions.

--- 16 bit signed two’s complement value.

6 Bits are added to the OPCODE for the same reason as in the task B. There 32 instruction for which we need 5 bits but we also need 3 other bits to determine which addressing mode we use The 2 registers have each 5 Bits. And the DISPLACEMENT, we add the value to the Base Register.

1. 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

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.

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



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **7bits** |  | **1bit** |  | **24bits** |
| **OPCODE** |  | **INDEX** |  | **ADDRESS SIZE** |

Based on the machine given above, it says that it is capable of performing 128 operations, so we have added 7 bits to the OPCODE to be able to achieve that. The INDEX field is either 0 or 1. It is 0 (zero) when there isn’t any indexing and it is 1 (one) where there is indexing. And for the ADDRESS Field I put 24 bits to be able to address all the memory locations.

1. 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).

* of all memory access for data = data access/(instruction access + data access)
  + (41%)/(1+41%)
  + 29.07%

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

* of all memory access that are reads = read/all access
  + (instruction access + load)/( instruction access + load+ store)
  + (1 + 2/3(41%))/(1+41%)
  + (1.27)/(1.41) = 90%