Aaron Van De Brook

12/05/2019

CEC470

Final Project Questions

1. Of the 256 possible opcodes we can get from tan 8-bit opcode, how many are not being used in our instruction set, i.e., how many instructions could we add for future expansions of our processor?

    The memory and math & logic instructions use up the entirety of the bits allotted for different types of instructions, so there is no extra space for new instructions of those type. However, there is extra room in the branch instruction for at least 1 more instruction, and possible more using a different prefix (e.g. the bits that come before the bits specifying function, destination, etc.).

2. What would we need to add to our simulator to be able to include the following instructions: compare ACC with a constant, PUSH to or PULL from the stack, and take the 2's complement of ACC?

    Extra space would need to be added to the arithmetic instructions to allow for comparisons or we would need another subgroup of instructions specifically for comparison operations.

    The same case would apply to push and pull operations from a stack, we would either need to add more bits to memory operations or add another subgroup of instructions for push and pull operations.

    Since 2's complement is just a combination of 2 instructions that have already been implemented we could add a special instruction that maps to those to instructions executing sequentially.

3. If executeInstruction() were divided into two parts, decode and execute, what additional global resources would be needed for your simulator?

    Registers or variables would be needed to store the operand or operands of the instruction and/or where to store the result of an operation.

4. Make suggestions for ways to further subdivide the executeInstruction() function.

    As previously stated, the execute cycle could be divided into decode and execute cycles, but it could also be divided into another fetch cycle after the decode to grab the necessary operands for an instruction.