12.14.5

a) ALU Op = 1 (for register immediate operations)

Funct = 2 (subtraction operation)

RegDst = 0 (register is specified)

Branch = 0 (no branch instruction)

MemRead = 0 (doesn’t use memory reading)

MemWrite = 1 (instruction uses memory writing)

b) 0xadac001A: $t4 contains what is about to be stored, $t5 holds the memory address, the offset is therefore 20, making the word placed at $t5+20, it updates to the next instruction and repeats, making the next instruction at the location 0xadac001A

c) Each time an instruction is fetched the PC’s counter is increased by 4, through using the PC’s program counter. When the PC advanced by one position, it increments by four, making the value equal to the address of the instruction plus 4

d)

MUX1: Inputs: PC+4, branch target Output:

PC MUX2: Inputs: instruction[25:0] (0xadac0016), PC+4 Output: ReadData

MUX3: Inputs: Reg[rs], Reg[rt] Output: ALUInput1

MUX4: Inputs: ReadData, SignExtended Output: ALUInput2

MUX5: Inputs: Reg[rd], Zero Output: ALUInput2

MUX6: Inputs: ALUOutput, MemoryData Output: WriteData

e) $t4 & $t5 are the register outputs because the date from the register file outputs from Reg [xn]

12.14.7

a) Latency of an R-type instruction = 30 + 250 + 150 + 25 + 200 + 25 + 20 = 700 ps

b) Latency of lw = 30 + 250 + 150 + 25 + 200 + 250 + 25 + 20 = 950 ps

c) Latency of sw = 30 + 250 + 150 + 200 + 25 + 250 = 905 ps

d) Latency of beq = 30 + 250 + 150 + 25 + 200 + 5 + 25 + 20 = 705 ps

e) Latency of I-type instruction = 30 + 250 + 150 + 25 + 200 + 25 + 20 = 700 ps

f) Minimum clock period for this CPU = 950 ps

12.14.10

a) Clock cycle (normal) = 250 + 150 + 25 + 200 + 150 + 5 + 30 + 20 + 50 + 50 = 930 ps

Clock cycle (improved) = 250 + 160 + 25 + 200 + 150 + 5 + 30 + 20 + 50 + 50 = 940 ps

35-35\*0.12 = 30.8 🡪 31

Total number of instructions = 52 + 31 + 11 + 2 = 96

Speed = (930/100)/(940/96) = 0.95

b) Cost (normal) = 1000 + 200 + 10 + 100 + 30 + 2000 + 5 + 100 + 1 + 500 = 3946

Cost (improved) = 100 + 400 + 10 + 100 + 30 + 2000 + 5 + 100 + 1 + 500 = 4146

Cost (ratio) = (4146/940)/(3946/930) = **1.03**

Performance (normal) = 1/(100\*930\*10-12) = 10.75 \* 106

Performance (improved) = 1/(96\*940\*10-12) = 11.08 \* 106

Performance (ratio) = 11.08/10.75 = **1.03**

Therefore, the cost and performance ratios are the same, as shown above.

c) When performance improvement is important and not cost, adding more registers would be the best case (since registers increase cost and performance). However, you would NOT add registers if you want it to be more cost-efficient rather than improve performance.

12.14.16

a) Clock cycle time (pipelined): 350 ps (slowest stage)

Clock cycle time (non-pipelined): 250 + 350 + 150 + 300 + 200 = 1250 ps

b) Latency of lw (pipelined): 5\*350 = 1750 ps

Latency of lw (non-pipelined): 250 + 350 + 150 + 300 + 200 = 1250 ps

c) I would split the slowest stage which is ID at 350 ps, making the new clock cycle time the second slowest stage, MEM at 300 ps.

d) Utilization of data memory = distribution of lw and sw instructions = 20% + 15% = 35% of the clock cycles

e) Utilization of the write-register port of the “Registers” unit = distribution of lw and alu instruction = 20% + 45% = 65% of the clock cycles