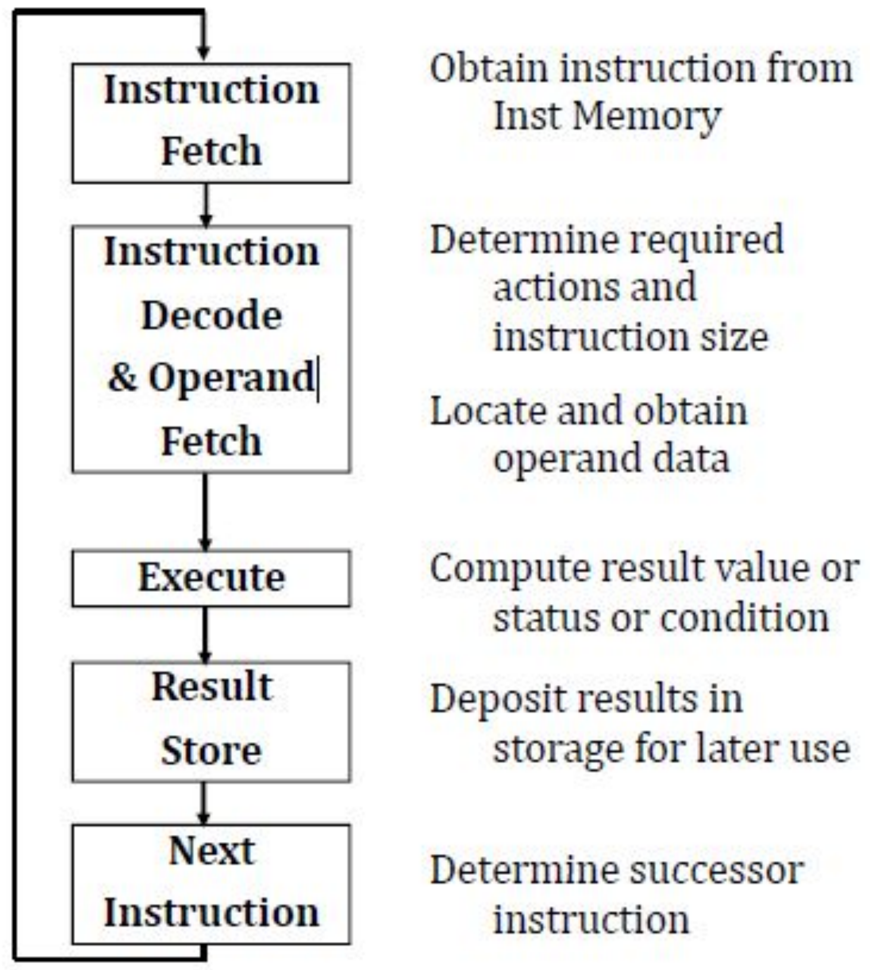
Pipelining

Prepared By: Aatira Anum

Fundamental Execution Cycle


Instruction Fetch
Instruction Decode & Operand Fetch
Execute
Result Store
Next Instruction
Obtain instruction from Inst Memory
Determine required actions and instruction size
Locate and obtain operand data
Compute result value or status or condition
Deposit results in storage for later use
Determine successor instruction

Performance

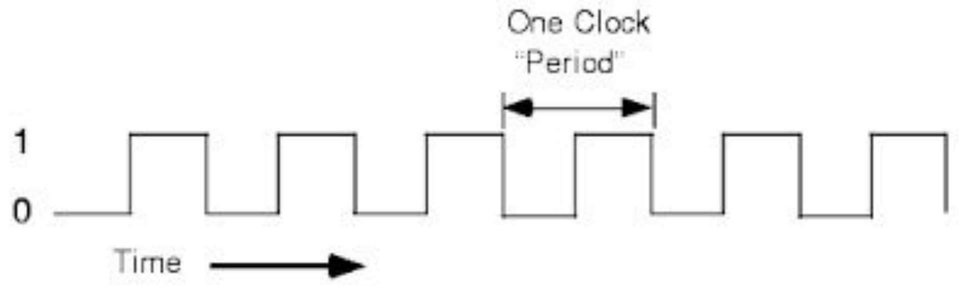
- If one instruction takes 3 nanoseconds to execute the cycle then 3 instructions will take 9 nanoseconds.
- 5 instructions will take 15 nanoseconds.
- 10 instructions will take 30 nanoseconds
- 20 instructions will take 60 nanoseconds.

How to improve Performance?

- Increase Clock Frequency of CPU
- Internal clock to synchronize all operations – ensures all circuits inside a computer works together at same time
- Clock Cycle Time: Amount of time between 2 adjacent pulses of an oscillator
- Single increment of the CPU clock during which the smallest unit of processor activity is carried out
- If one instruction cycle takes 3ns then clock cycle time will be of 3ns
- Clock Frequency: Number of clock cycles CPU executes per second

Clock Frequency

- Measured In Hertz.
- 1 Hertz = 1 clock cycle/second
- 1 MHz = 10^6 clock cycles per second
- 1 GHz = 10^9 clock cycles per second
- Indicator of Processor speed



Clock Frequency

- What is the clock frequency of the processor if one clock cycle takes 10 ns?

- Clock frequency = 1/clock cycle time

  $= 1/(10 \times 10^{-9})$

  $= 100000000$ Hz

  $= 0.1$ GHz

Pipelining

- Multiple instructions are overlapped in a single execution
- It takes advantages of the parallelism that exists among the actions needed to execute an instruction
- Key implementation technique to make CPU faster

Pipelining

- A pipeline is just like an assembly line
- In an automobile manufacturing line, there are many steps each contributing something to the construction of the car
- Each step operates in parallel with other steps although with a different car

Real World Analogy

- Automobile manufacturing line without pipelining
- Assemble (A), Paint (P), Fix tires (T)
- Assuming A+P+T take 3 hours

Time line	0-3 hours	3-6 Hours	6 -9Hours
Car 1	A+P+T		
Car 2		A+P+T	
Car 3			A+P+T

Real World Analogy

- Automobile manufacturing line with pipelining, in 3 stages
- Assemble (A), Paint (P), Fix tires (T)
- Assuming each stage takes 1 hour

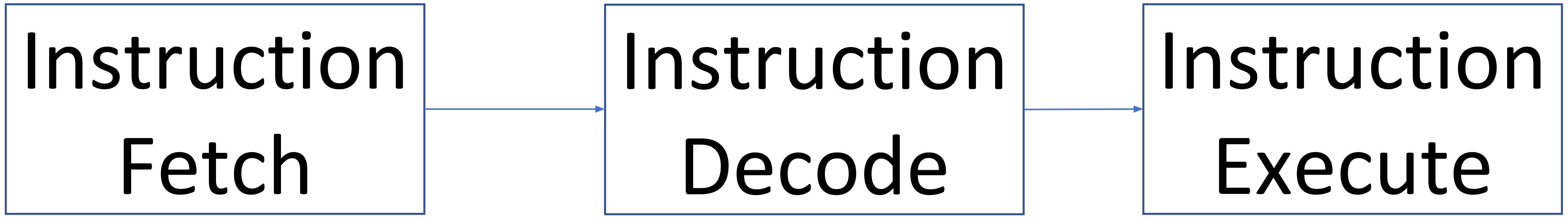
Time line	1st hour	2nd hour	3rd hour	4th hour	5th hour	6th hour	7th hour	8th hour	9th hour
Car 1	A	P	**T**						
Car 2		A	**P**	T					
Car 3			**A**	P	T				

- It takes 5 hours with pipelining and 9hours without pipelining
- Speedup = 9/5

Stages in Execution Cycle

- Different architectures divide the execution cycle in different number of stages

- E.g., a 3 stage pipeline will be like this



Stages in Execution Cycle

- Fetch - Obtain Instruction from Memory
- Decode – Determine required action and obtain operands
- Execute – Execute the instruction and store the result

Performance of Pipelined Processor

- It takes 3 nanoseconds to execute the whole cycle
- Suppose each stage takes 1 nanoseconds
- How much time it will take to complete 3 instructions?

Performance of Pipelined Processor

	1 ns	2ns	3ns	4ns	5ns
1st instruction	**F**	**D**	**E**		
2nd instruction		**F**	**D**	**E**	
3rd instruction			**F**	**D**	**E**

Performance of Pipelined Processor

- Total time taken to execute 3 instructions in pipelined processor = 5ns
- Total time taken to execute 3 instructions in non-pipelined processor = 9ns
- Speedup = 9/5 = 1.8

Latency and Throughput

- **Latency – time taken to complete an instruction**

  Non-pipelined = 3ns

  Pipelined = 3ns

- **Throughput – number of instructions executed per unit time**

  Non-pipelined = 3/9 = 0.3 instructions/ns

  Pipelined = 3/5 = 0.6 instructions/ns

- Observation : Observation: Pipeline does not improve latency, it only improves throughput

Performance of Pipelined Processor

- Assumption
  - All stages take equal time.
- It takes 3 nanoseconds to execute the whole cycle
- Suppose Fetch stage takes 0.5 nanoseconds, Decode stage takes 2 nanoseconds and Execute stage takes 0.5 nanoseconds.
- How much time it will take to complete 3 instructions?

Performance of Pipelined Processor

1st instruction	**0-0.5**	**0.5-2.5**	**2.5-3**		
2nd instruction		**0.5-2.5**	**2.5-4.5**	**4.5-5**	
3rd instruction			**2.5-4.5**	**4.5-6.5**	**6.5-7**

Fetch stage takes 0.5 nanoseconds,
Decode stage takes 2 nanoseconds
Execute stage takes 0.5 nanoseconds.

Latency and Throughput

- Pipelined with all stages not of equal time
- Total time taken = 7ns
- Latency = 4.5 ns
- Throughput = 3/7 instructions/ns

Performance of Pipelined Processor

- Assumption
  - There is no time taken to move from one stage to another.

- Pipeline buffers stores the output of each stage.

- It takes time for pipeline buffers to operate and this adds to the instruction cycle time.

- Latch time/delay is the time taken to transition from one stage to another.

- Suppose in the previous example, the latch time is 0.2 nanoseconds, how much time is taken to execute 3 instructions?

Performance of Pipelined Processor

		■		■		■		■	
1st	**F**	■	**D**	■	**E**	■		■	
2nd		■	**F**	■	**D**	■	**E**	■	
3rd		■		■	**F**	■	**D**	■	**E**

Fetch stage takes 0.5 nanoseconds,
Decode stage takes 2 nanoseconds
Execute stage takes 0.5 nanoseconds.
Latch time is 0.2 nanoseconds.

Latency and Throughput

- Pipelined with all stages not of equal time and latch time

- Total time taken = 7 + 0.2 x4 = 7.8 ns

- Latency = 4.5 + 0.2 x2 = 4.9 ns

- Throughput = 3/7.8 instructions/ns

Throughput and Latency for 3 instructions

Case	Latency	Throughput	Speed up
Non pipeline	3	3/9=0.33	NA
Pipeline ideal case	3	3/5=0.6	9/5
Pipeline with all stages not of equal time	4.5	3/7=0.42	9/7
Non ideal case with latch time	4.9	3/7.8=0.38	9/7.8

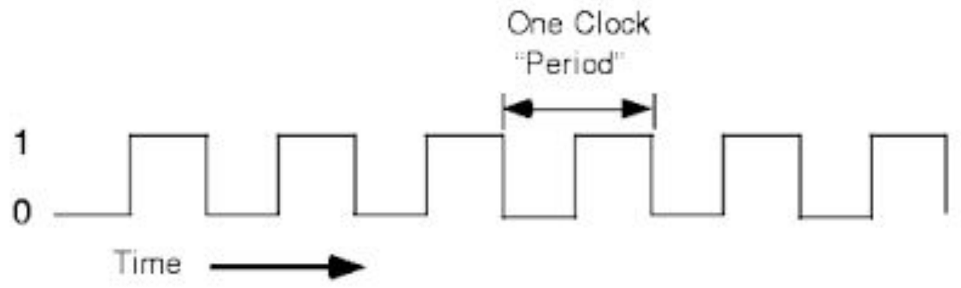
Clock Cycle

- Internal clock to synchronize all operations – ensures all circuits inside a computer works together at same time

- Clock Cycle Time: Single increment of the CPU clock during which the smallest unit of processor activity is carried out

- In processor with no pipelining
  - If one instruction cycle takes 3ns then clock cycle time should be of 3ns

- In ideal pipelining case
  - If each stage takes 1 ns
  - Clock cyle time should be of 1ns



- In pipelining with different stages taking different time
  - F takes 0.5 ns, D takes 2 ns E takes 0.5ns
  - Clock cyle time should be of 2 ns
  - The stage with maximum time will be the time taken for each stage
  - Each stage will complete in 1 clock cycle time

Clock Cycle

- Each stage will complete in 1 clock cycle time so the time for each stage will be 2 ns.

1st instruction	**0-2**	**2-4**	**4-6**		
2nd instruction		**2-4**	**4-6**	**6-8**	
3rd instruction			**4-6**	**6-8**	**8-10**

Latency and Throughput

- Total time taken = 10 ns
- Latency = 6 ns
- Throughput = 3/10 instructions/ns (this is worse than non-pipelined version)
- This issue can be resolve if stages are more or less of equal time duration (decode stage can be divided into further stages)

Pipeline with 5 stages

- **Fetch instruction (FI):** Read the next expected instruction into a buffer.
- **Decode instruction (DI):** Determine the opcode and the operand specifiers.
- **Fetch operands (FO):** Fetch each operand.
- **Execute instruction (EI):** Perform the indicated operation.
- **Write operand (WO):** Store the result in register or memory.

Pipeline in Computer:: Stages

- Five-stage pipeline can reduce the execution time for 3 instructions from 15 clock cycles to 7 clock cycles.
- Throughput with pipeline= 3/7
- Throughput without pipeline= 3/15
- Speedup= 15/7

Instruction 1	FI	DI	FO	EI	WO		
Instruction 2		FI	DI	FO	EI	WO	
Instruction 3			FI	DI	FO	EI	WO

Pipeline in Computer:: Performance Ideal case (Derivations)

- If
  - All stages take equal amount of time T
  - Latch time=0
  - Stages =k
  - Number of instructions = n
- Then
  - Clock cycle of pipeline=T, clock cycle of non pipeline= k*T
  - Frequency of pipeline=1/T , Frequency of non pipeline=1/k*T
  - Time taken to complete n instructions without pipeline= n*k*T= n*clock cycle of non pipeline
  - Through put for n instructions without pipeline= n/ n*k*T
  - Time taken to complete n instructions with pipeline= (k+n-1)*T= (k+n-1)*(Clock Cycle of pipeline)
  - Through put for n instructions with pipeline= n/ (k+n-1)*T
  - Speedup for n instructions = n*k*T/ (k+n-1)*T = n*k/ (k+n-1)
  - Latency without pipelining = k*T
  - Latency with pipelining= k*T

Pipeline in Computer:: Performance Non Ideal case (Derivations)

- If all stages do not take same time
  - Non pipeline processor takes T1 time to complete one instruction
  - In pipeline processor max time take by any stage is T2
  - Latch time=0
  - Stages =k
  - Number of instructions = n
- Then
  - Clock cycle of pipeline processor = T2, Clock Cycle of processor without pipelining = T1
  - Frequency of pipeline processor =1/T2, Frequency of processor without pipelining = 1/T1
  - Time taken to complete n instructions without pipeline= n*T1= n*clock cycle of non pipeline
  - Through put for n instructions without pipeline= n/ n*T1
  - Time taken to complete n instructions with pipeline= (k+n-1)*T2 =(k+n-1)*(Clock Cycle of pipeline)
  - Through put for n instructions with pipeline= n/ (k+n-1)*T2
  - Speedup for n instructions = n*T1/ (k+n-1)*T2
  - Latency without pipelining = T1
  - Latency with pipelining= K*T2

Pipeline in Computer:: Performance Non Ideal case with latch time (Derivations)

- If all stages do not take same time
  - Non pipeline processor takes T1 time to complete one instruction
  - In pipeline processor max time take by any stage is T2
  - Latch time=T3
  - Stages =k
  - Number of instructions = n
- Then
  - Clock cycle of pipeline processor = T2+T3, Clock Cycle of processor without pipelining = T1
  - Frequency of pipeline processor =1/(T2+T3), Frequency of processor without pipelining = 1/(T1)
  - Time taken to complete n instructions without pipeline= n*T1= n*clock cycle of non pipeline
  - Through put for n instructions without pipeline= n/ n*T1
  - Time taken to complete n instructions with pipeline= (k+n-1)*(T2+T3)= (k+n-1)*(Clock Cycle of pipeline)
  - Through put for n instructions with pipeline= n/ (k+n-1)*(T2+T3)
  - Speedup for n instructions = n*T1/ (k+n-1)*(T2+T3)
  - Latency without pipelining = T1
  - Latency with pipelining= K*(T2+T3)