cs 394 introduction to computer architecture

assignment: homework 1

name: morgan bergen

wsu id: b493r546

question 1

according to the definition of a credit hour describe in syllabus, briefly explain how many total hours you are expected to spend to succeed in this course

a - in a normal week

9 hours per week, 3 hours in class and 6 hours outside of class.

b - in a 15 week semester

question 2

a - what are the instructor's student / office hours?

tuesday 11:00 am - 12:30 pm, wednesday 10:00 am - 11:30 am

b - how to contact the instructor during the student / office hours?

in-person during office hours or via email

question 3

using the lecture slides, briefly discuss

a - computer

a computer is a machine that receives inputs, manipulates and or stores data, and provides useful outputs.

the levels of abstraction in a computer are as follows:

level 5 the problem oriented language level, you translate the program via a compiler to level 4

level 4 the assembly language level, you translate the program via an assembler to level 3

level 3 the operating system machine level, you partially interpret the program via the operating system to level 2

level 2 the instruction set architecture level, you interpret the microprogram or directly execute the program to level 1

level 1 the microarchitecture level, you execute the program via hardware to level 0

level 0 the digital logic level

b - computer architecture

computer architecture or digital computer organization is a set of rules and methods that describe the functionality, organization, and implementation of computer systems. computer organization helps optimize performance-based products. it can also be interpreted as the conceptual design and fundamental operational structure of a computer system.

c - von neumann architecture

the von neumann model is a computer architecture model designed by von neumann, it describes a system where the computer hardware is organized in a specific way to perform calculations and store data. the scheme is comprised of the following components:

a processing unit with both an arithmetic logic unit and processor registers

a control unit that includes an instruction register and a program counter

memory that stores data and instructions external mass storage

input and output mechanism

d - pure harvard architecture

the pure harvard architecture is a computer architecture with separate storage and signal pathways for instructions and data. it's contrasted with the von neumann architecture where the program instructions and data share the same memory and pathways. this architecture is often used in real time processing or low power applications. the components of the pure harvard architecture are the following:

alu - arithmetic logic unit

instruction memory

data memory

input/output

control unit

e - pdp-8 architecture

the pdp-8 architecture was developed in 1965 by DEC and was the first mass market minicomputer with 50,000 units sold. the pdp-8 architecture is a 12-bit machine with 4k words of memory. the pdp-8 had a major innovation: a single bus, the omnibus, which is simply just a collection of parallel wires used to connect the components of a computer.

question 4

based on the class lectures, briefly discuss major five steps to execute an instruction in a simple computer system

step 1 - instruction fetch - this process is done to take an instruction by fetching it from memory. the next instruction that needs to be executed will be handled by the program counter which holds the address of that next instruction. this address is sent to the memory and the instruction stored at that location is loaded into the instruction register of the cpu.

step 2 - instruction decode - once the instruction is fetched, the cpu needs to interpret what it needs to do. the control unit decodes the instruction stored in the instruction register. the instruction is then broken down into simple components, such as the operation code (opcode) and the addressing mode, which tells the cpu how to process the instruction and which operands it should use.

step 3 - instruction execution - after decoding, the cpu executes the instruction in the form of performing arithmetic, logical operations, or controlling the flow of the program like jumping to a different instruction. the actual execution depends on the type of instruction decoded.

step 4 - operand fetch - some instructions require additional data or operands to be fetched from memory or register before they can be fully executed. in this step, the cpu fetches the required operands from their locations, which could be in the cpu's registers, memory, or provided directly as part of the instruction itself.

step 5 - result write back to main memory - after executing the instruction, the result needs to be stored back into memory or a register so it can be used in future instructions. in this phase, the cpu writes the result of the executed instruction to the appropriate location, which could be a register or a specific memory address, depending on the instruction.

question 5

based on the class lectures, clearly discuss the major five steps to execute the following five separate instructions in a simple computer system

a - `[dreg−1] = 0`

components of the instruction

`dreg` is the data register that holds a memory address

`dreg − 1` represents the memory location one unit before the address stored in `dreg`

`[dreg − 1]` this is the memory location at `dreg − 1`

`= 0` this assigns the value at the specified memory location to 0

step 1 instruction fetch

the instruction `[dreg − 1]` is fetched from memory by the cpu, the cpu is assisted by the program counter which is pointing to the memory address which contains the instruction. the instruction register is then loaded with the instruction.

step 2 instruction decode

the instruction that was loaded into the instruction register is decoded by the control unit of the cpu, this decoding process determines what operation is needed to be performed. the cpu breaks down the instruction into its components, such as the opcode and the addressing mode. from this the cpu determines that it needs to calculate the memory location `dreg - 1` and store the value 0 at that calculated memory location.

step 3 operand fetch

in order for the cpu to calculate the memory location `dreg - 1`, the cpu needs to fetch the value stored in `dreg`'s register and subtract `1` from it to obtain the target memory address, namely `[dreg - 1]`.

step 4 instruction execution

now that the cpu calculated the target memory address. the cpu prepares for execution by setting up the necessary data paths to place the value `0` (in binary) into the data bus. the data bus is a communication system that transfers the data from the cpu to memory.

step 5 result write back to main memory

the cpu writes the value `0` in binary to the memory location at the address `[dreg - 1]`. this is done by sending the value `0` to the memory module at location `[dreg - 1]`.

b - `[mem - 1] = 1`

components of the instruction

`mem` is a memory address stored in a register

`mem - 1` represents the memory address one unit before the value stored in `mem`

`[mem - 1]` this denotes the memory location at the address `mem - 1`

`= 1` this operation assigns the value `1` to be stored at the memory location `[mem -`

`1]`

step 1 instruction fetch

the cpu fetches the instruction `[mem -1]` from memory, the program counter holds the address of this instruction, which is then loaded into the instruction register for processing

step 2 instruction decode

the control unit decodes the fetched instruction to determine what operation needs to be performed. the decoding involves breaking down the instruction into its components, such as the opcode and the addressing mode. the cpu determines that the target memory address must be calculated by subtracting `1` from the value stored in the `mem` register. this is where the value `1` will be written into.

step 3 operand fetch

the cpu fetches the value stored in `mem` and then subtracts `1` from it to determine the exact memory address.

step 4 instruction execution

the cpu now executes the instruction by storing the value `1` at the target memory address. the value `1` in binary is placed on the data bus.

step 5 result write back to main memory

the cpu finally writes the value `1` into the memory location at the address `mem - 1`. now the memory location `[mem - 1]` contains the value `1`

c - `[mem - 2] = 2 * [mem - 2]`

components of the instruction

`mem` is the memory address stored in a register

`mem - 2` indicates the memory address two units before the value stored in `mem`

`[mem - 2]]` denotes the content stored at the memory location `mem - 2`

`2 * [mem - 2]` is an expression that multiples the value stored at `[mem - 2]` by `2` `[mem - 2] = 2 * [mem - 2]` means the result of the multiplication should be stored back at the same memory location `[mem - 2]`

step 1 instruction fetch

the cpu fetches the instruction from memory, the program counter holds the address of this instruction and loads it into the instruction register for processing

step 2 instruction decode

the control unit decodes the fetched instruction, breaking it down into its components, such as the opcode and the addressing mode. there are three operations identified in this instruction. first the memory address is calculated by subtracting `2` from the value `mem`, the value at `[mem - 2]` is retrieved and multiplied by `2`, the resulting solution is then stored by at the memory location `[mem - 2]`.

step 3 operand fetch

memory location `[mem - 2]` contain the calculated value and the cpu fetches this operand, the cpu reads the the memory location `mem - 2` using the address bus, which contains needed value.

step 4 instruction execution

the cpu multiplies the fetched value by `2` and teh resulting value is then prepared for storage.

step 5 result write back to main memory

the cpu writes the result of the value back into the memory location `[mem -2]`. the cpu places the value on the data bus and sends it to the memory location overwriting the previous value.

d - `go to mem - 3`

components of the instruction

`go to` is an operation code (opcode) that tells the cpu what action to perform, in this case `go to` is a control operation that instructs the cpu to jump to a specific memory address.

`mem` represents a memory address which contains the value of `mem`

`- 3` is an offset that tells the cpu to jump to the memory address `3` units before the value stored in `mem`

step 1 instruction fetch

the cpu fetches the instruction from memory, the program counter points to the memory address where this instruction resides. the instruction is then loaded into the instruction register for processing.

step 2 instruction decode

the control unit decodes the fetched instruction and identifies the operation and updates the program counter to a new address based on the `go to` instruction. the cpu also decodes the operand `mem - 3` which indicates the memory address to jump to.

step 3 operand fetch

`mem - 3` is calculated, the cpu subtracts 3 from the value in `mem` to determine the target memory address.

step 4 instruction execution

the cpu updates the program counter to the new address `mem - 3`. then the cpu will fetch and execute the instruction at the new address.

step 5 result write back to main memory

the result of the execution was simply to update the program counter, and cpu now fetches the next instruction from the address `mem - 3`. so there is no actual data written back to memory.