UCF

School of Computer Science
CDA 4150 Computer Architecture
Summer 2005

Homework: The VN-H16 Architecture

Due 6/6/05

In this homework you have to design, based on the VN machine explained in class, a one-address, 16 bits word, Harvard architecture. This machine will be called the VN-H16 computer and it has to be designed according to the following specification:

1.- 16 bits word (4 bits for instruction and 12 bits for addresses)

2.- Each time an instruction is executed, another one has to be prefetched.

3.- There will be three bits denominated "condition codes(CC)". These bits will indicate whether the result obtained from the ALU(and stored in Accumulator), after the execution of an arithmetic instruction, is equal, greater or less than zero(E,G,L). The CC flags are located in the PSW.

4.- Explain, the actions to be taken in case of the triggering of an interrupt. Consider the following cases: Overflow, I/O, Timer, System Call(SVC). Take into account that VN-H16 is a pipeline machine with two stages.

5.- Describe, using the Transfer Register Notation, the following ISA for the VN-H16 computer.

Load → LD <address>

Store → ST <address>

Add → ADD <address>

Sub → SUB <address>

Jump → JMP <address>

Skip → SKIP CC<10 bits unused> Increment PC by two if CC field in the instructions matches the CC flags; otherwise, PC is incremented by one.

What to turn in:

A.- The design of the architecture in a Power Point presentation(in a floppy), showing the execution cycle of each instruction in TRN.(including the Fetch cycle).

B.- A word/pdf document(hard copy) with the specification of the machine and the ISA.

Hints on how to describe the execution of VN-H16 architecture.

1. The clock distributor(CD) can be simulated using an array initialized to [1,0,0,0,…].At each clock tick the value of the clock distributor is shifted to the right, for example:
   [1,0,0,0,…] → [0,1,0,0,…] → [0,0,1,0,…].
2. Every time the DECODER is set to zero, the clock distributor is set to [1,0,0,0,…].
3. The execution of the instruction can be simulated this way:

IF CD[1] = 1 and DECODER = 00 Then MAR ← PC

IF CD[2] = 1 and DECODER = 00 Then MDR ← MEMORY[MAR]

IF CD[3] = 1 and DECODER = 00 Then IR ← MDR