Computer Systems II

Li Lu
Room 605, CaoGuangbiao Building
li.lu@zju.edu.cn
https://person.zju.edu.cn/lynnluli

Pipelining

Pipelining?

You already knew!

Pipelining

- What is pipelining?
- How is the pipelining implemented?
- What makes pipelining hard to implement?

Cafeteria:

- Did you wait until all others finish?

Cafeteria: Order

Cafeteria: Pay

Cafeteria: Enjoy

- While some others are ordering or paying

Cafeteria: Observations?

- Besides eating…

Cafeteria: Observations?

- co-use **dependent function areas** speed up the dining process of all

Cafeteria: Observations?

- Individual perspective?

…… a potentially very, very long queue

fastest if only one to serve

order | enjoy | pay

Cafeteria: Observations?

- Average - faster
- Individual – slower *(service time)*

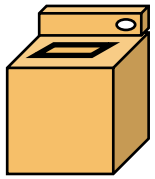
  but much less time in queue

- Individual – faster: queue + service

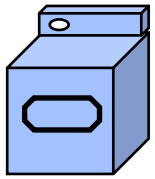
(classic) laundry example

Laundry Example

- **A**nn, **B**rian, **C**athy, **D**ave
- Each has one load of clothes to
- wash, dry, fold.



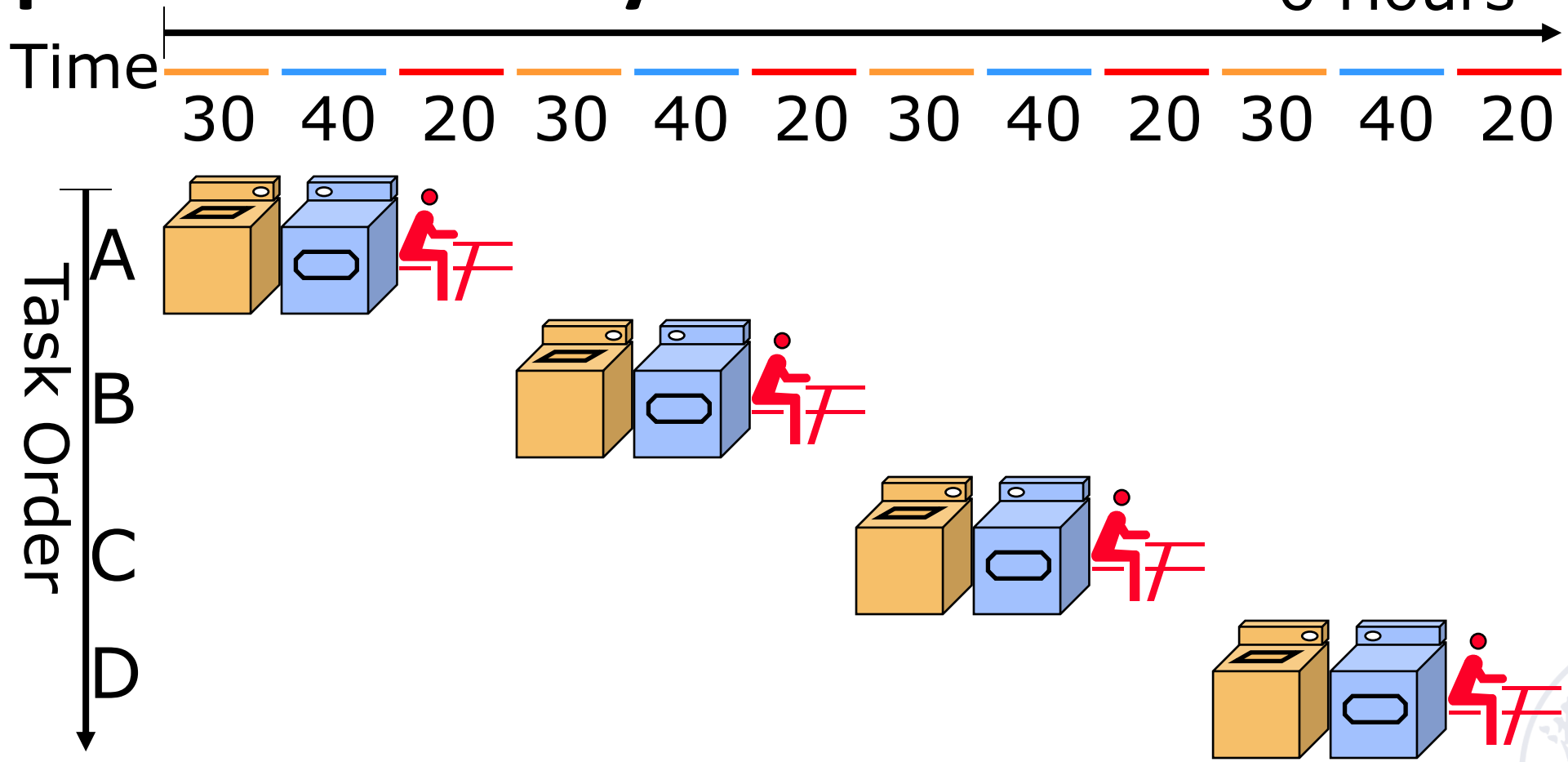
washer
30 mins



dryer
40 mins

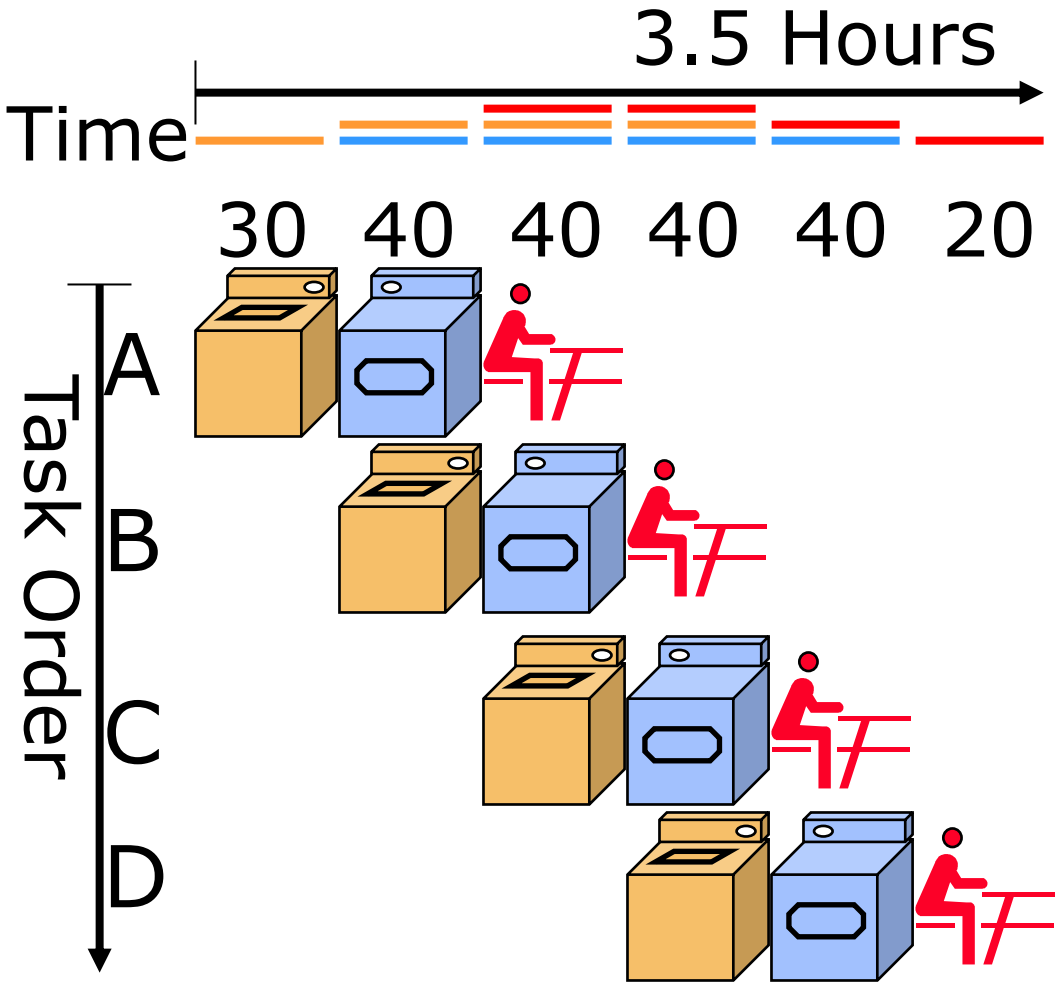
folder
20 mins

Sequential Laundry

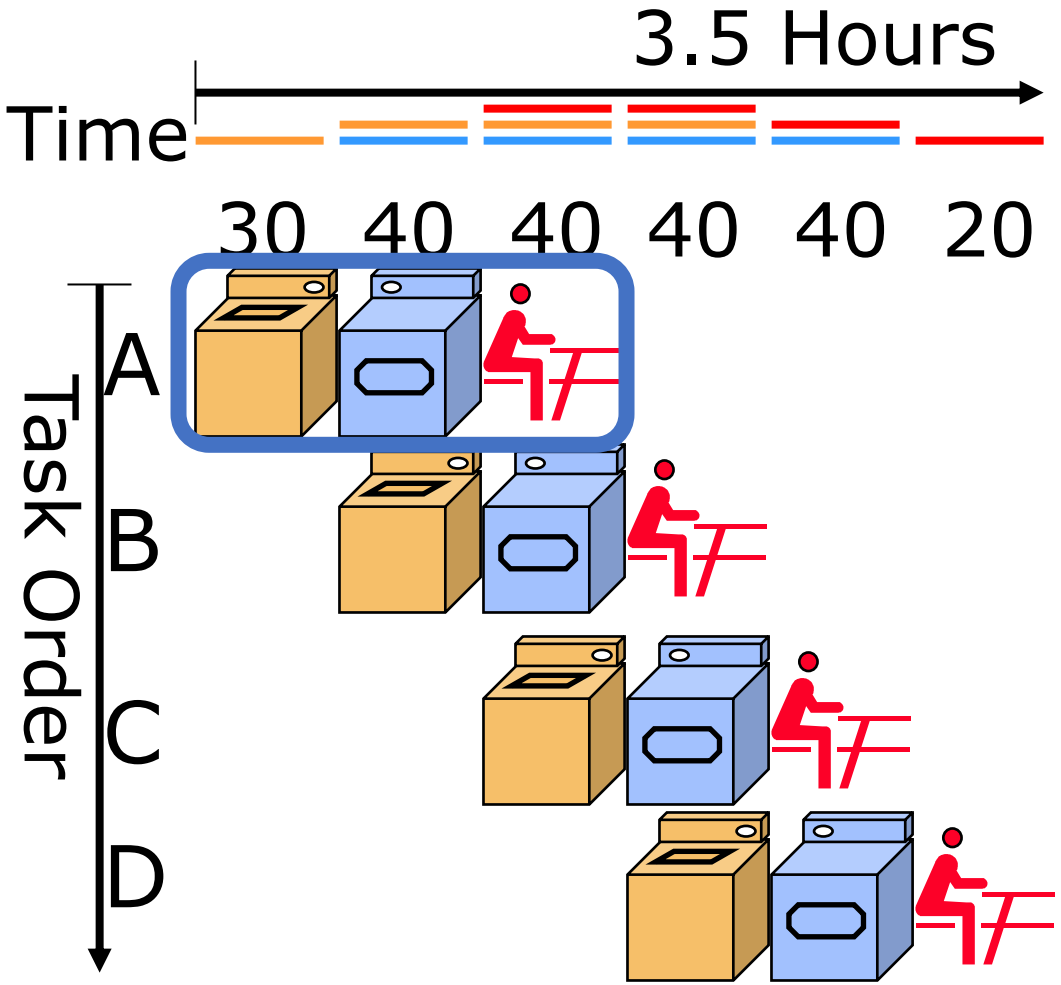


What would you do?

Pipelining Laundry



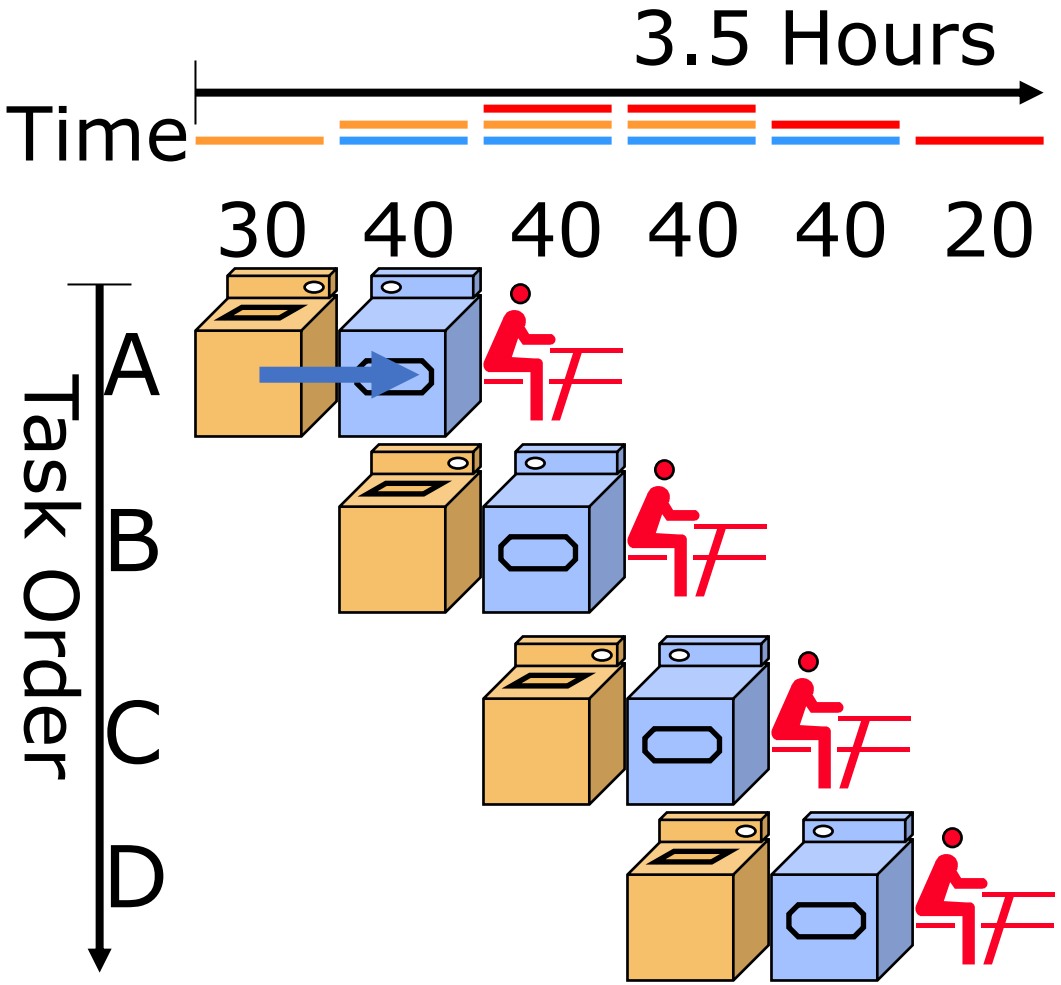
Pipelining Laundry



Observations

- A task has a series of stages;

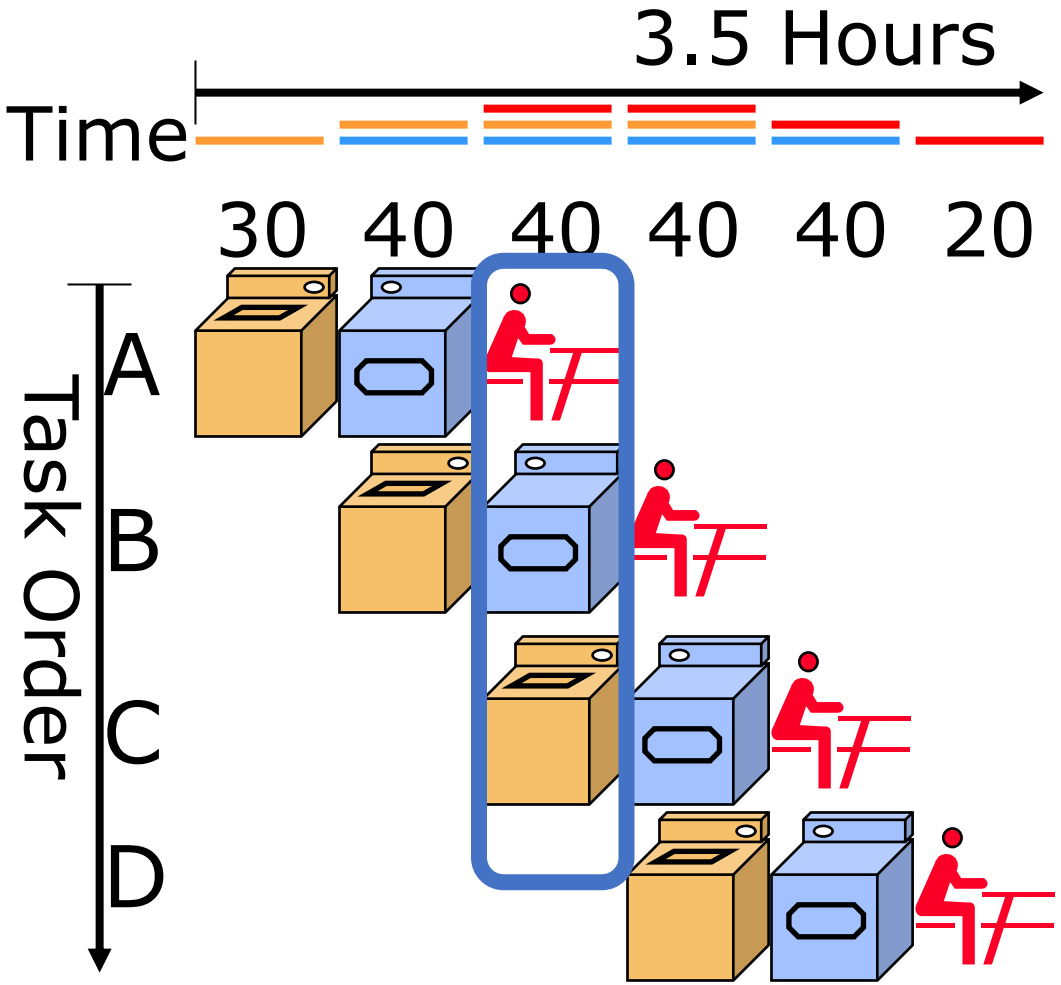
Pipelining Laundry



Observations

- A task has a series of stages;
- Stage dependency:
  e.g., wash before dry

Pipelining Laundry

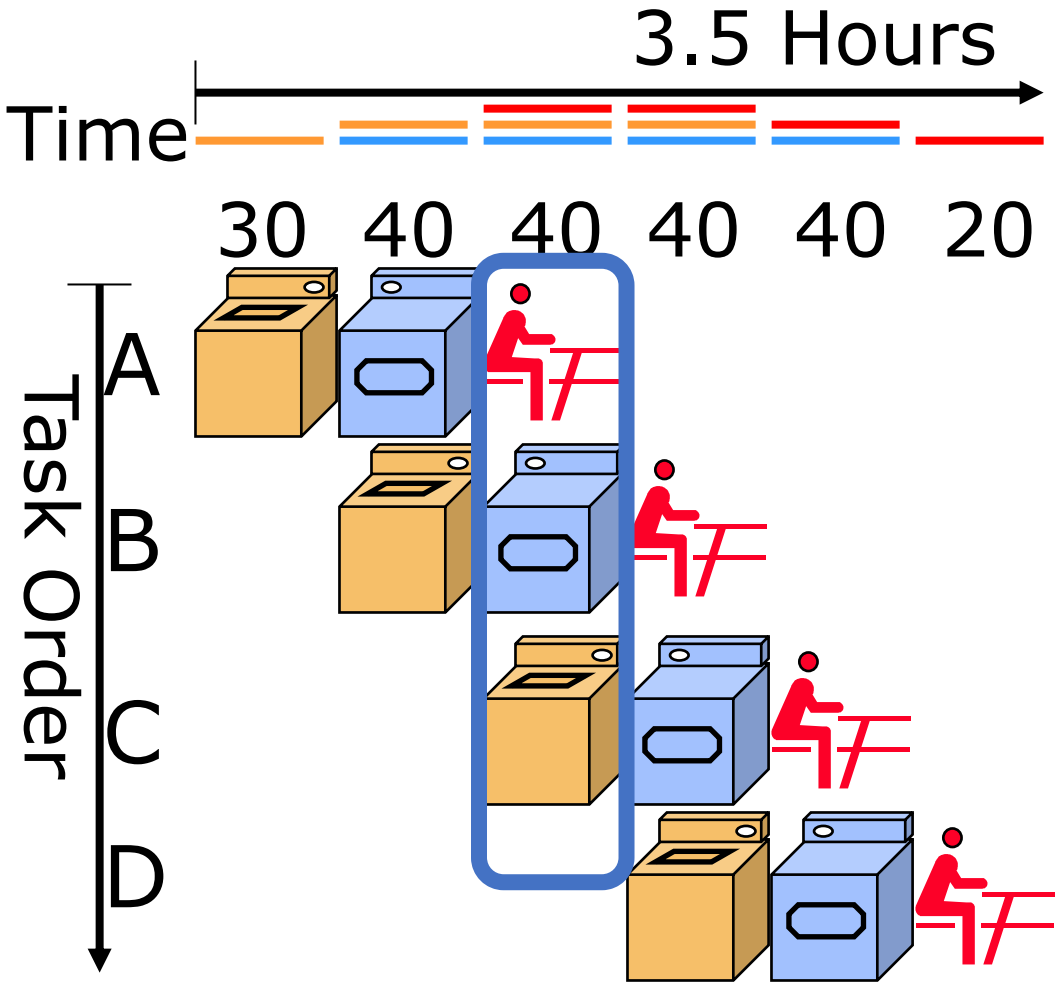


Observations

- A task has a series of stages;
- Stage dependency:

  e.g., wash before dry;
- Multi tasks with overlapping stages;

Pipelining Laundry

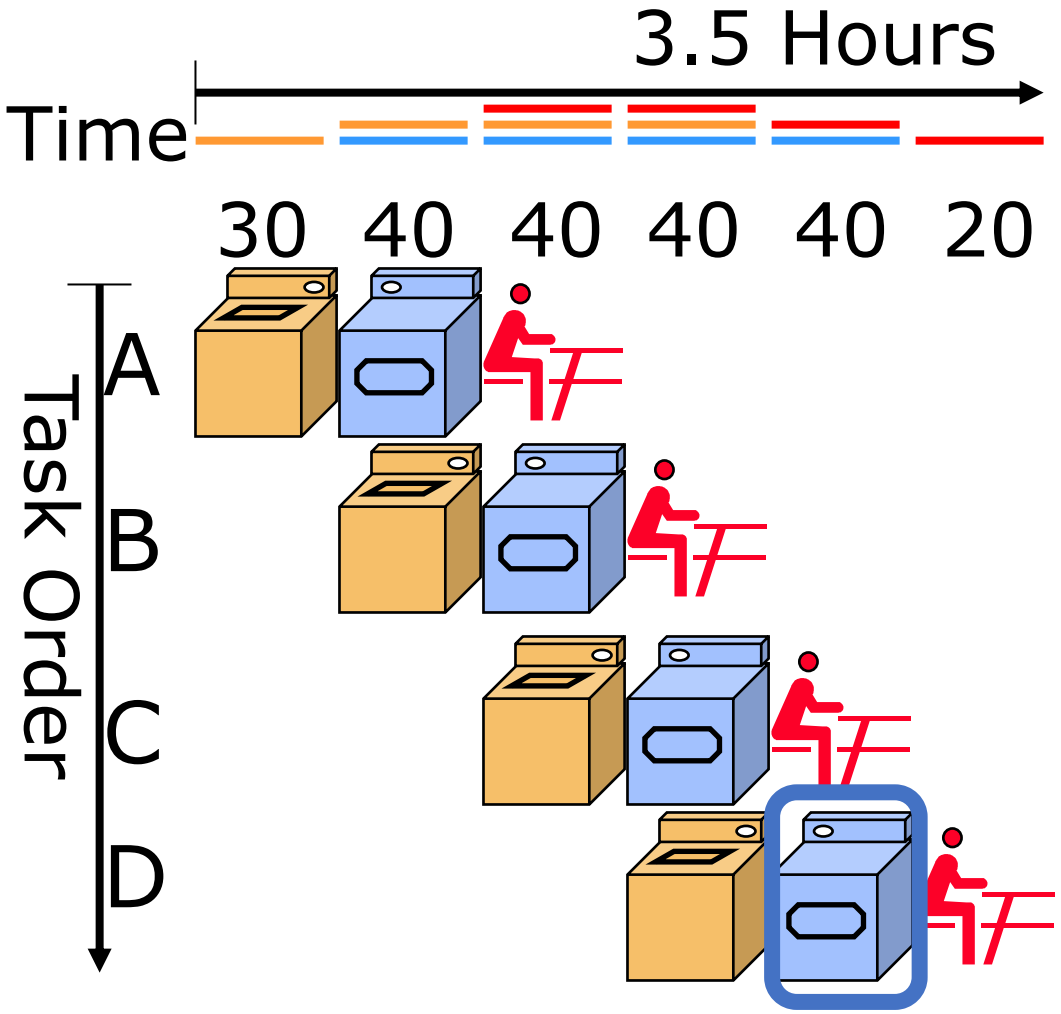


Observations

- A task has a series of stages;
- Stage dependency:

  e.g., wash before dry;
- Multi tasks with overlapping stages;
- Simultaneously use diff resources to speed up;

Pipelining Laundry

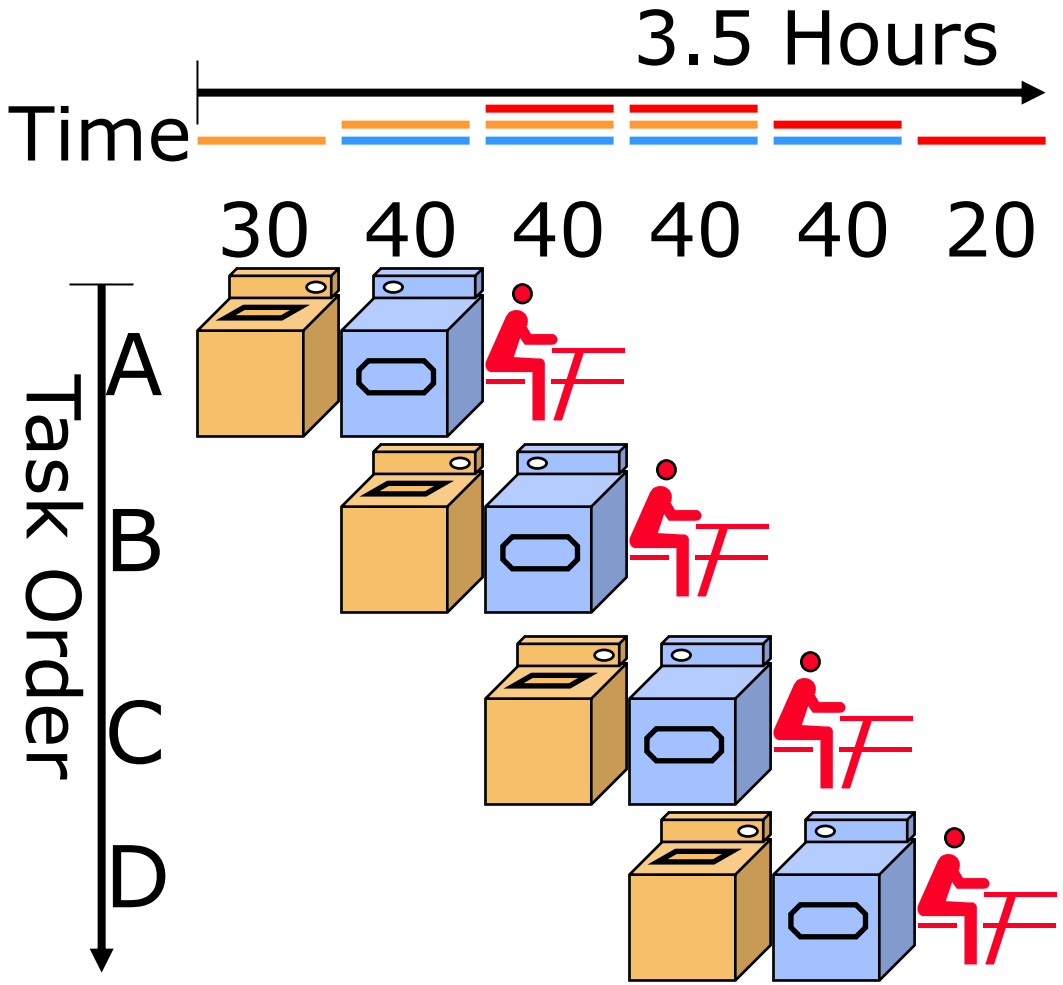


Observations

- A task has a series of stages;
- Stage dependency:

  e.g., wash before dry;
- Multi tasks with overlapping stages;
- Simultaneously use diff resources to speed up;
- Slowest stage determines the finish time;

Pipelining Laundry

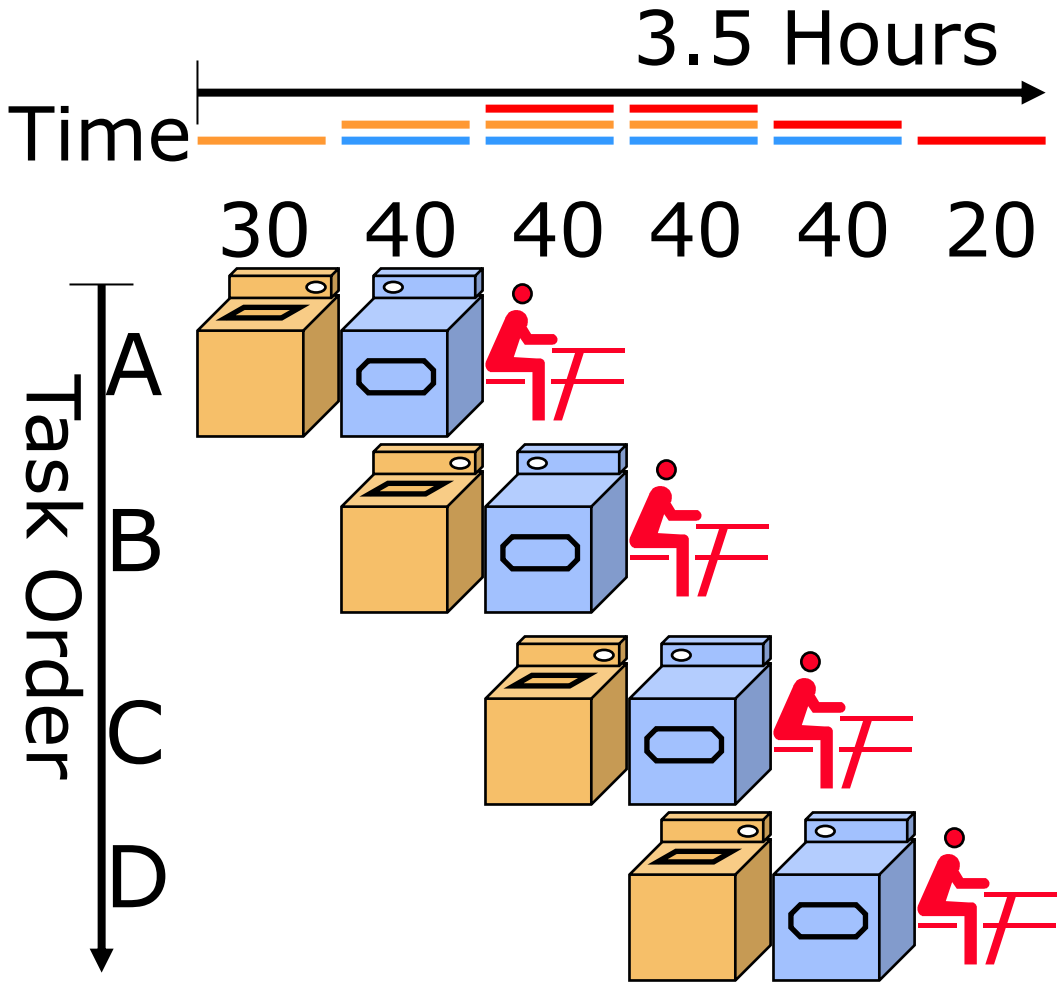


Observations

- No speed up for individual task;

  e.g., A still takes 30+40+20=90

Pipelining Laundry



Observations

- No speed up for individual task;

  e.g., A still takes 30+40+20=90

- But speed up for average task execution time;

  e.g., 3.5*60/4=52.5 < 30+40+20=90

Pipelining Elsewhere: Assembly Line

Auto

Cola

What exactly is pipelining in computer arch?

Speed up the execution of instructions (machine language)

- Shorten the execution time of each instruction
  - More high-speed devices
  - Better calculation methods
  - Improve the parallelism of each microoperation in the instruction
  - reduce the number of beats needed in the interpretation process

- Reduces the execution time of the entire program (machine language)
  - Through the control mechanism, the interpretation of the entire program can be speeded up by means of simultaneous interpretation of two, multiple or even whole programs.

- *Pipelining is an implementation technique whereby multiple instructions are overlapped in execution; it takes advantage of parallelism that exists among the actions needed to execute an instruction.*

- *Today, pipelining is the key implementation technique used to make fast CPUs.*

Pipelining

- *"A technique designed into some computers to increase speed by starting the execution of one instruction before completing the previous one."*

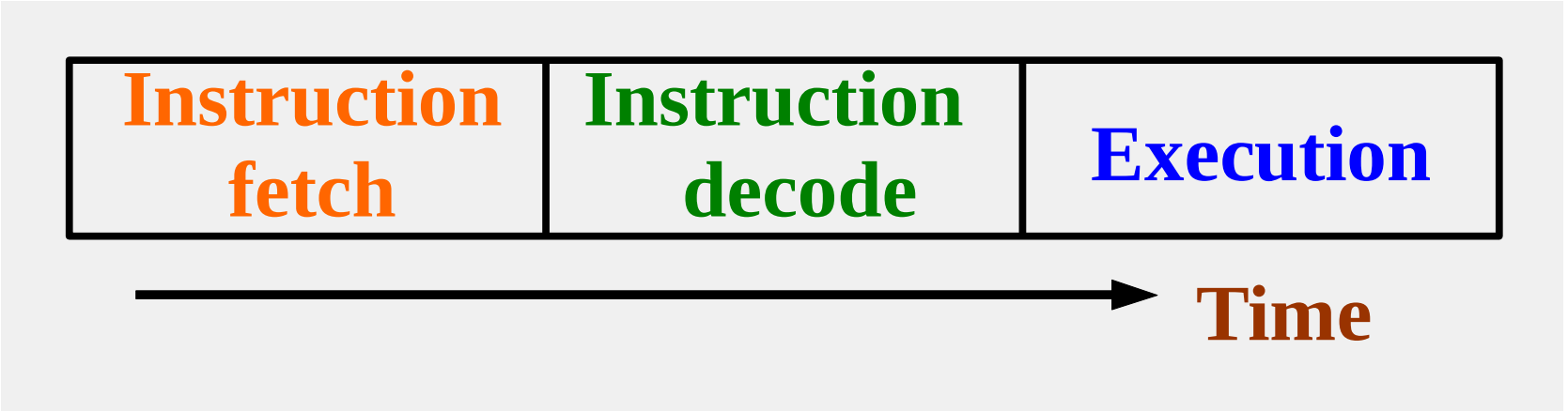
  ----Modern English-Chinese Dictionary

- implementation technique whereby different instructions are overlapped in execution at the same time.
- implementation technique to make fast CPUs

Three modes of execution

- Sequential execution
- Single overlapping execution
- Twice overlapping execution

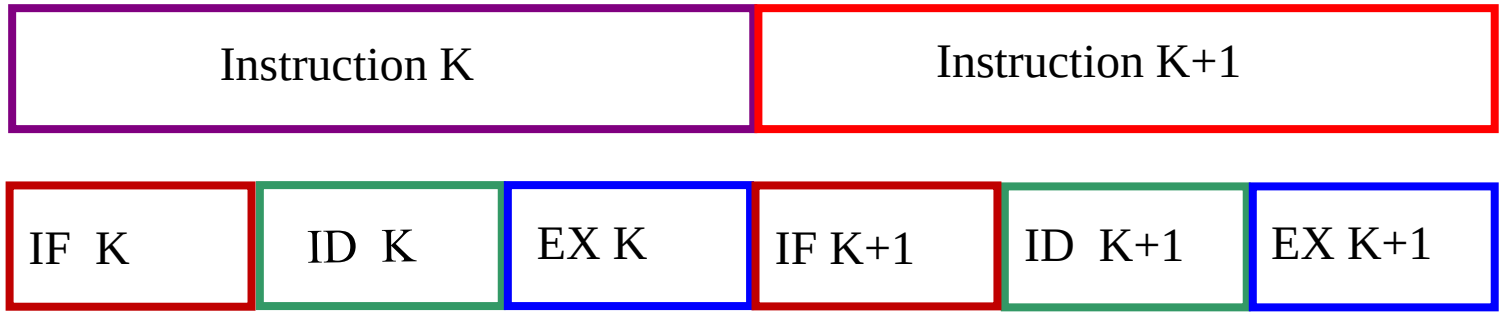
The execution of an instruction is divided into three stages：



The execution of an instruction

Sequential execution

- The execution of the instructions

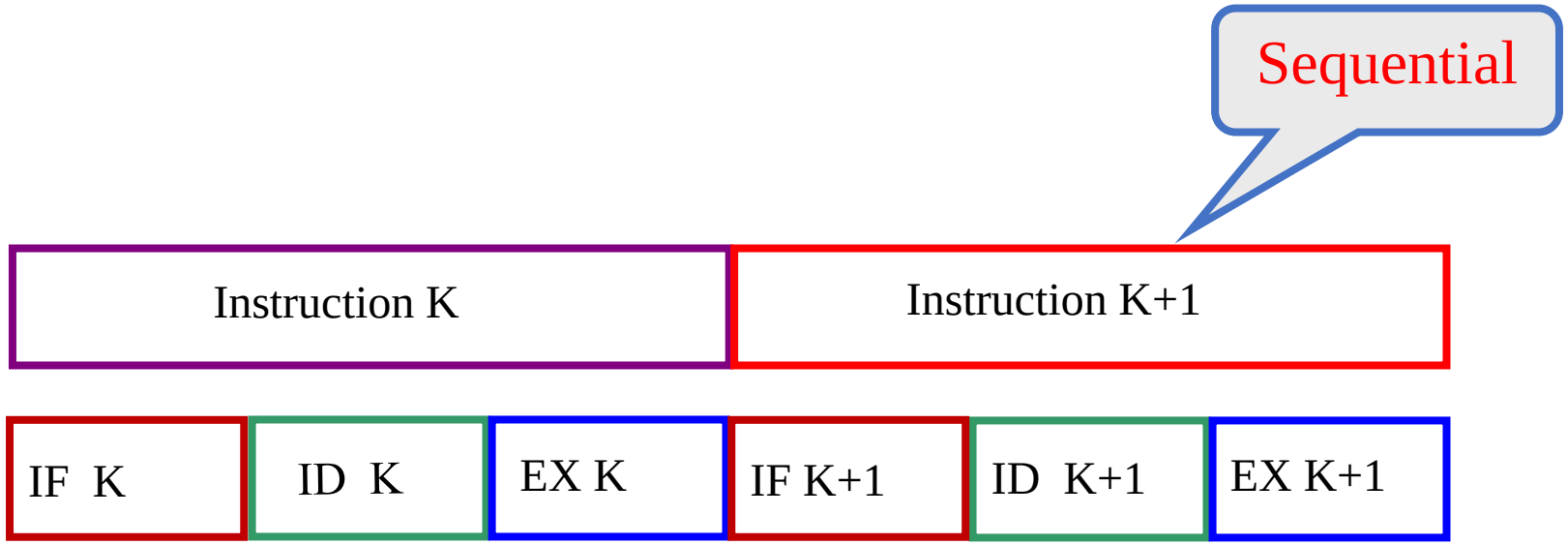


- Instructions are executed sequentially, and each microoperation in the instruction is also executed sequentially.

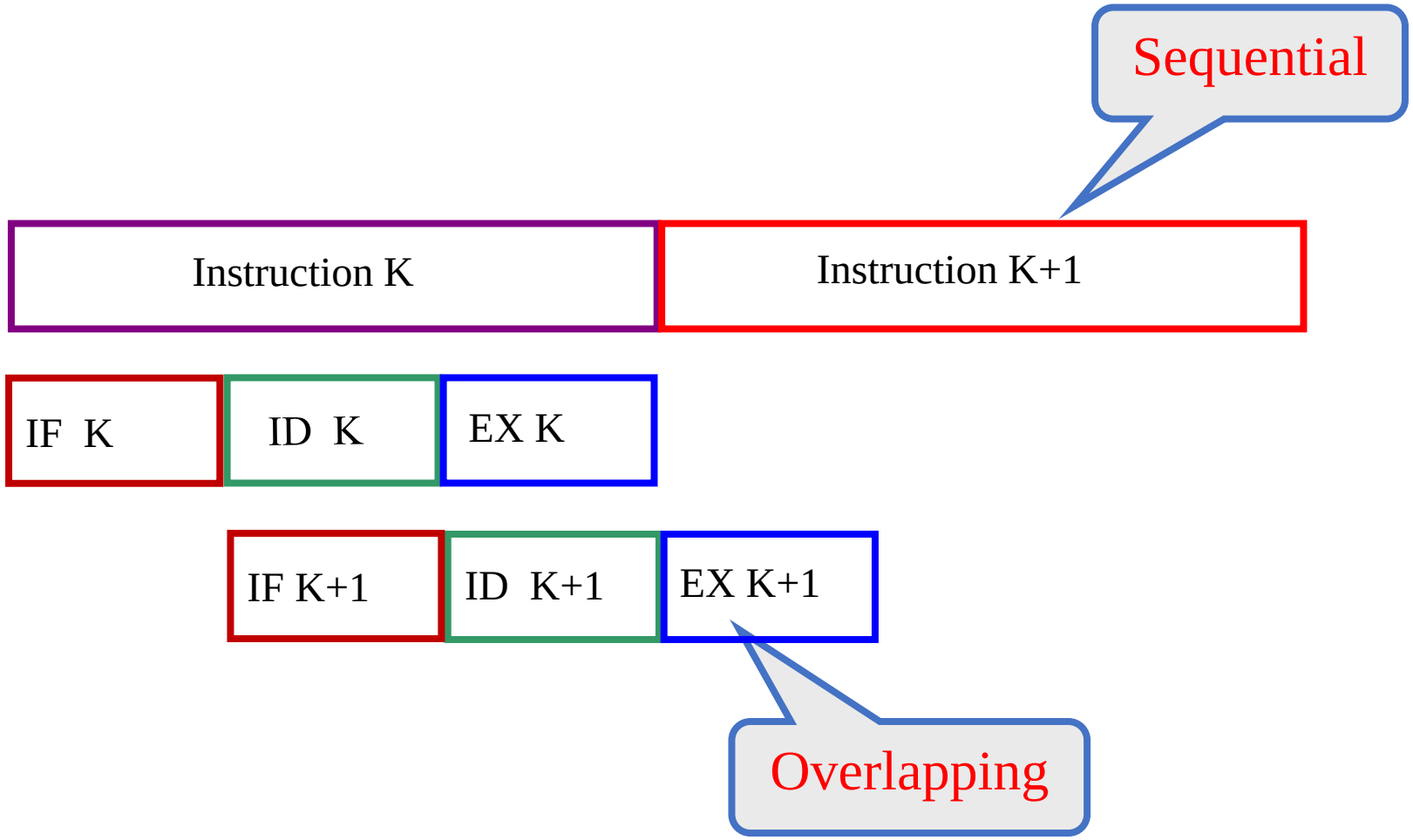
- Execution Time:

$$T = \sum_{i=1}^{n} (t_{IFi} + t_{IDi} + t_{EXi})$$

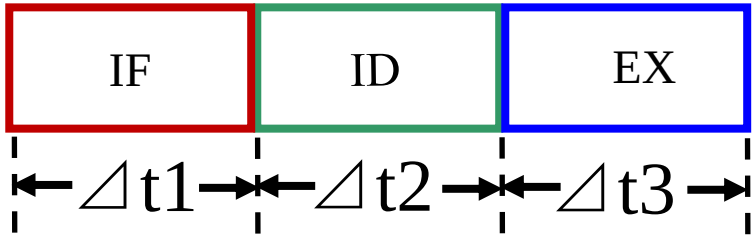
Overlapping execution



Overlapping execution



Overlapping execution

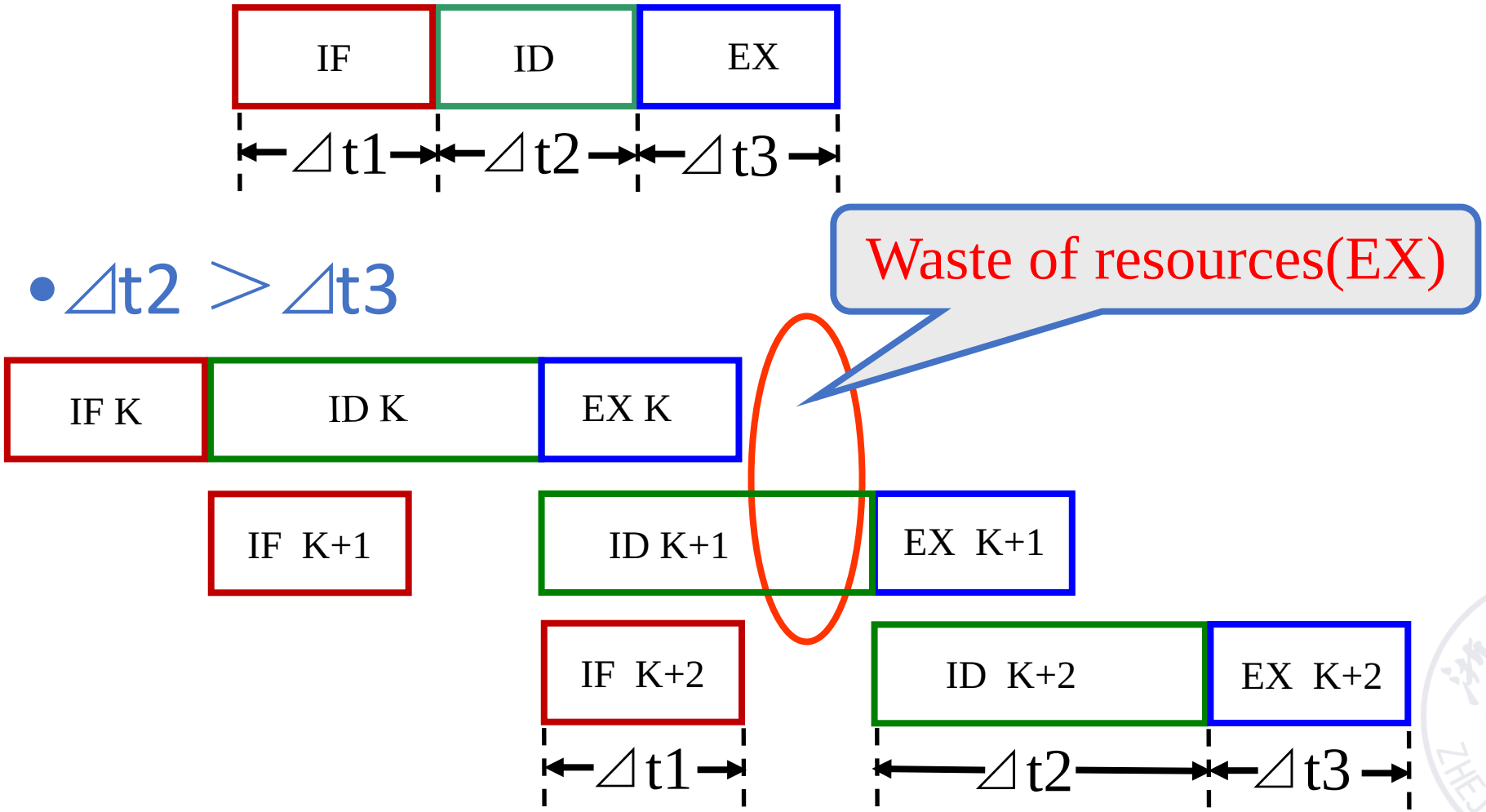


- ⊿t1 =⊿t2 =⊿t3 = ⊿t

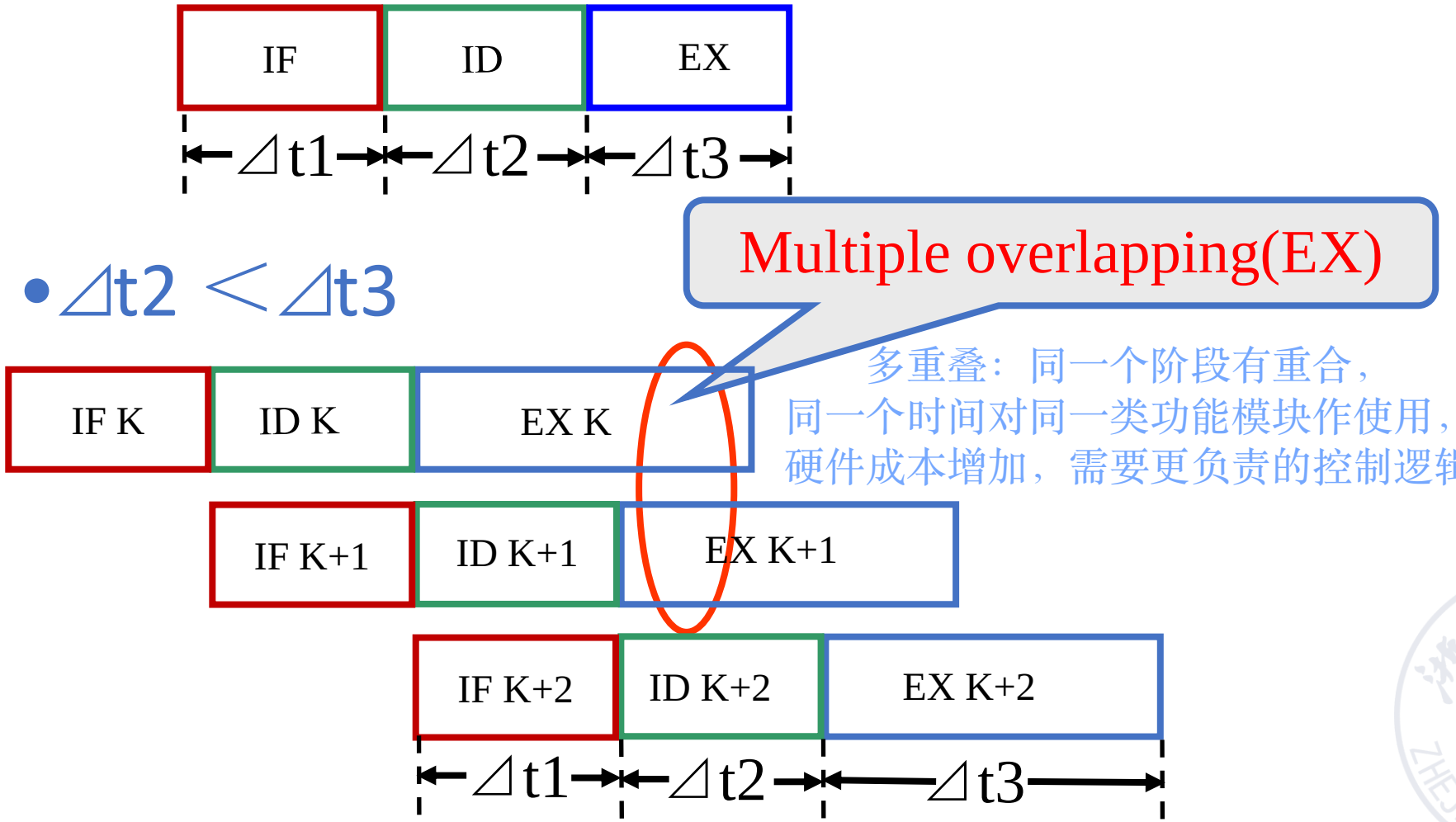
- Execute time for n instructions in sequence

$$T = \sum_{i=0}^{n} (t_{IFi} + t_{IDi} + t_{EXi}) = 3n\Delta t$$

Overlapping execution



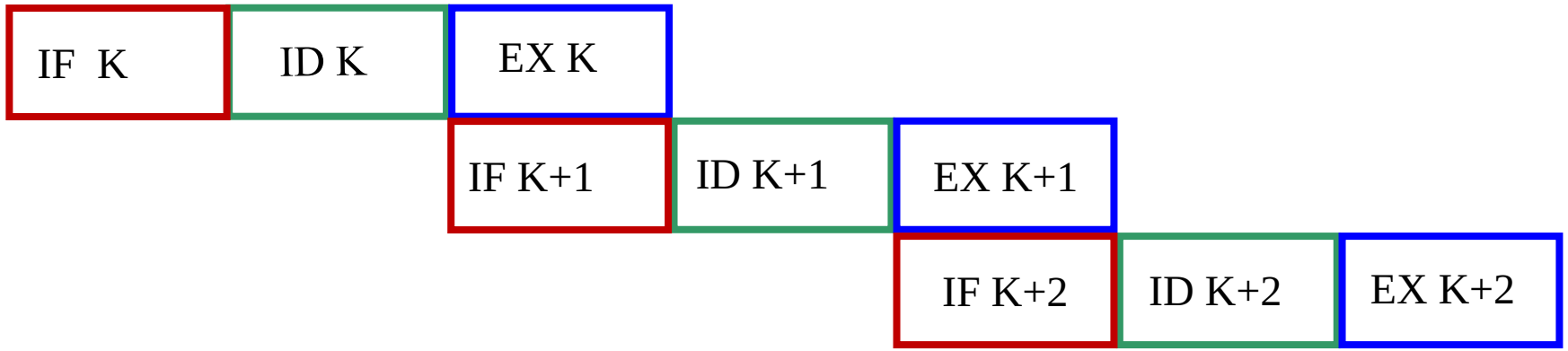
Overlapping execution



- Sequential execution
  - ✓ Simple control, saving equipment
- Overlapping execution
  - ✓ High-usage of functional unit

Overlapping execution

- Single overlapping execution
- Execute the k instruction at the same time as fetch the k+1 instruction



- Execute time for n instructions by single overlapping execution

$$T = (2n + 1)\Delta t$$

Overlapping execution

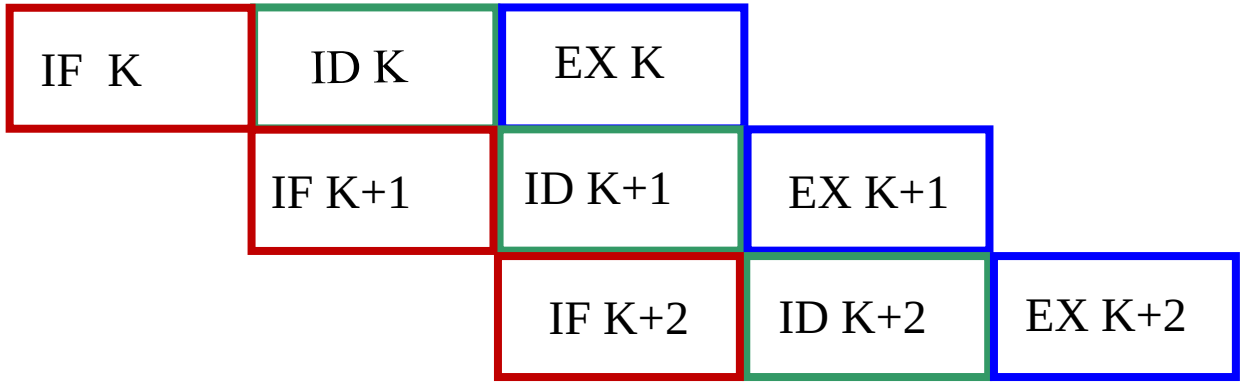
- Advantages
  - Execution time was reduced by nearly 1/3.
  - The utilization rate of functional unit is improved obviously.

- Disadvantages
  - Some additional hardware was needed, and the control process became complicated

引入额外的硬件去做同样的事
控制逻辑变得更复杂了

Overlapping execution

- Twice overlapping execution
- Decode the k instruction at the same time as fetch the k+1 instruction
- Execute the k instruction at the same time as decode the k+1 instruction



- Execute time for n instructions by twice overlapping execution

$$T = (n + 2)\Delta t$$

Twice Overlapping execution

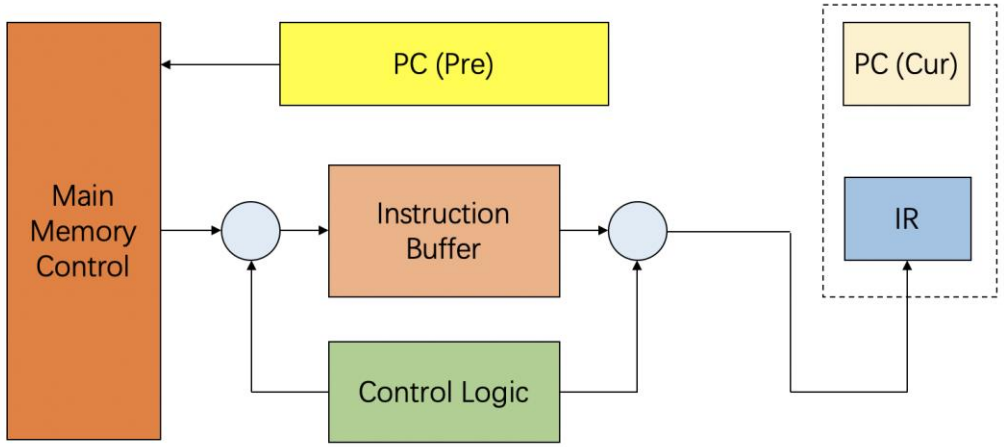
- Advantages
  - Execution time was reduced by nearly 2/3.
  - The utilization rate of functional unit is improved obviously.

- Disadvantages
  - Much more hardware was needed.
  - Separate fetch, decode, and execution components are required.

Overlapping execution

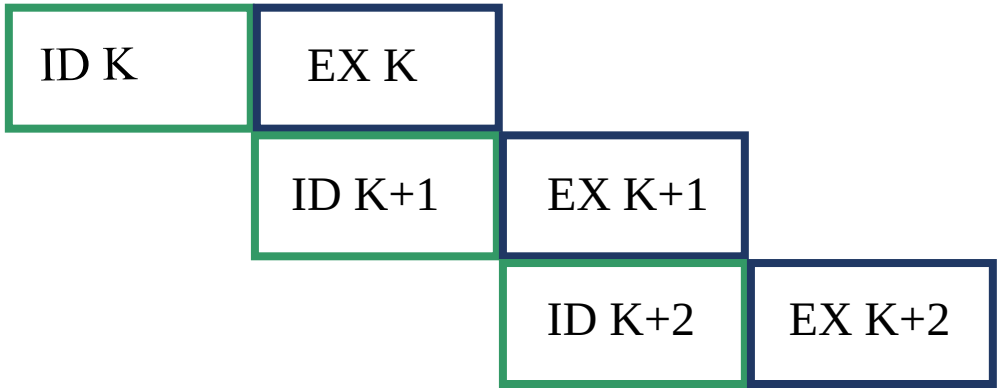
- Conflict in access memory 访存冲突
  - Instruction memory & data memory
  - Instruction cache & data cache (same memory): Harvard structure
  - Adding instruction buffer between memory and instruction decode unit

Instruction Buffer Structure



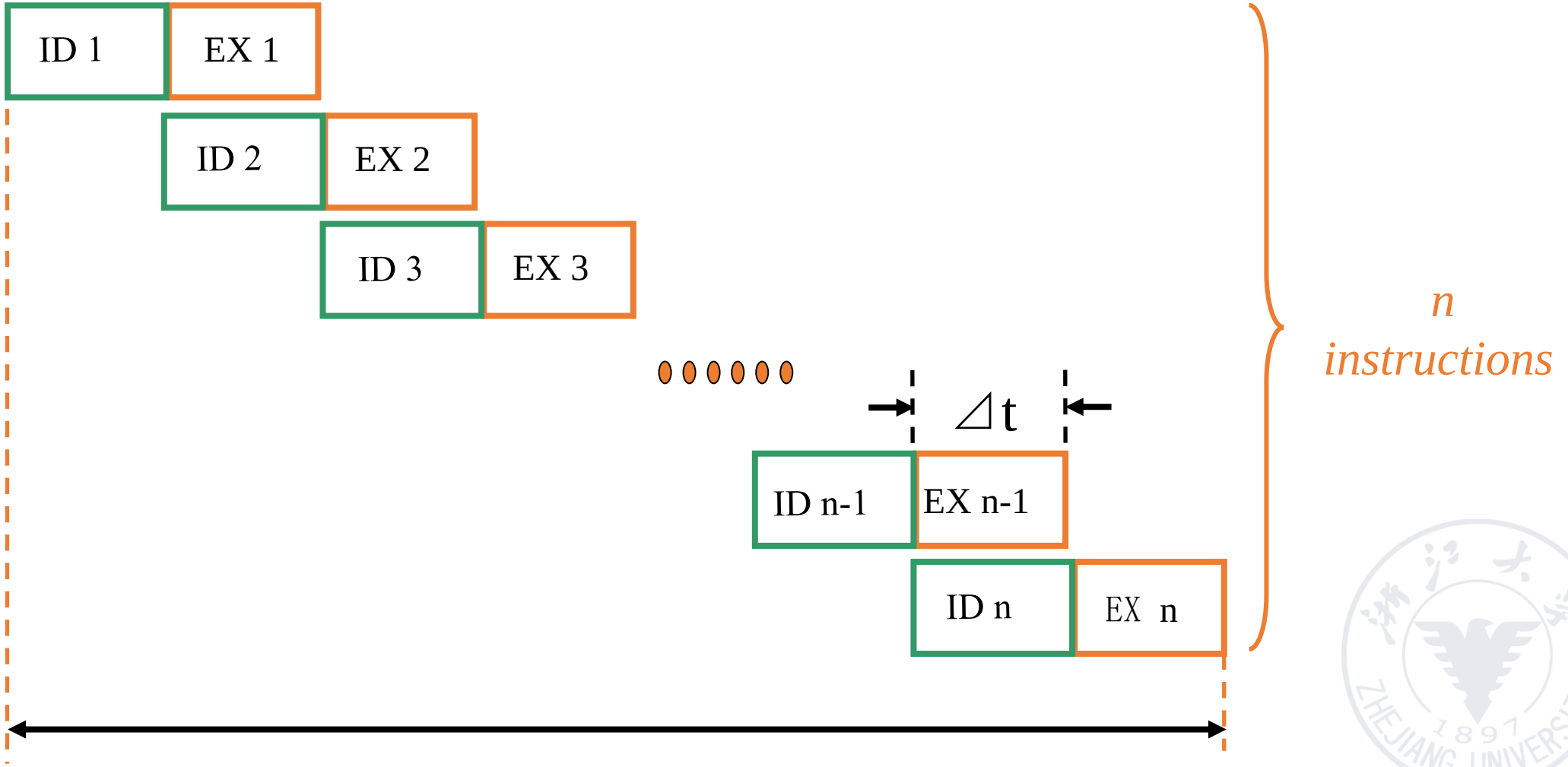
Single Overlapping execution

- If the time of fetching instruction phase is very short, fetch operation can be incorporated into the decode operation. The twice overlapping becomes single overlapping.

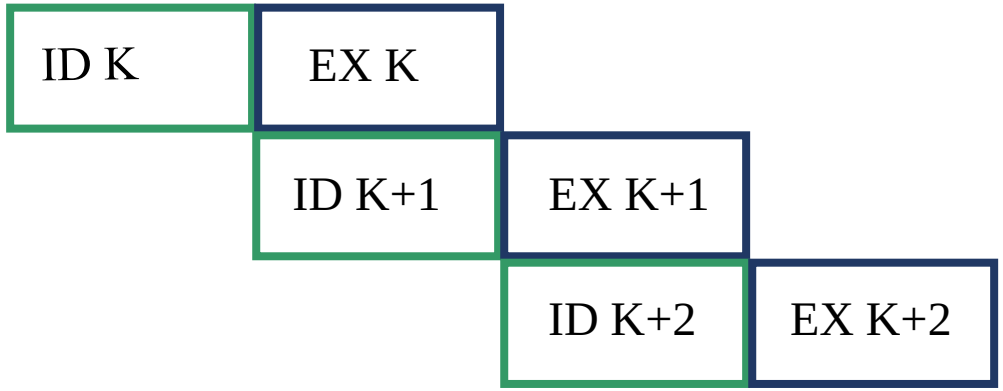


Single Overlapping execution

Single Overlapping execution

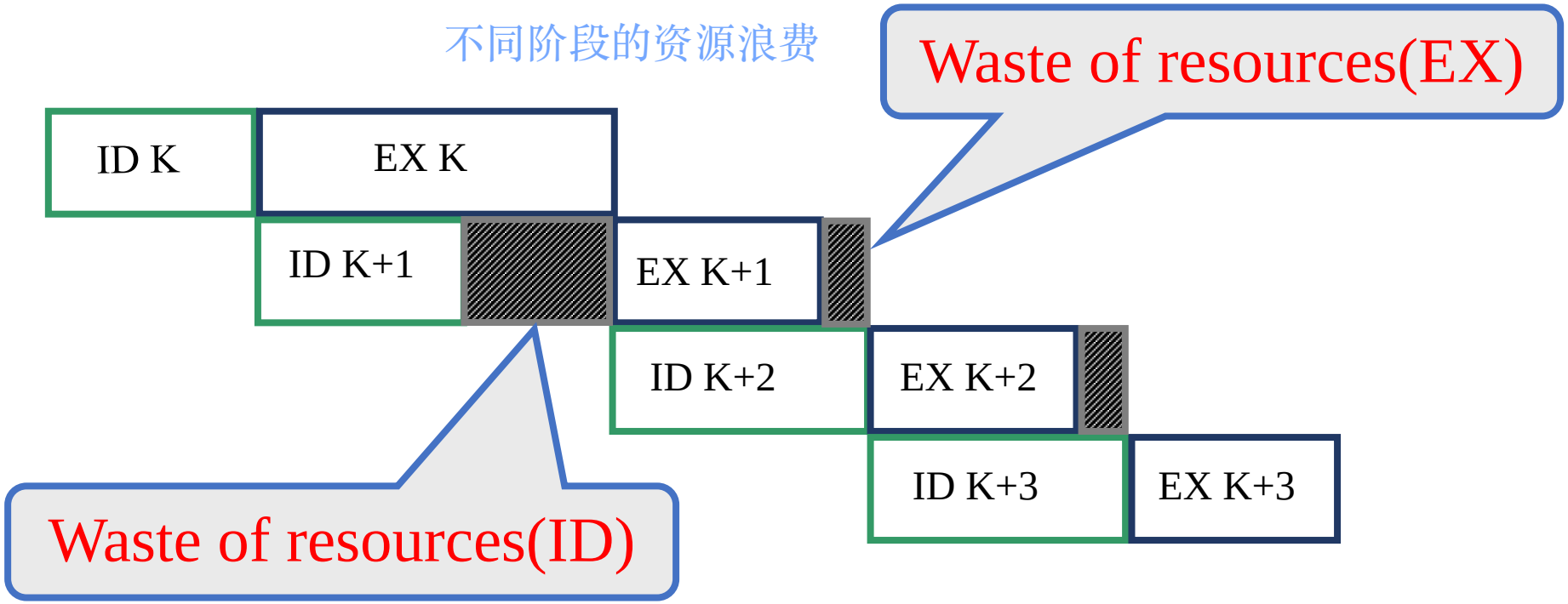


Single Overlapping execution



- $\Delta t_{ID} = \Delta t_{EX}$

Single Overlapping execution



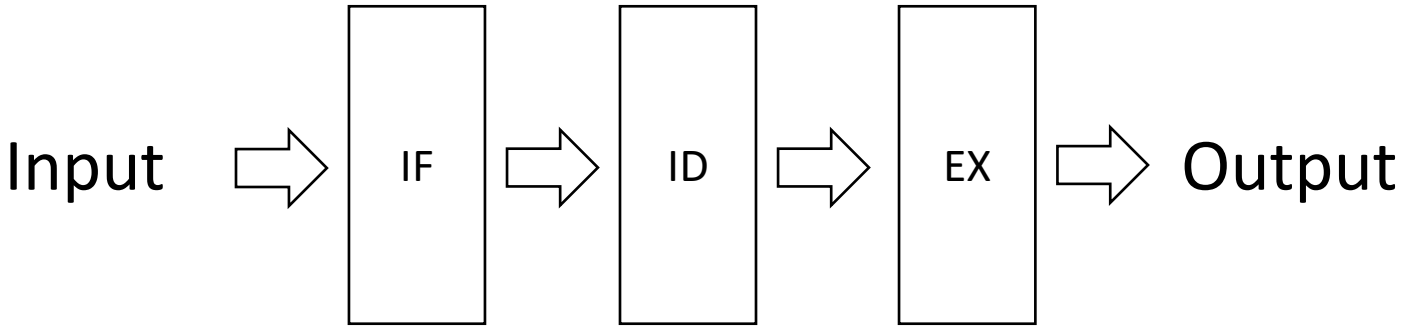
- $\Delta t_{ID} \neq \Delta t_{EX}$

What is pipelining ?

- Pipelining: The process of an instruction is divided into m (m > 2) sub processes with equal time, and the process of m adjacent instructions are staggered and overlapped in the same time.

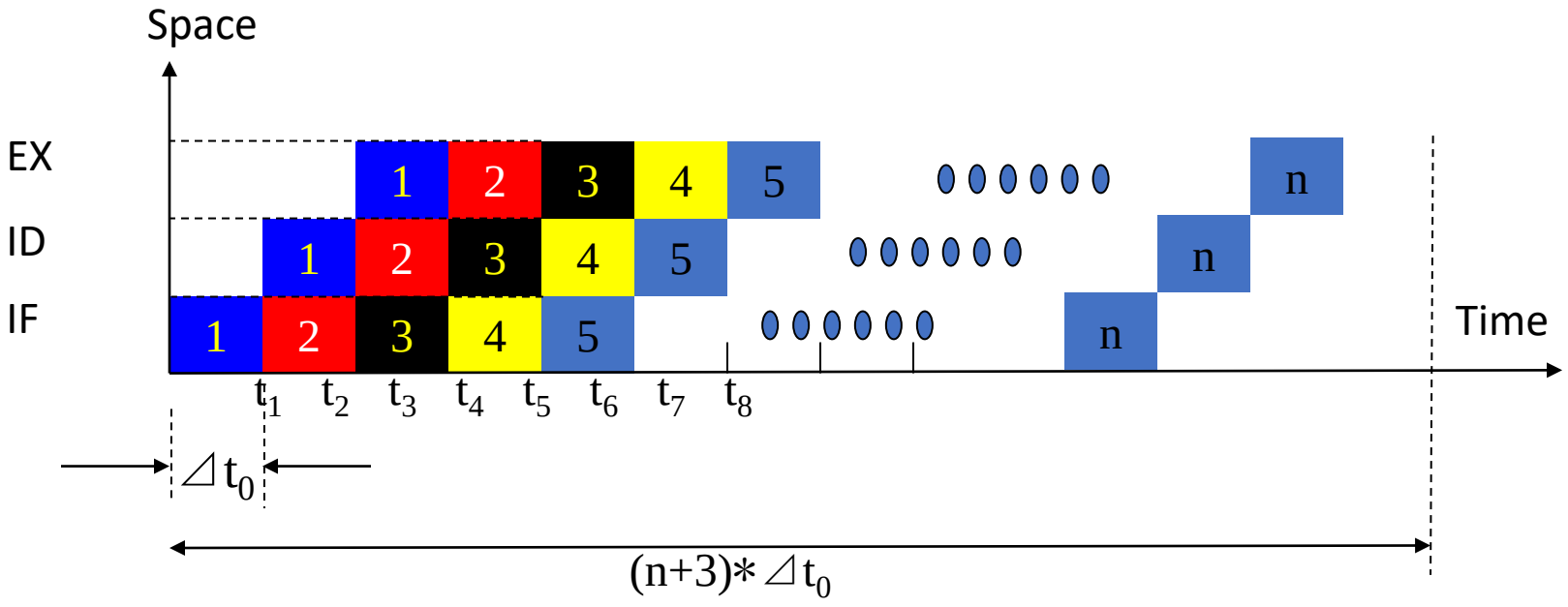
- Pipelining can be regarded as the extension of overlapping execution.
- Each subprocess and its functional components in the pipelining are called stages or segments of the pipelining, which are connected to form a pipelining.
- The number of segments in a pipelining is called the depth of pipelining.

What is pipelining ?



Pipelining

What is pipelining ?



How much faster?

Characteristics of pipelining

- The pipelining divides a process into several sub processes, each of which is implemented by a special functional unit.
- The time of each stage in the pipelining should be equal as much as possible, otherwise the pipelining will be blocked and cut off. A longest stage will become the bottleneck of the pipelining.
- Every functional part of the pipelining must have a buffer register (latch), which is called pipelining register. 每一个功能模块紧接着一个缓存，叫做流水线寄存器，在两个相邻的阶段中传输数据
- Function: transfer data between two adjacent stages to ensure the data to be used later, and separate the processing work of each stage from each other.

Characteristics of pipelining

- Pipelining technology is suitable for a large number of repetitive sequential processes. Only when tasks are continuously provided at the input, the efficiency of pipelining can be brought into full play.

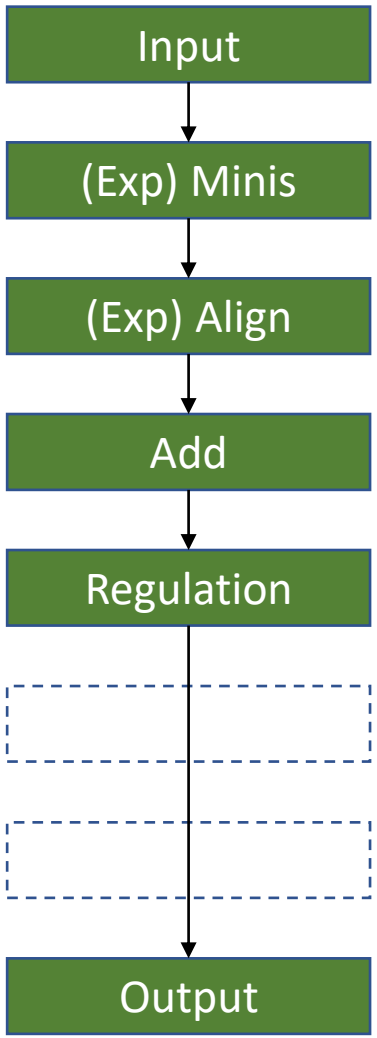
- The pipelining needs the pass time and the empty time
  - Pass time: the time for the first task from beginning (entering the pipelining) to ending.
  - Empty time: the time for the last task from entering the pipelining to have the result.

Classes of pipelining

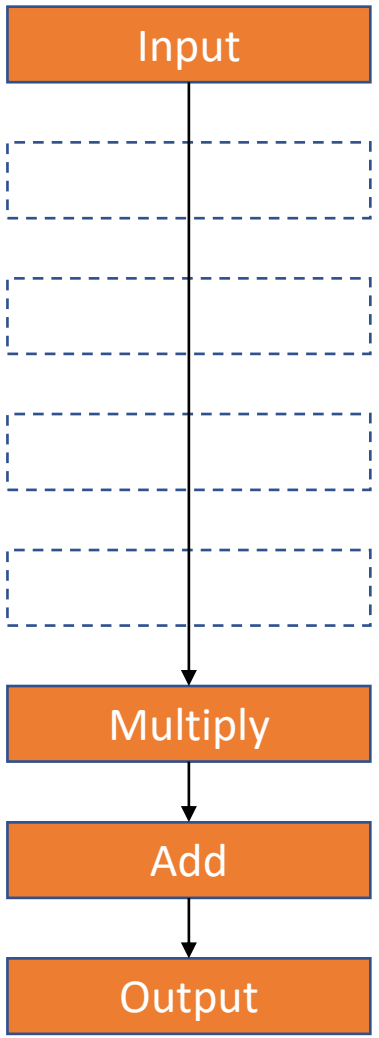
- Single function pipelining: only one fixed function pipelining.
- Multi function pipelining: each stage of the pipelining can be connected differently for several different functions.

§2.3 Classes of pipelining

Segmentation



Floating Point Arithmetic



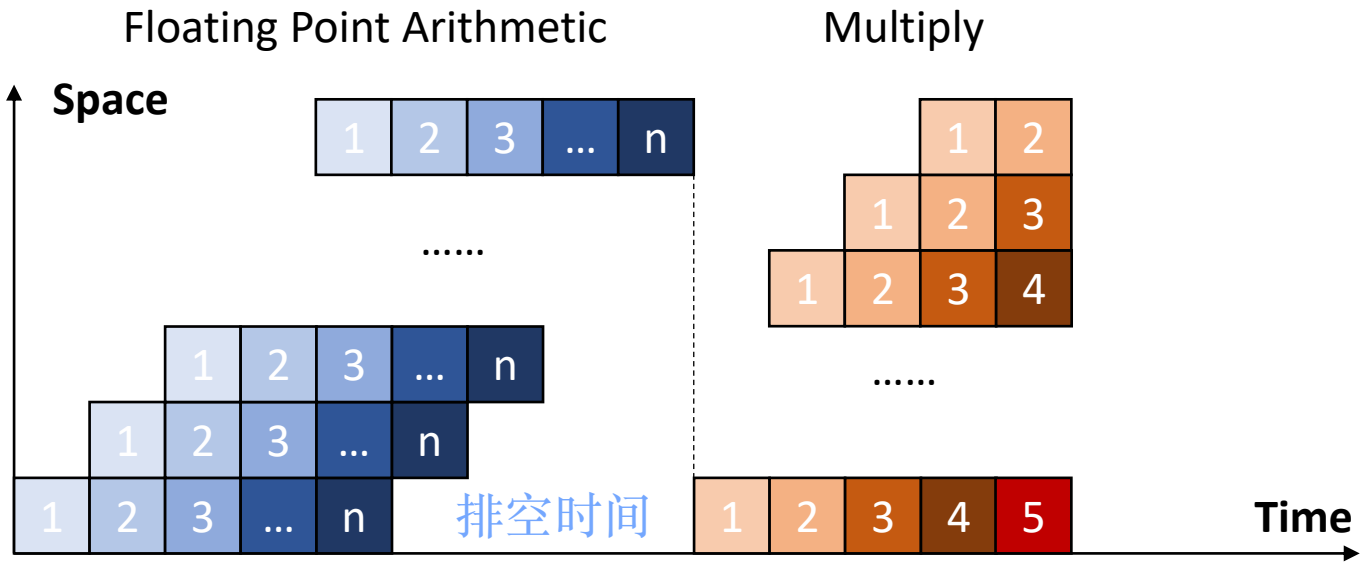
Multiply

Classes of pipelining

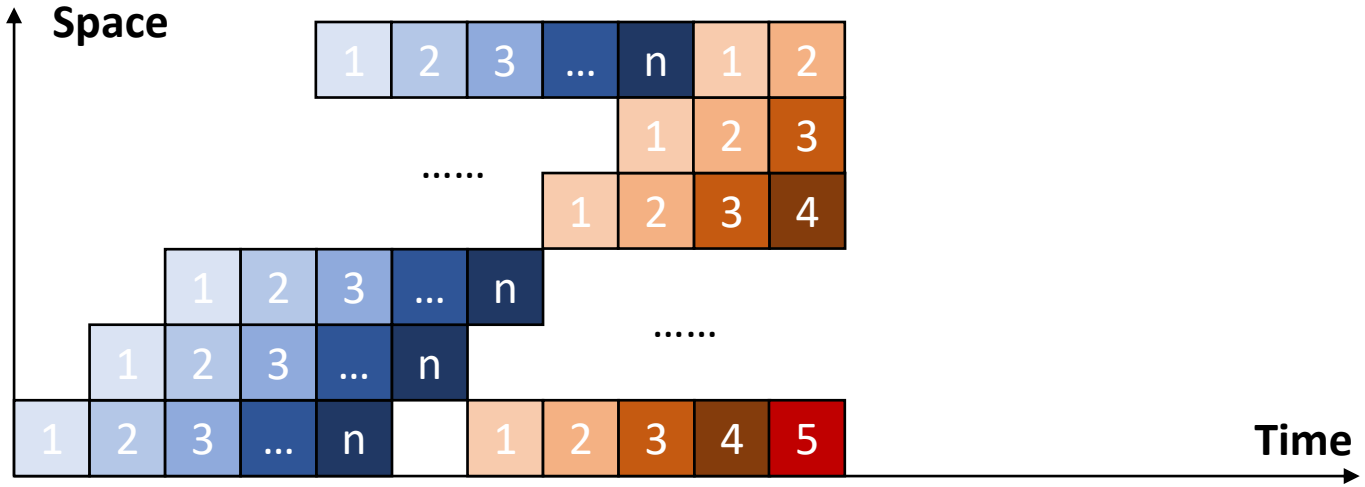
- Static pipelining: In the same time, each segment of the multi-functional pipelining can only work according to the connection mode of the same function.
  - For static pipelining, only the input is a series of the same operation tasks, the efficiency of pipelining can be brought into full play.

- Dynamic pipelining: In the same time, each segment of the multi-functional pipelining can be connected in different ways and perform multiple functions at the same time.
  - It is flexible but with complex control.
  - It can improve the availability of functional units.

Static Pipelining



Dynamic Pipelining

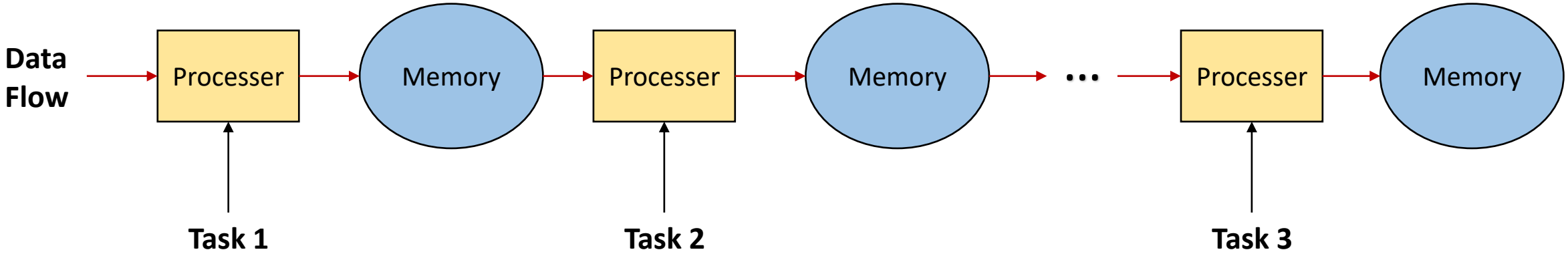


Component level pipelining (in component - operation pipelining) : The arithmetic and logic operation components of the processor are divided into segments, so that various types of operation can be carried out by pipelining.

Processor level pipelining (inter component - instruction pipelining): The interpretation and execution of instructions are implemented through pipelining. The execution process of an instruction is divided into several sub processes, each of which is executed in an independent functional unit.

Inter processor pipelining (inter processor - macro pipelining): It is a serial connection of two or more processors to process the same data stream, and each processor completes a part of the whole task.

§2.3 Classes of pipelining



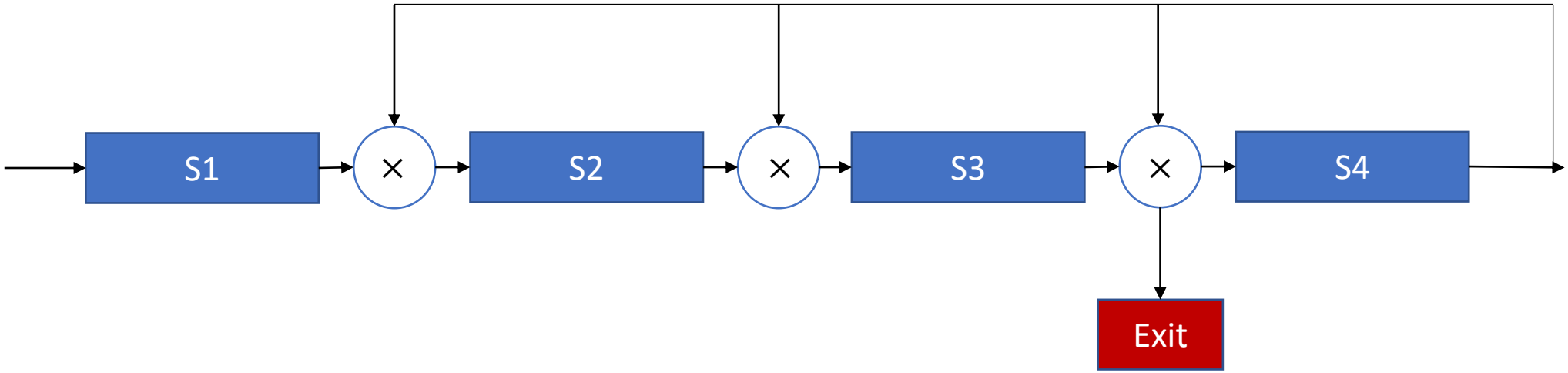
Linear pipelining: Each stage of the pipelining is connected serially without feedback loop. When data passes through each segment in the pipelining, each segment can only flow once at most.

Nonlinear pipelining: In addition to the serial connection, there is also a feedback loop in the pipelining.

Scheduling problem of nonlinear pipelining.

Determine when to introduce a new task to the pipelining, so that the task will not conflict with the task previously entering the pipelining.

Nonlinear pipelining



Task: → S1 → S2 → S3 → S4 → S2 → S3 → S4 → S3 →

Ordered pipelining: In the pipelining, the outflow order of tasks is exactly the same as the inflow order. Each task flows by sequence in each segment of the pipelining.

Disordered pipelining: In the pipelining, the outflow order of tasks is not the same as the inflow order. The later tasks are allowed completed first.

Scalar processor: The processor does not have vector data representation and vector instructions, and only deal with scalar data through pipelining.

Vector pipelining processor: The processor has vector data representation and vector instructions. It is the combination of vector data representation and pipelining technology.