CE/CZ3001: Advanced Computer Architecture

Tutorial-1

1. Answer:

   Cycle count = 200 × 1 + 500 × 2 + 300 × 3 = 2100 clock cycles.
   Average CPI = (Cycle count)/(Instruction count).
   Instruction count = 200 + 500 + 300 = 1000 instructions.
   Average CPI = 2100/1000 = 2.1 clock cycles per instruction.
   Execution time = (Cycle count) × (Clock period) = 2100 × 100 ns = 210 microsecond.
   Execution time = IC × Average CPI × Tc= 1000 × 2.1 ×100ns =210 microsecond.

2. Answer:

   (a) Average CPI = execution time in seconds × clock frequency in Hz / instruction count.
   Average CPI of P1 on M1= (10 x 200 × 10^6) /200 × 10^6 = 10.
   Average CPI of P1 on M2= (5 x 300 × 10^6) /160 × 10^6 = 9.375.
   (b) IC= execution time in seconds × clock frequency in Hz / average CPI.
   Instruction count of P2 when runs on M1= (4 × 200 × 10^6) /10 = 80 × 10^6 instructions.
   Instruction count of P2 when runs on M2= (3 × 300 × 10^6) /9.375 = 96 × 10^6 instructions.

3. Answer:

   (a) Clock period of M1 and M2 = [1/(500 × 10^6)] seconds.
   Clock period of M3 = [1/(750 × 10^6)] seconds.
   Execution time of a program = no. of clock cycles required by the program × duration of clock period.
   Execution time of P1 on M2 = (10*N*/4) × [1/(500 × 10^6)] seconds.
   Execution time of P1 on M3 = (12*N*/4) × [1/(750 × 10^6)] seconds.
   Speedup of M3 over M2 = (Exe. time of P1 on M2)/(Exe. time of P1 on M3) = {(10*N*/4) × [1/(500 × 10^6)]}/{(12*N*/4) × [1/(750 × 10^6)]}= (5/6) × (750/500) = 1.25
   (b) If all instructions of program P2 are of category A, then the no. of clock cycles required for the program on M1, M2, and M3 are, respectively, 2*N, N*, and 2*N*.
   Speedup of M2 over M1 = 2*N*/*N* = 2
   (*Note: It is the ratio of no. of clock cycles required, since M1 and M2 have the same clock frequency*.)
   Speedup of M3 over M1 = [1/(500 × 10^6)]/[1/(750 × 10^6)] = 750/500 = 1.5
   (*Note: It is the ratio of clock frequencies, since in both M1 and M3 the cycle count is the same.*)

Tutorial-2

1. Case-1:

   (i) Dynamic power consumption
   P1(dyn) = A × C × 3.0 × 3.0 × f1,
   P2(dyn) = A × C × 3.3 × 3.3 × f2,
   f2/f1 = 3.3/3.0 = 1.1
   P2(dyn)/P1(dyn) = (3.3 × 3.3 × f2)/(3.0 × 3.0 × *f*1) = (1.21) × (1.1) = 1.331 => 33.1% increase in dynamic power consumption.

(ii) Static power consumption

P1(st) = 3.0 × Ileak

P2(st) = 3.3 × Ileak

P2(st)/P1(st) = (3.3/3.0) = 1.1

10% increase in static power consumption.

(iii) Performance

Speedup due to higher voltage = Tc1/Tc2 = f2/f1 = 3.3/3.0 = 1.1 Performance increases by a factor of 1.1.

(iv) Energy consumption

E2(st)/E1(lst) = [P2(st)/P1(st)](Tc2/Tc1) = 1.1/1.1 = 1

No change in static energy consumption

E2(dyn)/E1(dyn) = (P2(dyn)×Tc2)/(P1(dyn)×Tc1) = 1.331 / 1.1 = 1.21 => 21% increase in dynamic energy consumption.

Answer Case-2:

(i) Dynamic power consumption

P1(dyn) = A × C × 3.0 × 3.0 × f,

P2(dyn) = A × C × 3.3 × 3.3 × f,

P2(dyn)/P1(dyn) = (3.3 × 3.3 × f)/(3.0 × 3.0 × f) = 1.21 => 21% increase in dynamic power consumption.

(ii) Static power consumption

P1(st) = 3.0 × Ileak

P2(st) = 3.3 × Ileak

P 2(st)/P1(st) = (3.3/3.0) = 1.1=> 10% increase in static power consumption.

(iii) Performance

Speedup due to higher voltage = Tc1/Tc2 = f /f = 1 No change in performance.

(iv) Energy consumption

E2(st)/E1(lst) = [P2(st)/P1(st)](Tc2/Tc1) = 1.1

10% increase in static energy consumption

E2(dyn)/E1(dyn) = (P2(dyn) × Tc2)/(P1(dyn) × Tc1) = 1.21 => 21% increase in dynamic energy consumption.

2.

(i) The LEGv8 code:

```
        ADDI  X2, X31, #101          # save loop termination index
loop:LDUR  X4,  [X11, #0]            # X4= b[i]
        ADD  X4,  X4,   X1           # X4 = b[i] = b[i] + c
        STUR   X4,  [X10, #0]        # store X4 to address of a[i]
        ADDI X10, X10,  #8           # X10 = address of a[i+1]
        ADDI X11, X11,  #8           # X11= address of b[i+1]
        SUBI  X2,  X2,   #1          # X2 = X2 - 1
        CBNZ   X2,    loop           # if (X2 = 0) go to finish
```

finish

(ii) 1 instruction is executed before the loop is executed: executed once. Loop body consists of 7 instructions: executed 101 times. Therefore, total no. of instructions executed: = 708.

(iii) Memory data reference: 101 × 2 = 202.

Tutorial-3

1. Answer:

   (i) **PC increment circuit** is used for ALL instructions.
   (ii) **Instruction memory** is used for ALL instructions.
   (iii) **Register file** is used for all instructions except 'unconditional branch'.
   (iv) **ALU** is used for R-type ALU instructions, immediate ALU instructions, load, store and conditional branch instructions.
   (v) **Data memory** is used for load and store instructions.

2. Answer:

   (i) R-format instructions: Imem + mux (before regfile) + Reg(R)+ Mux(before alu)+ ALU+ mux (wb)+ Reg(W)=
   500 +50+ 200 + 50 + 2000 + 50 + 200 = 3050 ps
   (ii) I-format ALU instructions: Imem + Reg(R)+ ALU+ mux (wb)+ Reg(W)=
   500 + 200 + 2000 + 50 + 200 = 2950 ps
   (iii) D- format-load word (LDUR): Imem + Reg(R)+ ALU+ Dmem+ mux (wb)+ Reg(W)=
   500 + 200 + 2000 + 2000 + 50 + 200 = 4950 ps
   (iv) D-format- store word (STUR): Imem + Reg(R)+ ALU+ Dmem=
   500 + 200 + 2000 + 2000 = 4700 ps
   (v) B format- conditional branch (CBZ): Imem +mux(before regfile)+ Reg(R)+ mux(beforealu) + ALU+ AND gate + OR gate +mux(branch)+Pcin->PCout=
   500 + 50+ 200 +50+ 2000 + 0+0+ 50 + 100 = 2950 ps
   (vi) B-format- unconditional branch (B): Imem + signext+ shift left 2+ Add+ mux(branch)+Pcin->PCout=500+25+0+1500+50+100=2175 ps

3. Answer:

   (i) 4950ps
   (ii) 2000ps

Tutorial 4

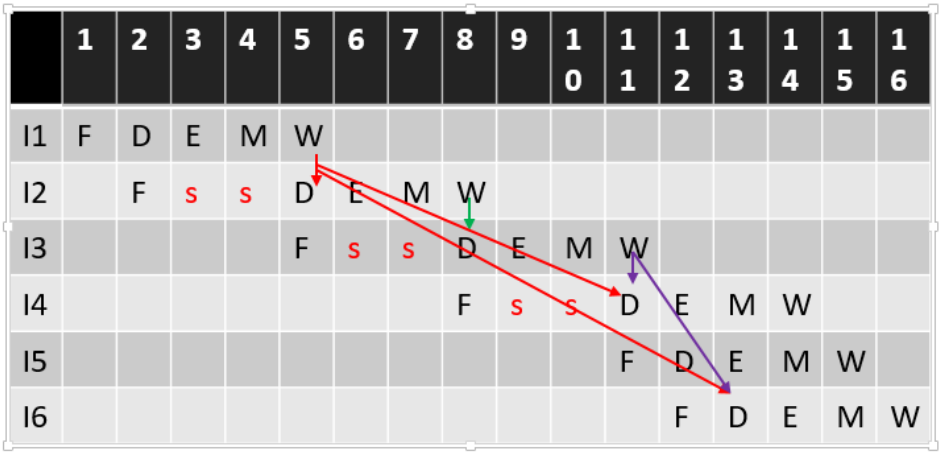
1. Answer:

   a) 500ps for pipelined processor, 1650ps for nonpipelined processor.
   b) 2500ps for pipelined processor, 1650ps for nonpipelined processor.
   c) Longest delay 500ps (MEM) should be split. Then the new cycle time is 400ps. d)
      1) LDUR and STUR use the data memory. So the data memory utilization is 15% + 10% = 25%.
      2) LDUR and ALU will write back to the register, so the register write-port utilization is 50%+15% = 65%.

2. Answer:

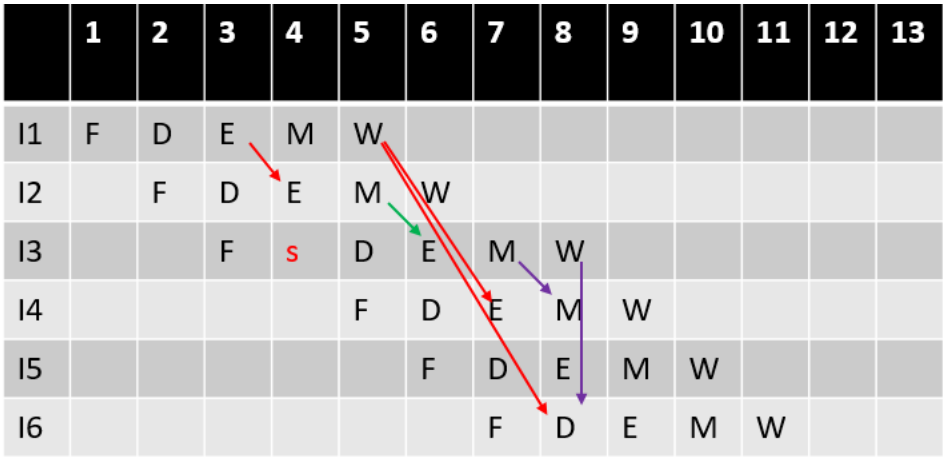
   a) 6 stalls are needed to solve the hazards without data forwarding.
   Data dependency:
   RAW: I1, I2 at X0; I1, I4 at X0; I1, I6 at X0, I2, I3 at X2, I3, I6 at X3 and I3 and I4 at X3

WAW: I5, I6 at X5



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
I1	F	D	E	M	W											
I2		F	s	s	D	E	M	W								
I3					F	s	s	D	E	M	W					
I4								F	s	s	D	E	M	W		
I5											F	D	E	M	W	
I6												F	D	E	M	W

b) Arrow indicates the data forwarding, only one cycle stall required.



	1	2	3	4	5	6	7	8	9	10	11	12	13
I1	F	D	E	M	W								
I2		F	D	E	M	W							
I3			F	s	D	E	M	W					
I4					F	D	E	M	W				
I5						F	D	E	M	W			
I6							F	D	E	M	W		

c) In case of no forwarding:
CPI = (No of instructions + no of stall)/No. of instruction=12/6 = 2
In case of full forwarding:
CPI = (6+1)/6 = 1.16

3. Answer:

a) The schedule is like the following:

Time		1	2	3	4	5	6	7	8	9	10
P1	Lane1	A	B	C	E	G	H				
	Lane2			D	F						
P2	Lane1	A	B	C	G	H					
	Lane2	D	E	F							

b) 1.2. Execution time of P1 / execution time of P2 = 6/5 = 1.2.