Homework 2 : Due Wed Sept. 26

1. For the Edu32 processor organization and control signals given in Lecture Notes #6b, design the CU FSM after the DECODE-&-INCR.-PC phase to execute the following instructions, and calculate the number of cc’s taken to execute each instruction not counting the INSTR. FETCH and DECODE-&-INCR.-PC phases:

(a) \([\text{ADD } r_i \ r_j \ r_k]\). Semantics: \(r_i \leftarrow r_j + r_k\).  
(b) \([\text{JMP 24-bit-offset } Y]\). Semantics: \(PC \leftarrow PC + Y\).  
(c) \([\text{DEC_BPL } r_j \ 16\text{-bit-offset } X]\). Semantics: Perform \(r_j \leftarrow r_j - 1\) and branch to address \(PC + X\) (i.e., \(PC \leftarrow PC + X\)) if the result of the subtraction is non-negative (i.e., if \(r_j - 1 \geq 0\)). Note that the sign of the result is indicated by the \(m_{31}\) status signal going to the C.U., and if \(r_j\) is non-negative before the decrement then \(m_{31}\) is a correct indication of the sign of the decremented \(r_j\) since there could not have been any overflow (overflow causes the sign of the result to be incorrect).  
(d) \([\text{LI } r_i \ 16\text{-bit-constant } Z]\). Semantics: \(r_i \leftarrow Z\).  
(e) \([\text{LW+ } r_i \ (r_j) \ 16\text{-bit-offset } X]\). This is a load word instruction using the auto-increment register-indirect addressing mode. Semantics: \(r_i \leftarrow \text{Mem}[r_j]; r_j \leftarrow r_j + X\).  
(f) \([\text{SW+ } (r_j) \ r_k \ 16\text{-bit-offset } X]\). This is a store word instruction using the auto-increment register-indirect addressing mode. Semantics: \(\text{Mem}[r_j] \leftarrow r_k; r_j \leftarrow r_j + X\).

Assume the following: (1) the memory is byte addressable; (2) all instructions are 32 bits; (3) all data are 32 bits; (4) the ALU has the following FUs: ADD, ADDA (increment for Bus A operand), SUB, SUBA (decrement for Bus A operand), OR, AND, NOPA, (pass through the ALU unchanged for Bus A operand), NOPB (pass through the ALU unchanged for Bus B operand) (you can assign these symbols to the \(alu\_sel\) control signals to perform the corresponding ALU operations); (5) all ALU FUs take 1 cc; (6) the register file has 16 registers. \(r_0\) to \(r_{11}\) are addressable by the assembly language programmer (i.e., the \(r_i, r_j, r_k\) fields have 4 bit addresses that range from only 0000 to 1011), while \(r_{12}\) to \(r_{15}\) are for scratch-pad use of the CU).

Please submit a copy of Edu32 with control signals from Lecture Notes #6b with your homework.

2. (a) Using the instructions of problem (1) as well as the instruction \([\text{LW } r_i \ 16\text{-bit-offset } X]\) discussed in class, write an assembly language program with comments to perform the following task given below in a structured high-level code:

\[
\text{integer } a[0..99], b[0..99], c[0..99] /* declaration of 3 integer arrays, each with 100 integers */
\]

\[
\text{for } i = 0 \text{ to } 99 \text{ begin}
\]
\[
c[i] = a[i] + b[i];
\]
\[
\text{end}
\]

Assume that your program starts from address 0 and that the \(a\), \(b\) and \(c\) arrays are stored starting from addresses 100, 500 and 900 (in decimal), respectively.

(b) Assuming that the INSTR. FETCH phase takes 4 cc’s and that the DECODE-&-INCR.-PC phase takes 1 cc, calculate the total number of cc’s taken to execute your assembly language code. If the clock speed is 250 MHz, how much time does your program take to execute?