**IT5002 Computer Systems and Applications**

**Tutorial 4 – Suggested Solutions**

1. Suppose the four stages in some 4-stage pipeline take the following timing: 2ns, 3ns, 4ns, and 2ns. Given 1000 instructions, what is the speedup (in two decimal places) of the pipelined processor compared to the non-pipelined single-cycle processor?

***Answer:***

Non-pipelined: 11ns × 1000 = 11,000ns.
In a pipelined system, each stage takes 4ns.
Pipelined: 12ns + (1,000 × 4) ns = 12 ns + 4,000ns = 4,012ns.
Speedup = 11,000/4,012 = **2.74**

2. Let's try to understand pipeline processor by doing a detailed trace. Suppose the pipeline registers (also known as pipeline latches) store the following information:

| IF / ID | |
|---|---|
| No Control Signal | ------ |
| PC+4 | |
| OpCode | |
| Rs | |
| Rt | |
| Rd | |
| Funct | |
| Imm(16) | |

| ID / EX | |
|---|---|
| MToR | |
| RegWr | |
| MemRd | |
| MemWr | |
| Branch | |
| RegDst | |
| ALUsrc | |
| ALUop | |
| PC+4 | |
| ALUOpr1 | |
| ALUOpr2 | |
| Rt | |
| Rd | |
| Imm(32) | |

| EX / MEM | |
|---|---|
| MToR | |
| RegWr | |
| MemRd | |
| MemWr | |
| Branch | |
| BrcTgt | |
| isZero? | |
| ALURes | |
| ALUOpr2 | |
| DstRNum | |

| MEM / WB | |
|---|---|
| MToR | |
| RegWr | |
| MemRes | |
| ALURes | |
| DstRNum | |

Show the progress of the following instructions through the pipeline stages by filling in the content of pipeline registers.

i. `0x8df80000 # lw $24, 0($15) #Inst.Addr = 0x100`

ii. `0x1023000C # beq $1, $3, 12 #Inst.Addr = 0x100`

iii. `0x0285c822 # sub $25, $20, $5 #Inst.Addr = 0x100`

Assume that registers 1 to 31 have been initialized to a value that is equal to 101 + its register number. i.e. [$1] = 102, [$31] = 132 etc. You can put "X" in fields that are irrelevant for that instruction. Do note that in reality, these fields are actually generated but not utilized.

Part (i) has been worked out for you.

i. `0x8df80000 # lw $24, 0($15) #Inst.Addr = 0x100`

| IF / ID | |
|---|---|
| No Control Signal | ------ |
| PC+4 | 0x104 |
| OpCode | 0x23 |
| Rs | $15 |
| Rt | $24 |
| Rd | X |
| Funct | X |
| Imm(16) | 0 |

| ID / EX | |
|---|---|
| MToR | 1 |
| RegWr | 1 |
| MemRd | 1 |
| MemWr | 0 |
| Branch | 0 |
| RegDst | 0 |
| ALUsrc | 1 |
| ALUop | 00 |
| PC+4 | 0x104 |
| ALUOpr1 | 116 |
| ALUOpr2 | X |
| Rt | $24 |
| Rd | X |
| Imm(32) | 0 |

| EX / MEM | |
|---|---|
| MToR | 1 |
| RegWr | 1 |
| MemRd | 1 |
| MemWr | 0 |
| Branch | 0 |
| BrcTgt | X |
| isZero? | X |
| ALURes | 116 |
| ALUOpr2 | X |
| DstRNum | $24 |

| MEM / WB | |
|---|---|
| MToR | 1 |
| RegWr | 1 |
| MemRes | Mem(116) |
| ALURes | X |
| DstRNum | $24 |

***Answers:***

ii. `0x1023000C # beq $1, $3, 12 #Inst.Addr = 0x100`

| IF / ID | |
|---|---|
| No Control Signal | ------ |
| PC+4 | 0x104 |
| OpCode | 4 |
| Rs | $1 |
| Rt | $3 |
| Rd | X |
| Funct | X |
| Imm(16) | 12 |

| ID / EX | |
|---|---|
| MToR | X |
| RegWr | 0 |
| MemRd | 0 |
| MemWr | 0 |
| Branch | 1 |
| RegDst | X |
| ALUsrc | 0 |
| ALUop | 01 |
| PC+4 | 0x104 |
| ALUOpr1 | 102 |
| ALUOpr2 | 104 |
| Rt | X |
| Rd | X |
| Imm(32) | 12 |

| EX / MEM | |
|---|---|
| MToR | X |
| RegWr | 0 |
| MemRd | 0 |
| MemWr | 0 |
| Branch | 1 |
| BrcTgt | 0x134 |
| isZero? | 0 |
| ALURes | -2 |
| ALUOpr2 | X |
| DstRNum | X |

| MEM / WB | |
|---|---|
| MToR | X |
| RegWr | 0 |
| MemRes | X |
| ALURes | X |
| DstRNum | X |

iii. `0x0285c822 # sub $25, $20, $5 #Inst.Addr = 0x100`

| IF / ID | |
|---|---|
| No Control Signal | ------ |

| ID / EX | |
|---|---|
| MToR | 0 |
| RegWr | 1 |
| MemRd | 0 |
| MemWr | 0 |
| Branch | 0 |

| EX / MEM | |
|---|---|
| MToR | 0 |
| RegWr | 1 |
| MemRd | 0 |
| MemWr | 0 |

| MEM / WB | |
|---|---|
| MToR | 0 |
| RegWr | 1 |

| | |
|---|---|
| PC+4 | 0x104 |
| OpCode | 0 |
| Rs | $20 |
| Rt | $5 |
| Rd | $25 |
| Funct | 22 |
| Imm(16) | X |

| | |
|---|---|
| RegDst | 1 |
| ALUsrc | 0 |
| ALUop | 10 |
| PC+4 | 0x104 |
| ALUOpr1 | 121 |
| ALUOpr2 | 106 |
| Rt | X |
| Rd | $25 |
| Imm(32) | X |

| | |
|---|---|
| Branch | 0 |
| BrcTgt | X |
| isZero? | X |
| ALURes | 15 |
| ALUOpr2 | X |
| DstRNum | $25 |

| | |
|---|---|
| MemRes | X |
| ALURes | 15 |
| DstRNum | $25 |

3. Given the following three formulas (See Lecture #9, Section 5 Performance):

$$CT_{seq} = \sum_{k=1}^{N} T_k$$

$$CT_{pipeline} = \max(T_k) + T_d$$

$$Speedup_{pipeline} = \frac{CT_{seq} \times InstNum}{CT_{pipeline} \times (N + InstNum - 1)}$$

For each of the following processor parameters, calculate $CT_{seq}$, $CT_{pipeline}$ and $Speedup_{pipeline}$ (to two decimal places) for 10 instructions and for 10 million instructions.

| | Stages Timing (for 5 stages, in ps) | Latency of pipeline register (in ps) |
|---|---|---|
| a. | 300, 100, 200, 300, 100 (slow memory) | 0 |
| b. | 200, 200, 200, 200, 200 | 40 |
| c. | 200, 200, 200, 200, 200 (ideal) | 0 |

***Answers:***

| | $CT_{seq}$ | $CT_{pipeline}$ | Speedup (10 inst) | Speedup (10m inst) |
|---|---|---|---|---|
| a. | **1000ps** | **300ps** | (1000×10)/(300×14)<br>**= 2.38** | (1000×10m)/(300 (10m+4))<br>**= 3.33** |
| b. | **1000ps** | **240ps** | (1000×10)/(240×14)<br>**= 2.98** | (1000×10m)/(240×(10m+4))<br>**= 4.17** |
| c. | **1000ps** | **200ps** | (1000×10)/(200×14)<br>**= 3.57** | (1000×10m)/(200×(10m+4))<br>**= 5.00** |