Advance Computer Architecture and x86 ISA

University of Science and Technology of Hanoi

MS. LE Nhu Chu Hiep

Pipeline

How reduce the time for computing on data?

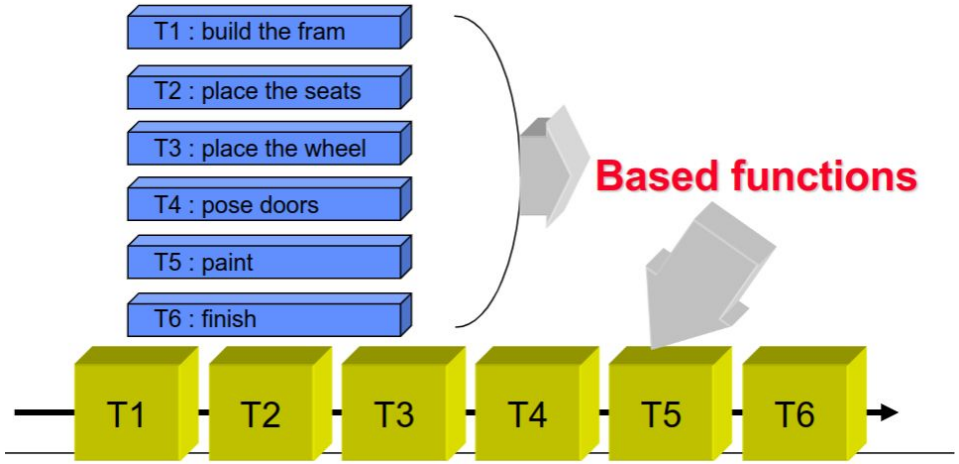
- Increase the clock frequency
- Parallel execution on several data

- Working on the chain

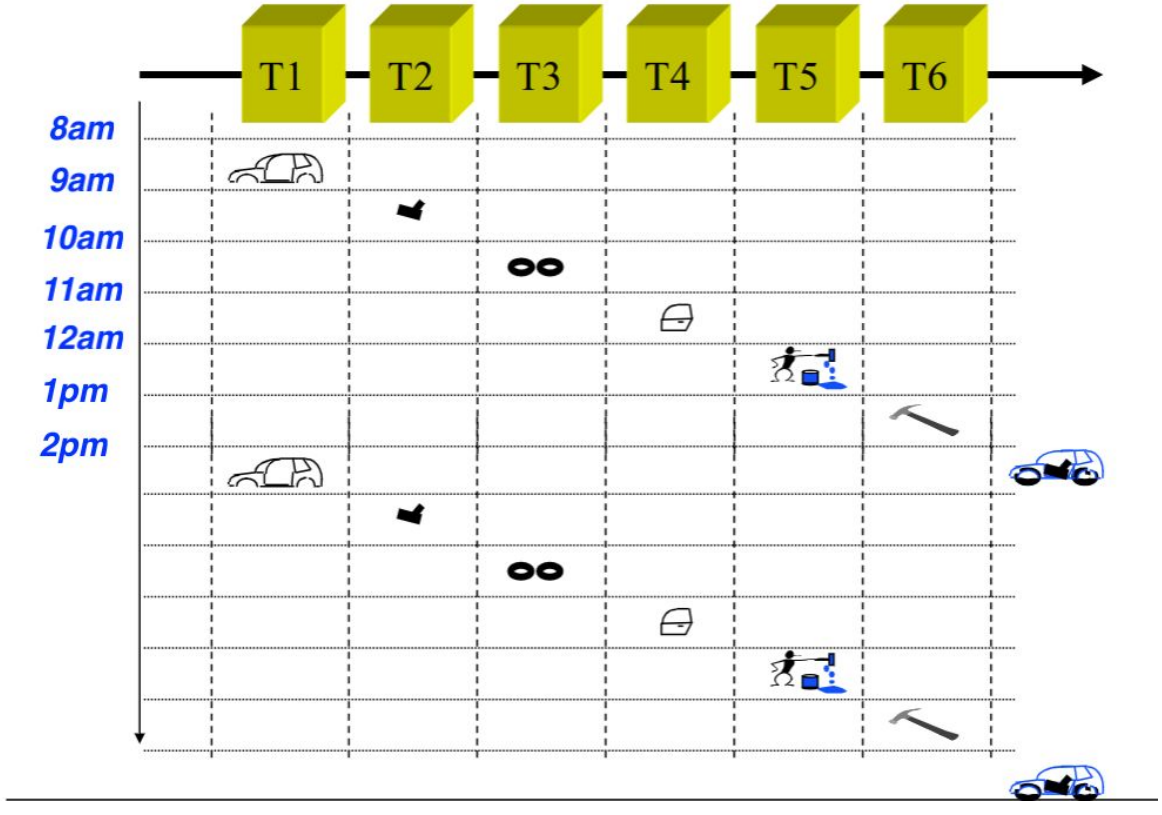
Assembly Line

Assembly line

- Task: building a car
- Composed of several sub-tasks
  - **T1:** Build the chase
  - **T2:** Place the seats
  - **T3:** Place the wheel
  - **T4:** Pose doors
  - **T5:** Paint
  - **T6:** Finish

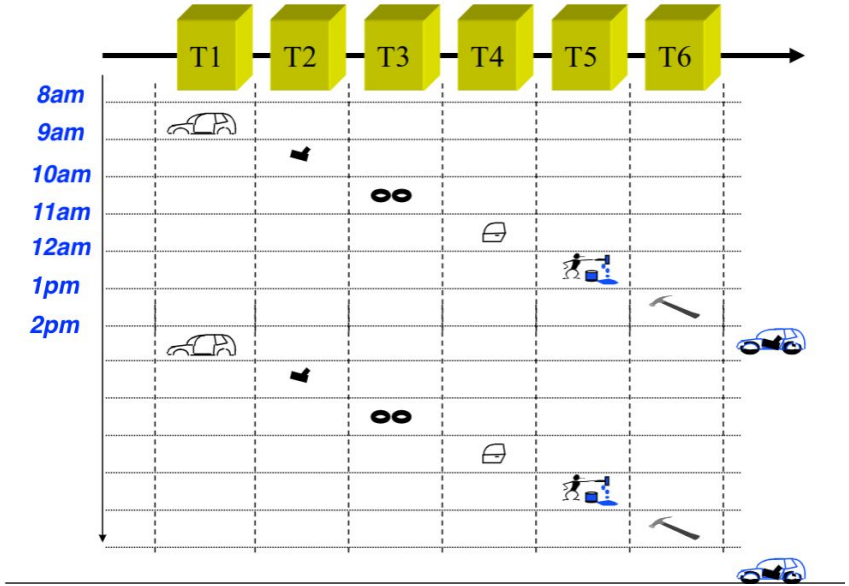


Building chain



Issues

- **Cost for this production technique**
  - 1 worker: Non specialist, no optimally, able to do everything
- **Time per car producing**
  - 6 UT (six unit time)
- **Production Cadence**
  - 1 new car each 6 time units

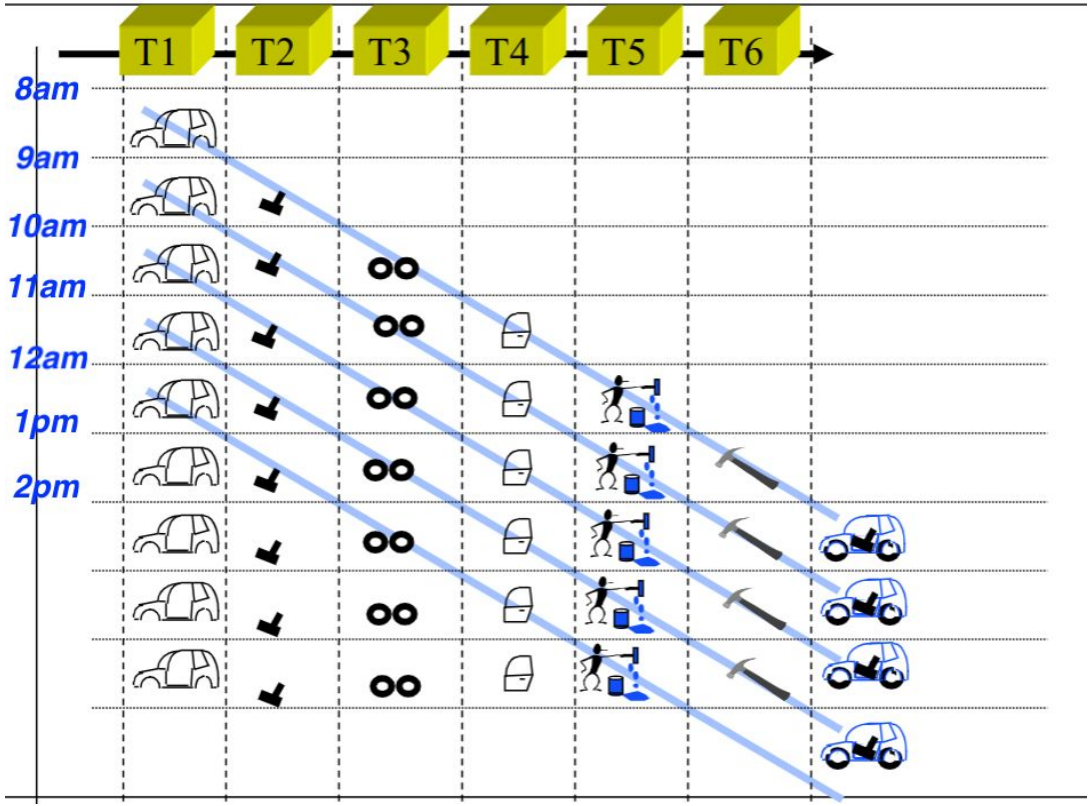


How to increase the production ?

How to increase the production ?

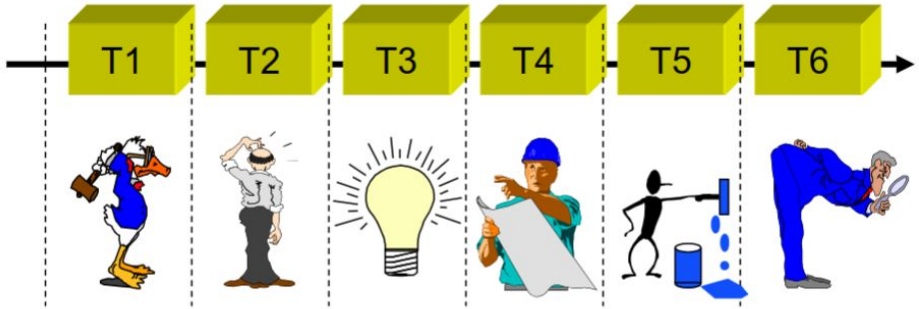
- Placing one worker for each elementary task
- Specialized worker, which is able to do one elementary thing
- Optimal work

Assembly line pipeline



Assembly line workers

- **Cost for this production technique**
  - 6 workers



- **Time per car producing**
  - 6 UT (six unit time)

- **Production Cadence**
  - 1 new car for each time unit

Instruction Pipeline

Definition

- An implementation technique
  - Multiple instructions are overlapped in execution
  - Takes advantage of parallelism that exists among the actions needed to execute an instruction
- Pipe stage / Pipe segment
  - Different steps are completing different parts of different instructions in parallel
- Processor cycle
  - Time required between moving an instruction one step down the pipeline

Definition

- The pipeline designer’s goal
  - Balance the length of each pipeline stage
  - If the stages are perfectly balanced, then the time per instruction on the pipelined processor

$$\frac{\text{Time per instruction on unpipelined machine}}{\text{Number of pipe stages}}$$

- Pipelining yields a reduction in
  - The number of clock cycles per instruction (CPI)
  - It is not visible to the programmer

RISC Instruction Set: Remind ?

- What is the basics instruction types of a RISC ?

- What is the implementation of a RISC ?

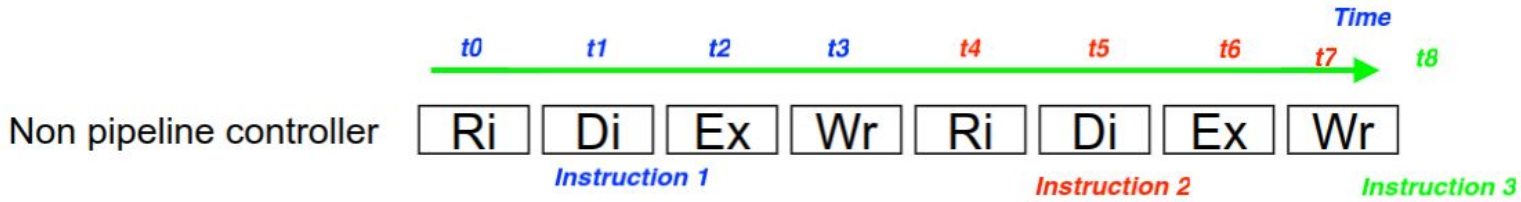
RISC Instruction Set: Remind ?

- What is the basics instruction types of a RISC ?
  - Control Instructions
  - Treatment Instructions
  - Transfer Instructions
- What is the implementation of a RISC ?
  - Instruction read cycle (RI)
  - Instruction decode/register fetch cycle (DI)
  - Execution/effective address cycle (EX)
  - Write-back cycle (WB)

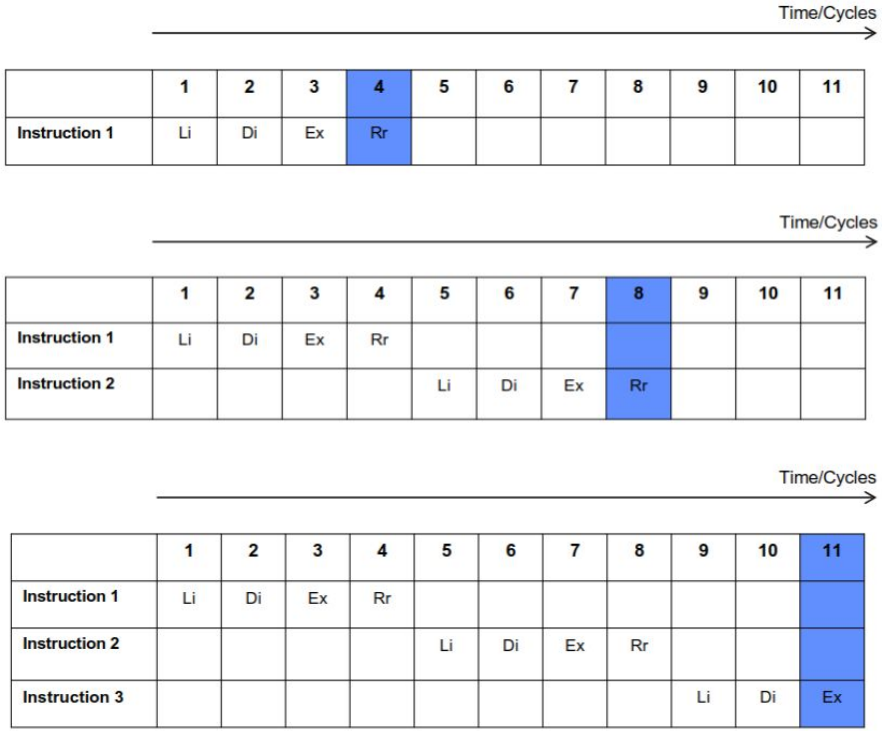
RISC is well designed to fit to the implementation stages per each cycle

Pipeline example

- Instruction read cycle (RI)
- Instruction decode/register fetch cycle (DI)
- Execution/effective address cycle (EX)
- Write-back cycle (WB)

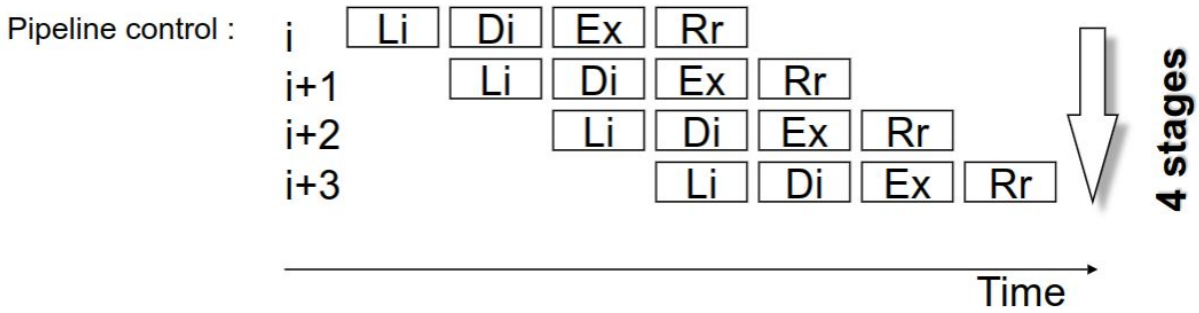


Pipeline example: non-pipeline

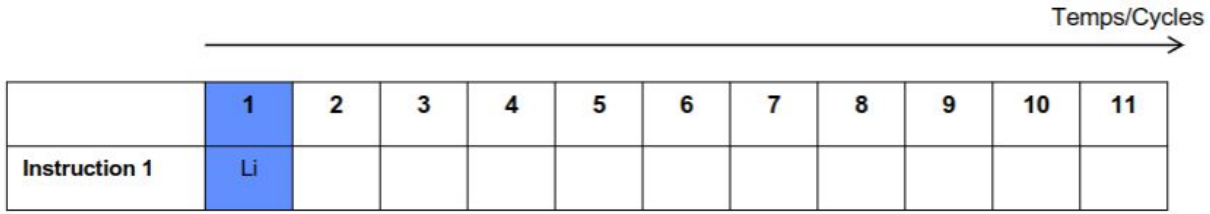


Pipeline example: with pipeline

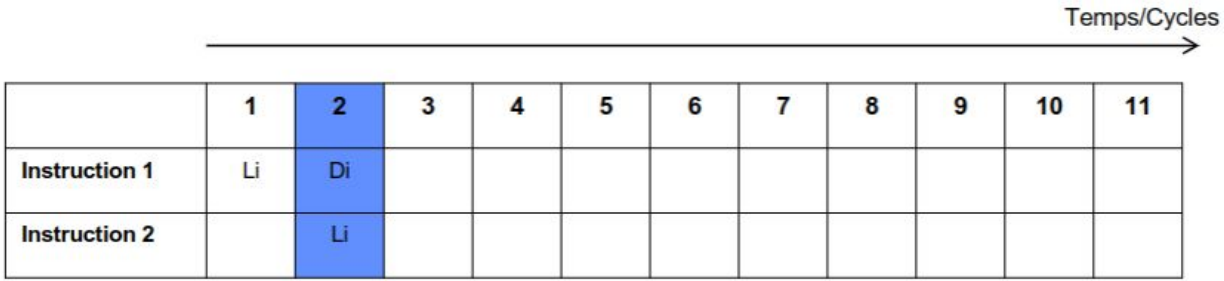
- 1 instruction per cycle
- Clock frequency increases by factor 4
- Speedup = *4



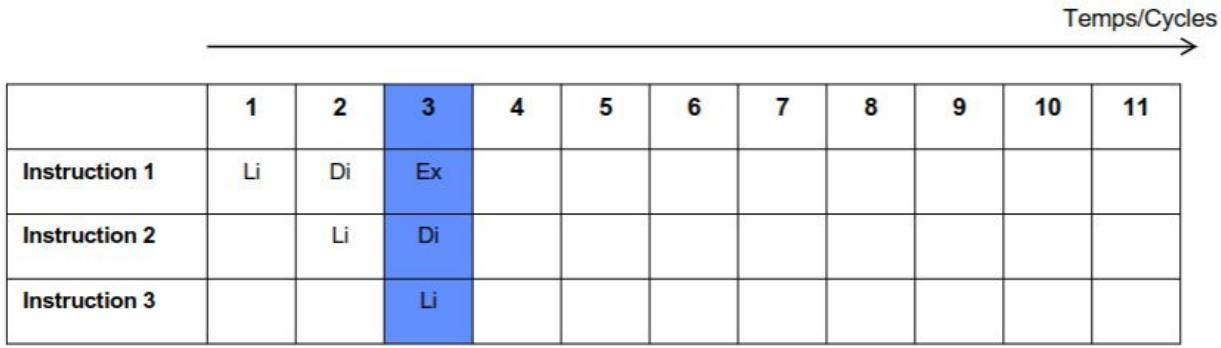
Pipeline example: with pipeline (1)



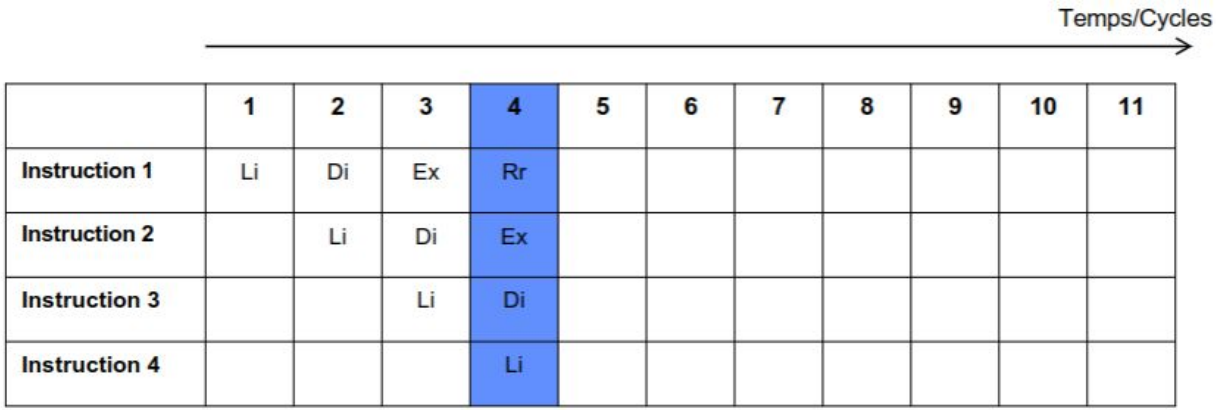
Pipeline example: with pipeline (2)



Pipeline example: with pipeline (3)

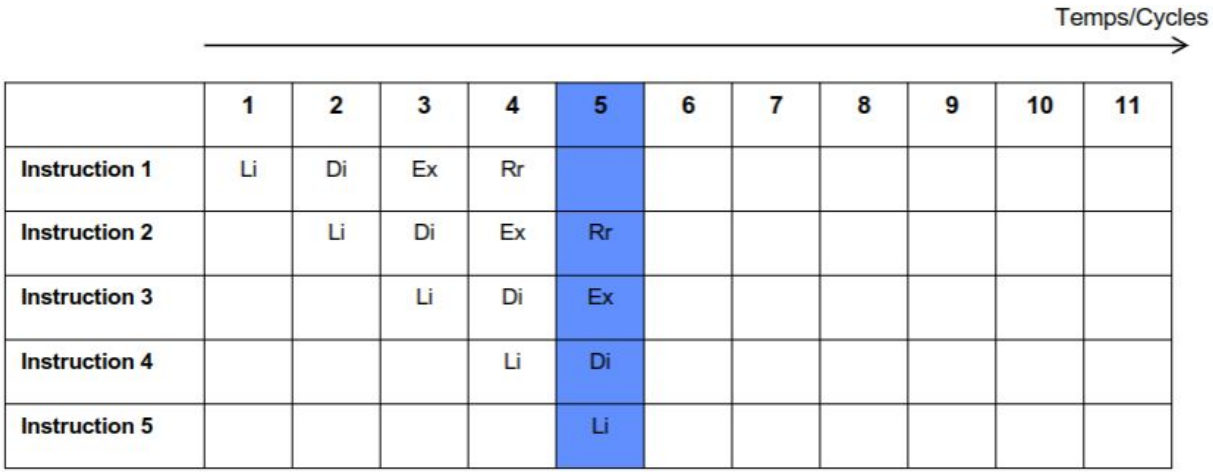


Pipeline example: with pipeline (4)



	1	2	3	4	5	6	7	8	9	10	11
Instruction 1	Li	Di	Ex	Rr							
Instruction 2		Li	Di	Ex							
Instruction 3			Li	Di							
Instruction 4				Li							

Pipeline example: with pipeline (5)



	1	2	3	4	5	6	7	8	9	10	11
Instruction 1	Li	Di	Ex	Rr							
Instruction 2		Li	Di	Ex	Rr						
Instruction 3			Li	Di	Ex						
Instruction 4				Li	Di						
Instruction 5					Li						

Pipeline example: with pipeline (6)

Temps/Cycles →

	1	2	3	4	5	6	7	8	9	10	11
Instruction 1	Li	Di	Ex	Rr							
Instruction 2		Li	Di	Ex	Rr						
Instruction 3			Li	Di	Ex	Rr					
Instruction 4				Li	Di	Ex					
Instruction 5					Li	Di					
Instruction 6						Li					

Pipeline example: with pipeline (7)

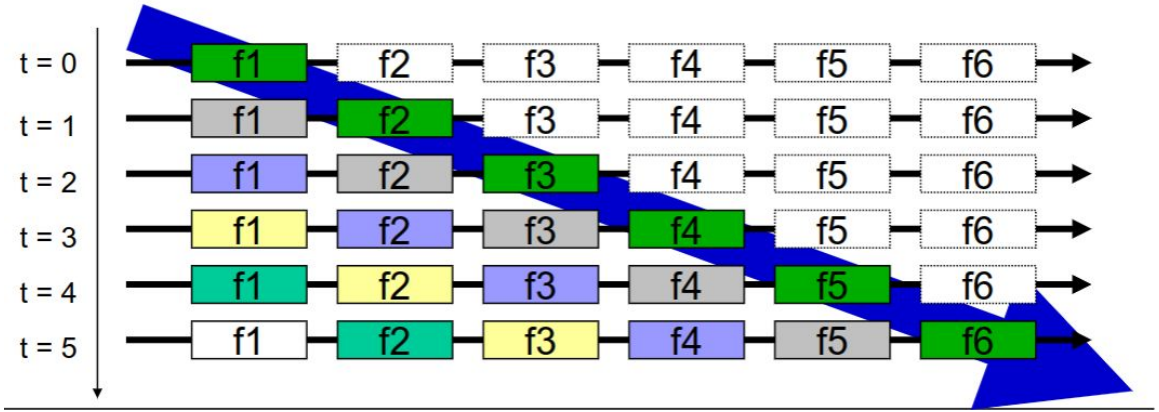
Temps/Cycles →

	1	2	3	4	5	6	7	8	9	10	11
Instruction 1	Li	Di	Ex	Rr							
Instruction 2		Li	Di	Ex	Rr						
Instruction 3			Li	Di	Ex	Rr					
Instruction 4				Li	Di	Ex	Rr				
Instruction 5					Li	Di	Ex				
Instruction 6						Li	Di				
Instruction 7							Li				

Data Computation Time

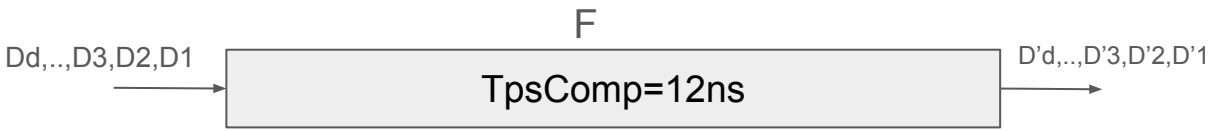
Data Computation Time

- Let F the computation to realized
- Let F = fn o fn-1 o ..... o f2 o f1 the decomposition of F
- Let tn, tn-1, ..., t2 and t1 the times for the computation of each elementary part
- The time to compute one data is $Tps = \sum ti$
- The cadence of computation is : Cadence = max (ti)



Generalization

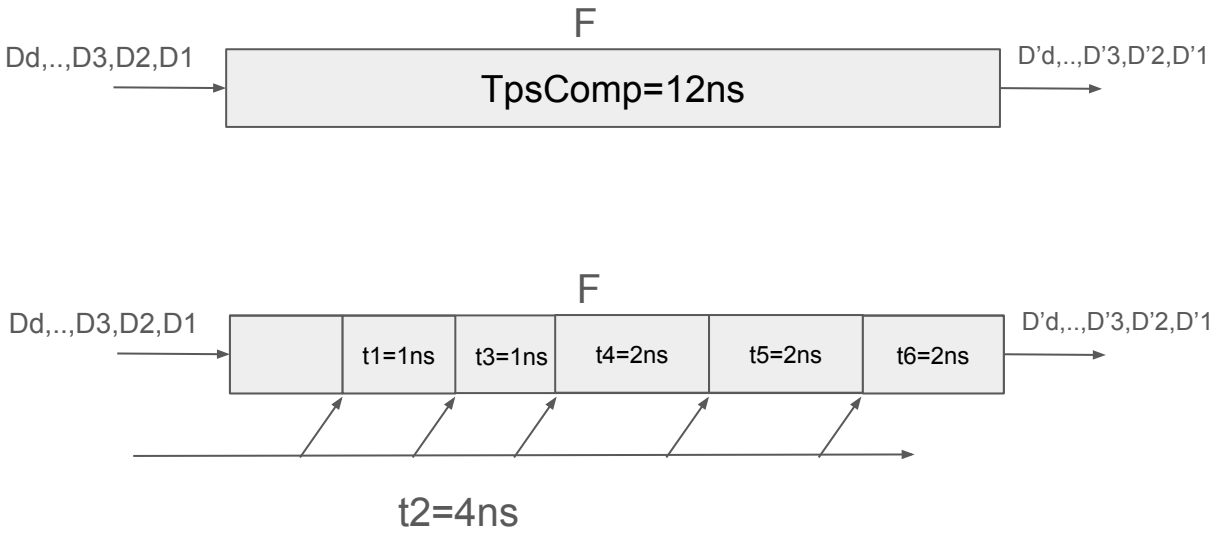
- Let TpsComp the computation time for the global treatment
- Let N the number of pipeline stages
- Let Tcycle the cycle time (cadence of computing)
    - Tcycle = TpsComp / N + Treg



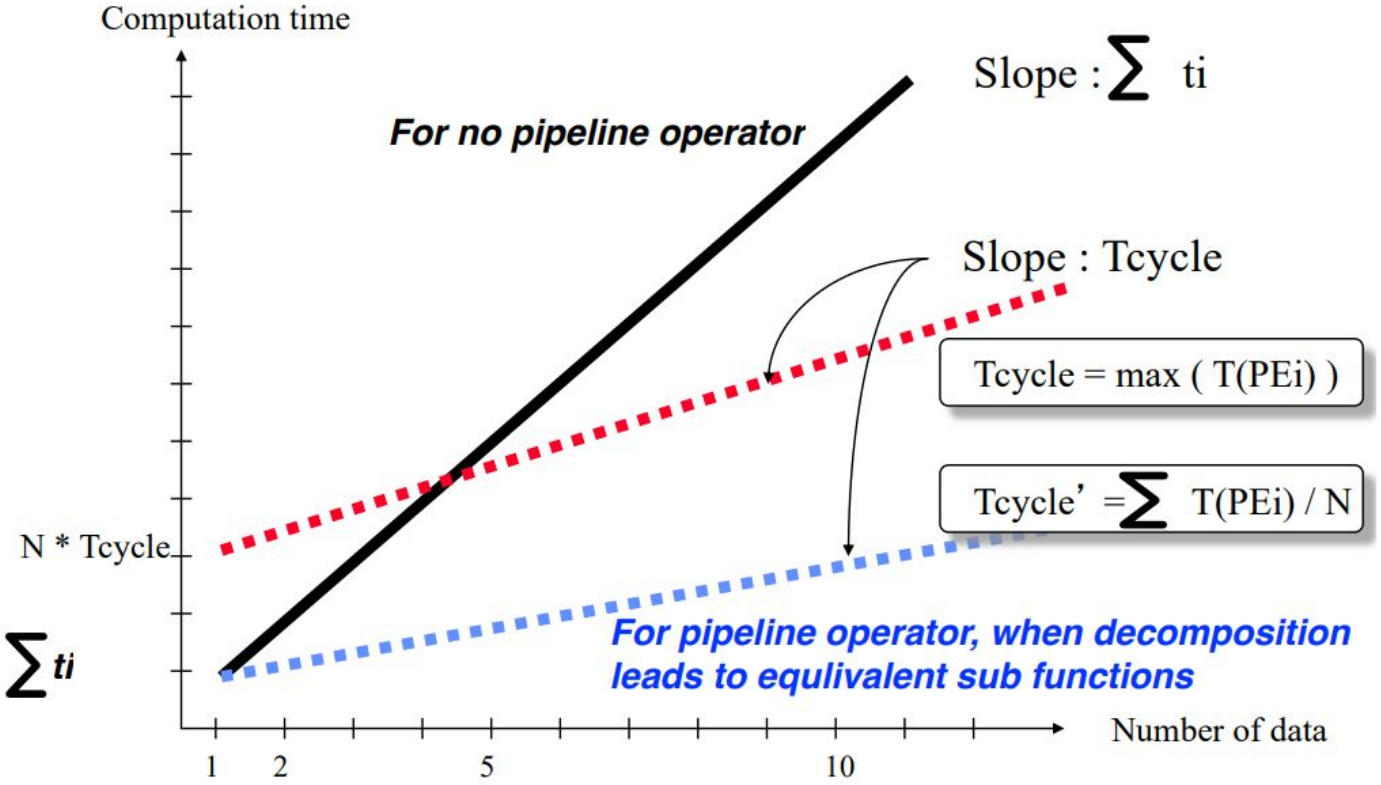
Generalization (cont.)

- Let D the number of data to transform
- Let T(D) the time to transform D data
  - T(D) = (D + N - 1) * Tcycle
- The time to transform one data if given by
  - T(D) / D = (D + N - 1) * (TpsComp / N + Treg) / D
- If D go to infinity and D >> N then the transformation time per data is
  - T(D) / D = TpsComp / N + Treg
- If N go to infinity and D << N then the transformation time per data is
  - T(D) / D = N / D * Treg

Generalization Example



Computation time performance



Thank you for you listening