

ART: Customizing Accelerators for DNN-Enabled Real-Time Safety-Critical Systems

Great Lakes Symposium on VLSI (GLSVLSI) 2025, New Orleans, LA, USA

Shixin Ji*, Xingzhen Chen*, Wei Zhang*, Zhuoping Yang*, Jinming Zhuang*, Sarah Schultz*,
Yukai Song[‡], Jingtong Hu[‡], Alex K. Jones[§], Zheng Dong[†], Peipei Zhou*

Brown University*; Wayne State University[†] ;
University of Pittsburgh[‡] ;Syracuse University[§]

shixin_ji@brown.edu
peipei_zhou@brown.edu

<https://peipeizhou-eecs.github.io/>



BROWN



WAYNE STATE
UNIVERSITY



University of
Pittsburgh®

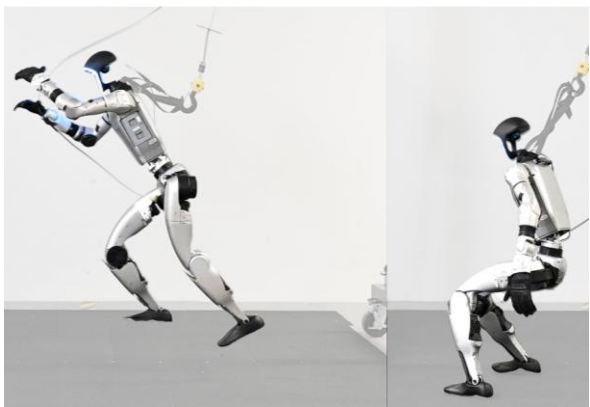


Syracuse University

Proliferate of Real-time Safety-critical Systems



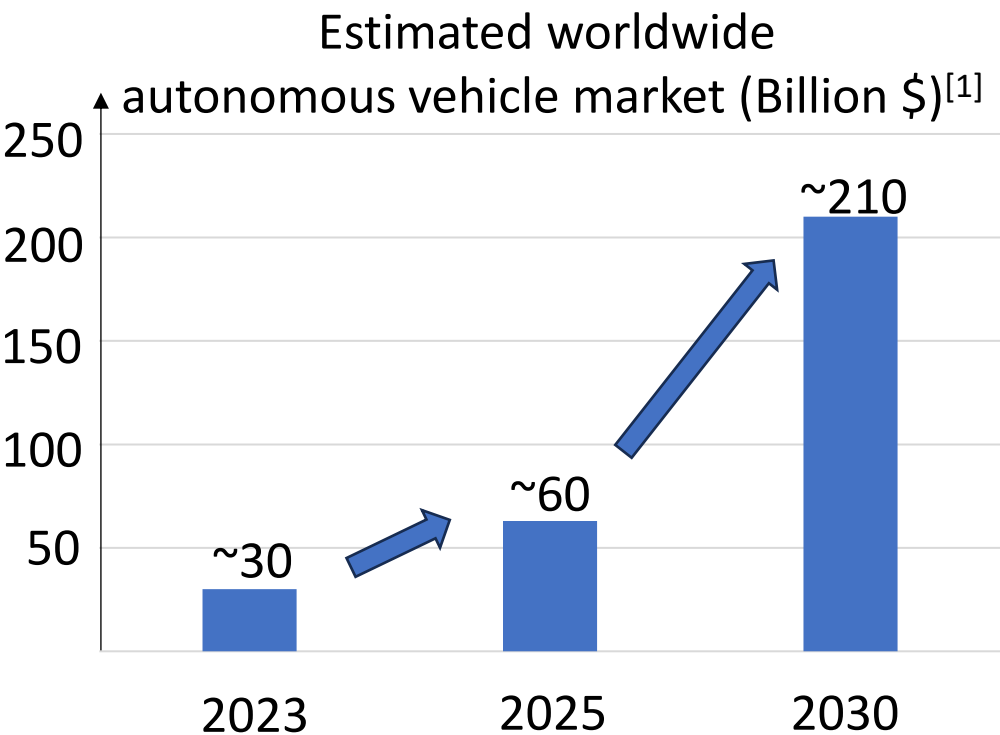
Autonomous driving



Robotics^[2]



UAVs



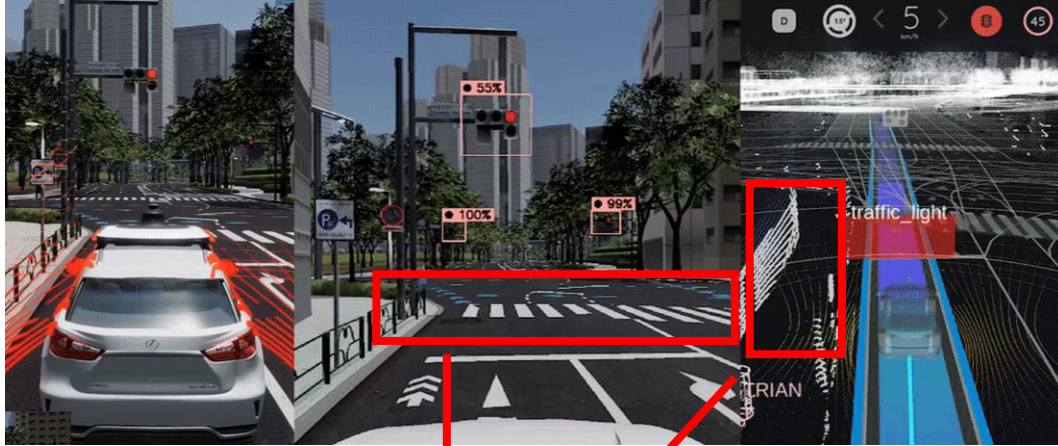
[1] <https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/autonomous-drivings-future-convenient-and-connected>

[2] He T, Gao J, Xiao W, et al. ASAP: Aligning Simulation and Real-World Physics for Learning Agile Humanoid Whole-Body Skills[J]. arXiv preprint arXiv:2502.01143, 2025.

Why Safety-critical



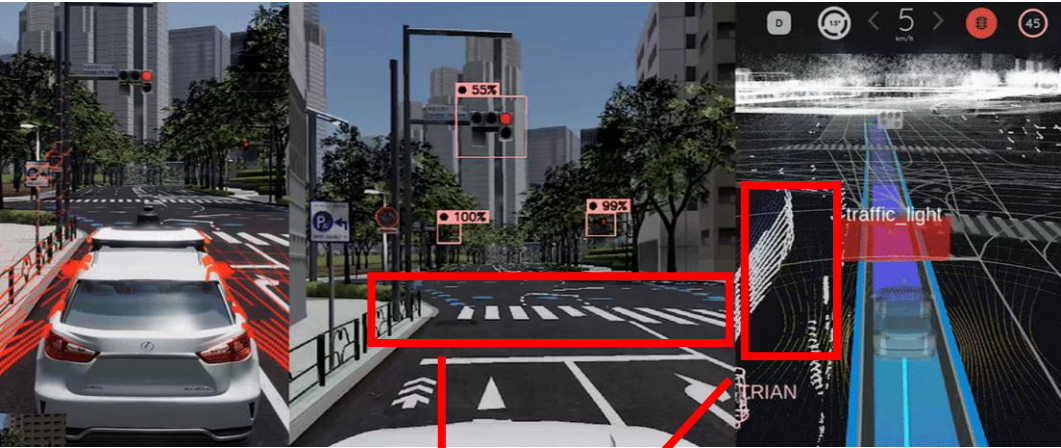
Why Safety-critical



Pedestrian
detection



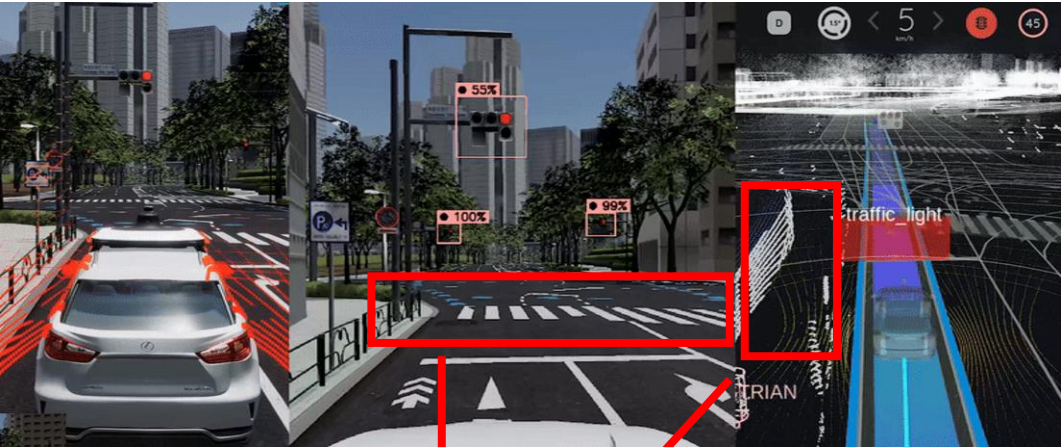
Why Safety-critical



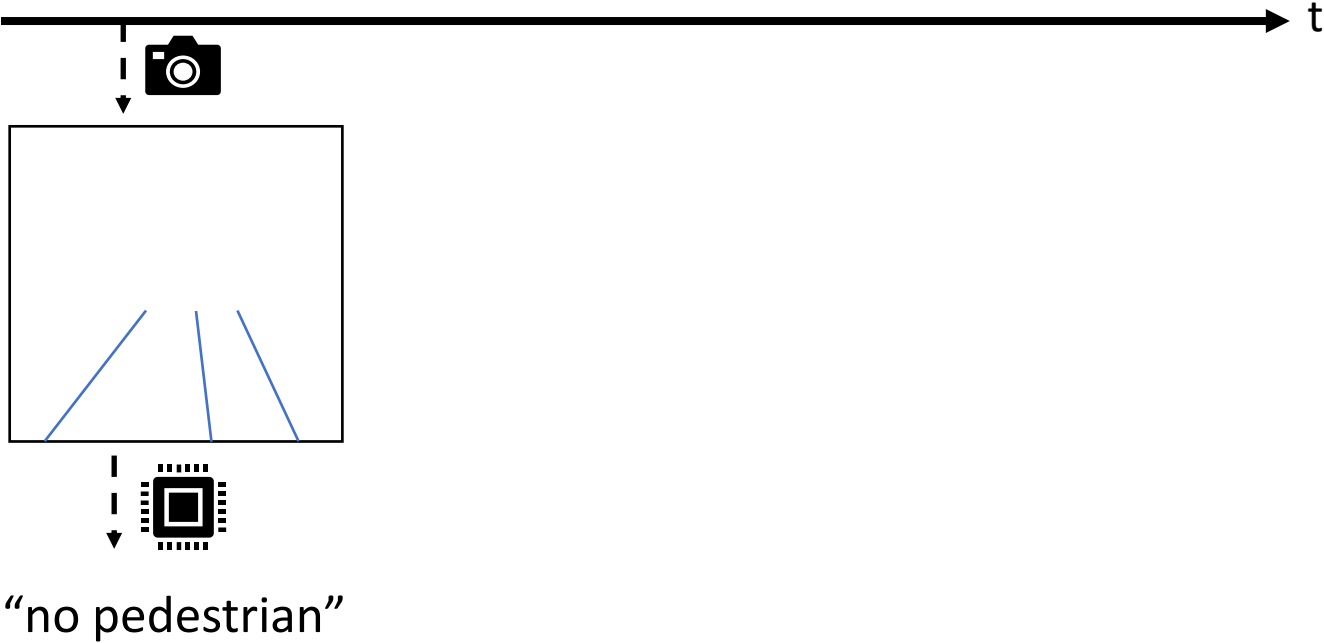
 Pedestrian
detection



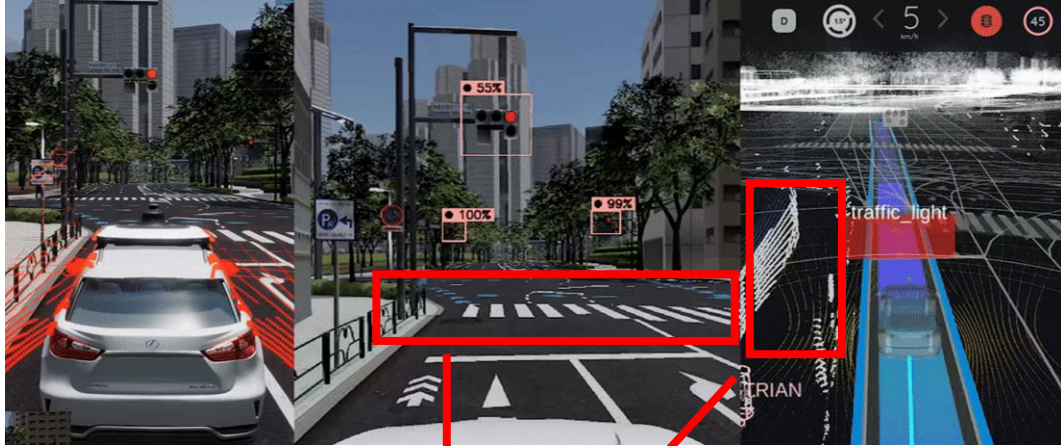
Why Safety-critical



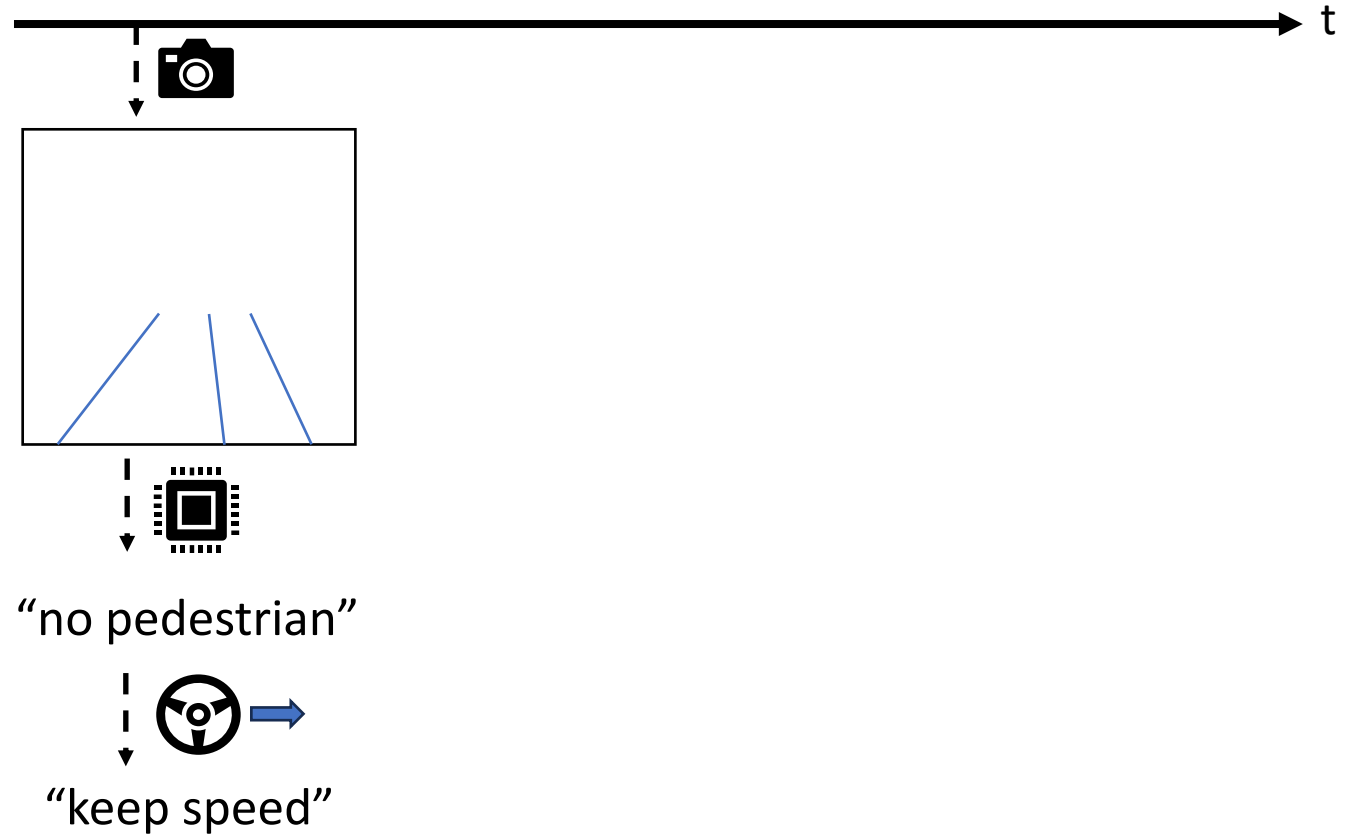
 Pedestrian detection



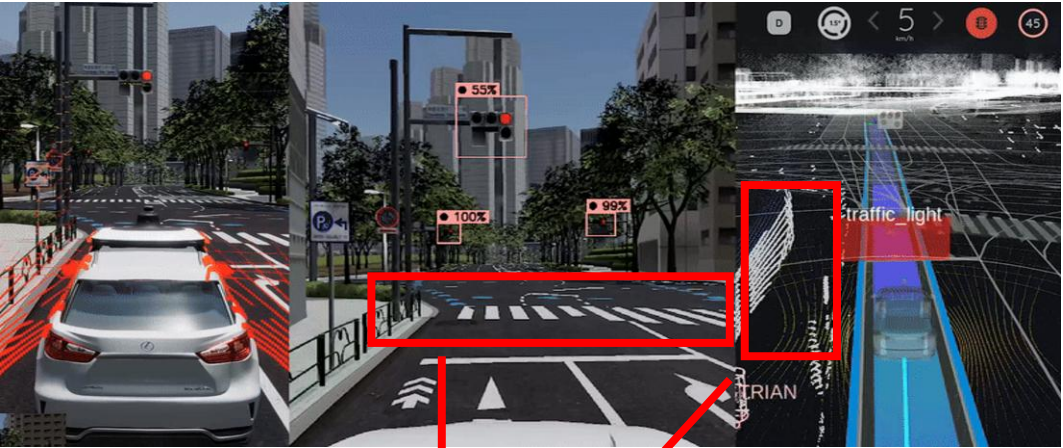
Why Safety-critical



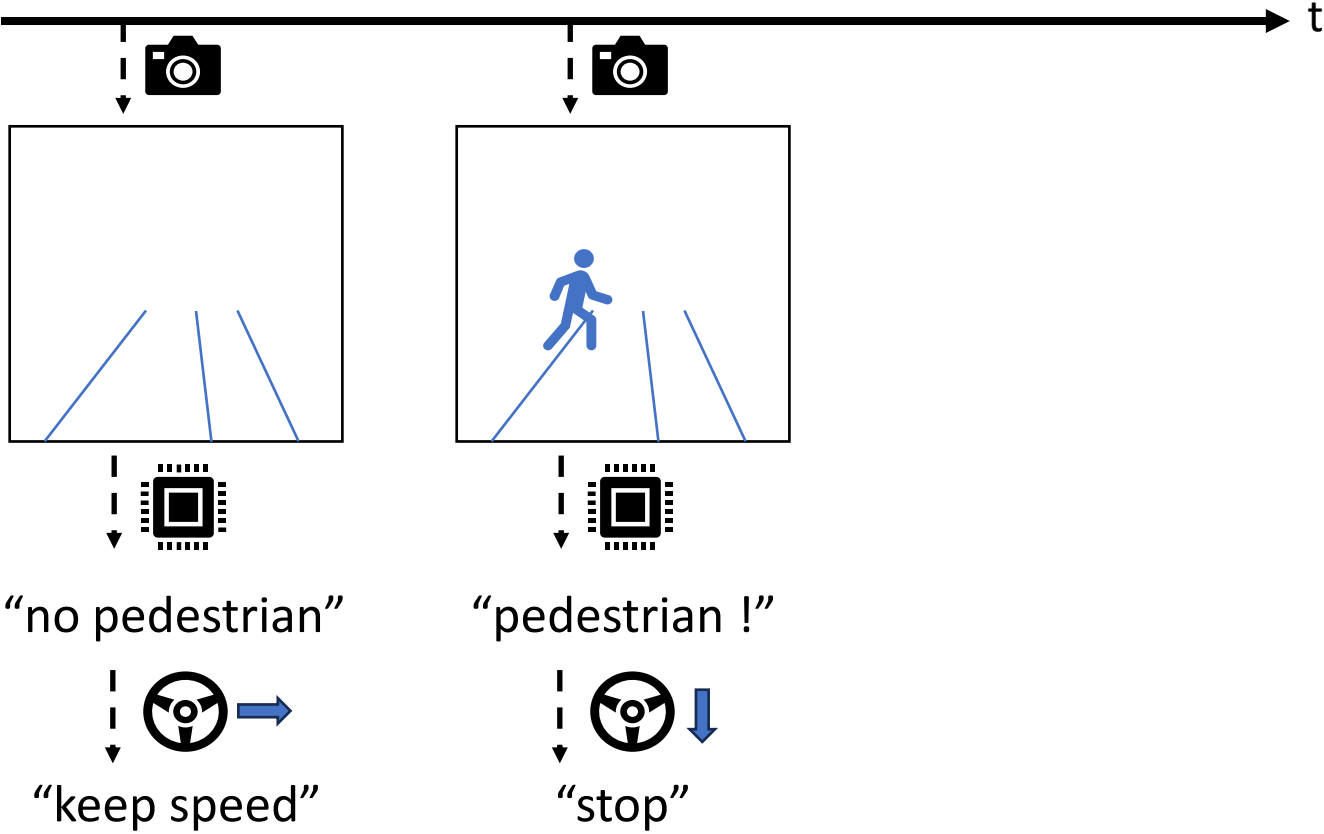
 Pedestrian
detection



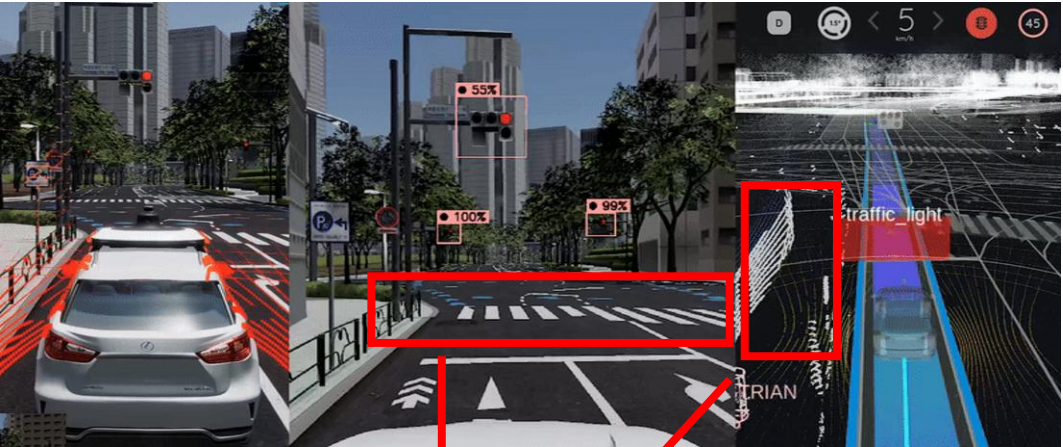
Why Safety-critical



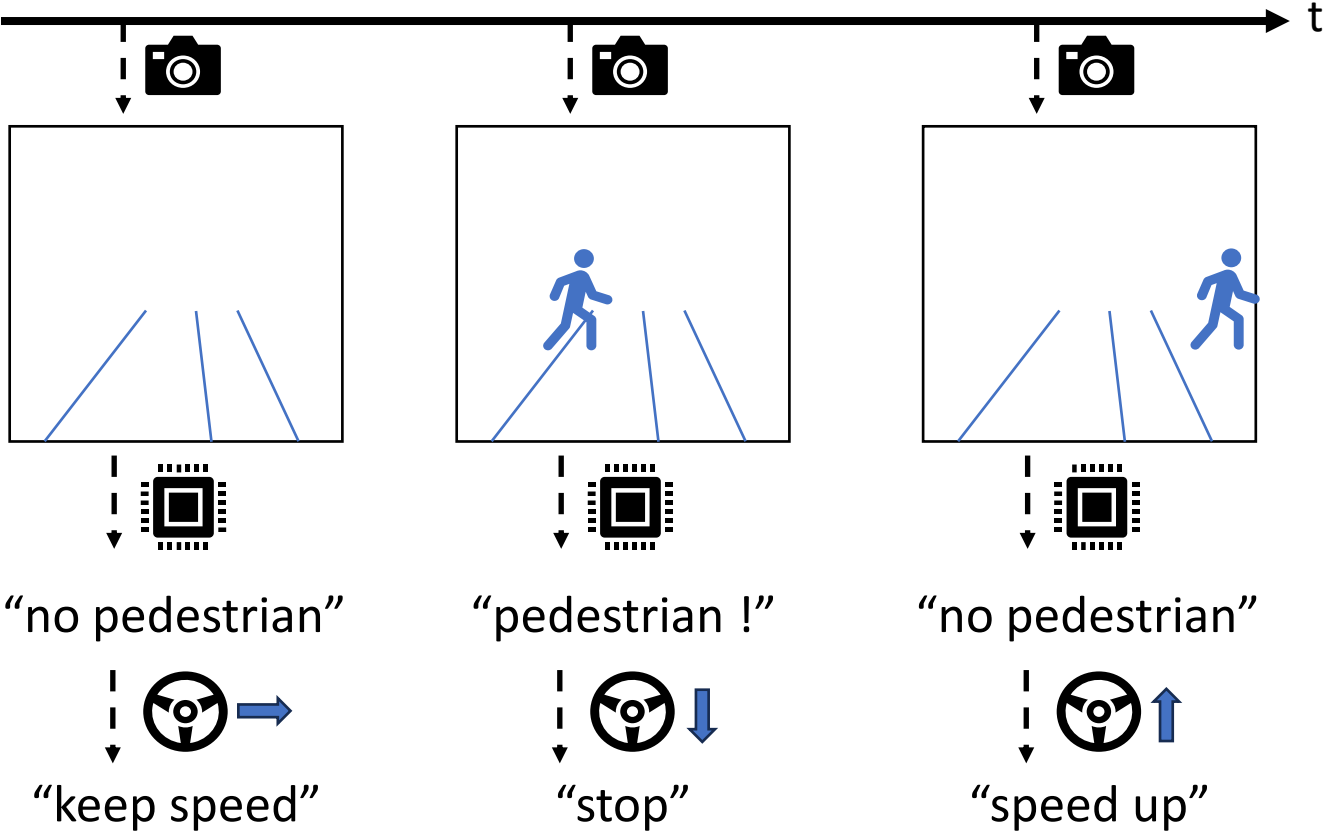
 Pedestrian
detection



Why Safety-critical



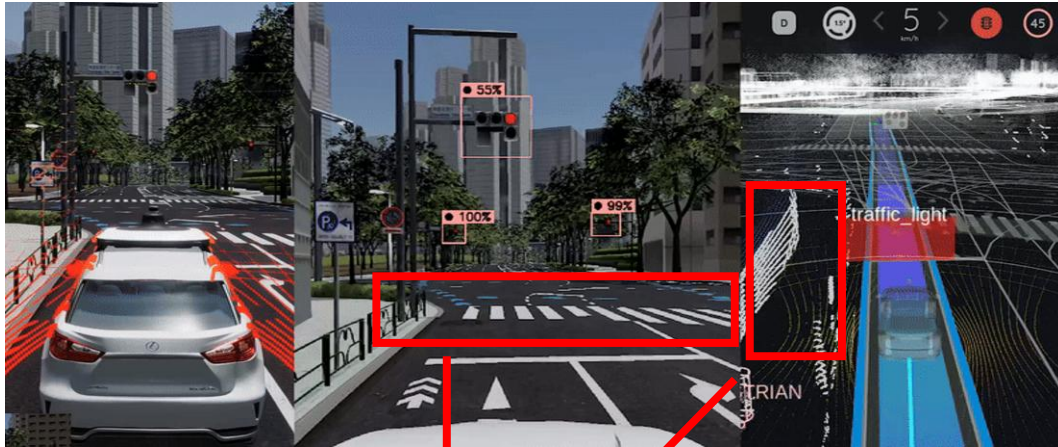
 Pedestrian detection



Why Safety-critical

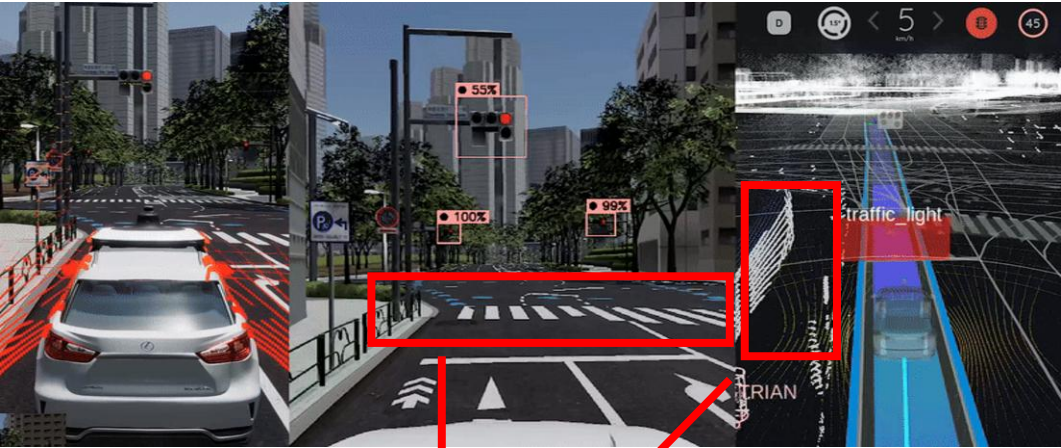
If the accelerator does not finish on time:

—————→ t



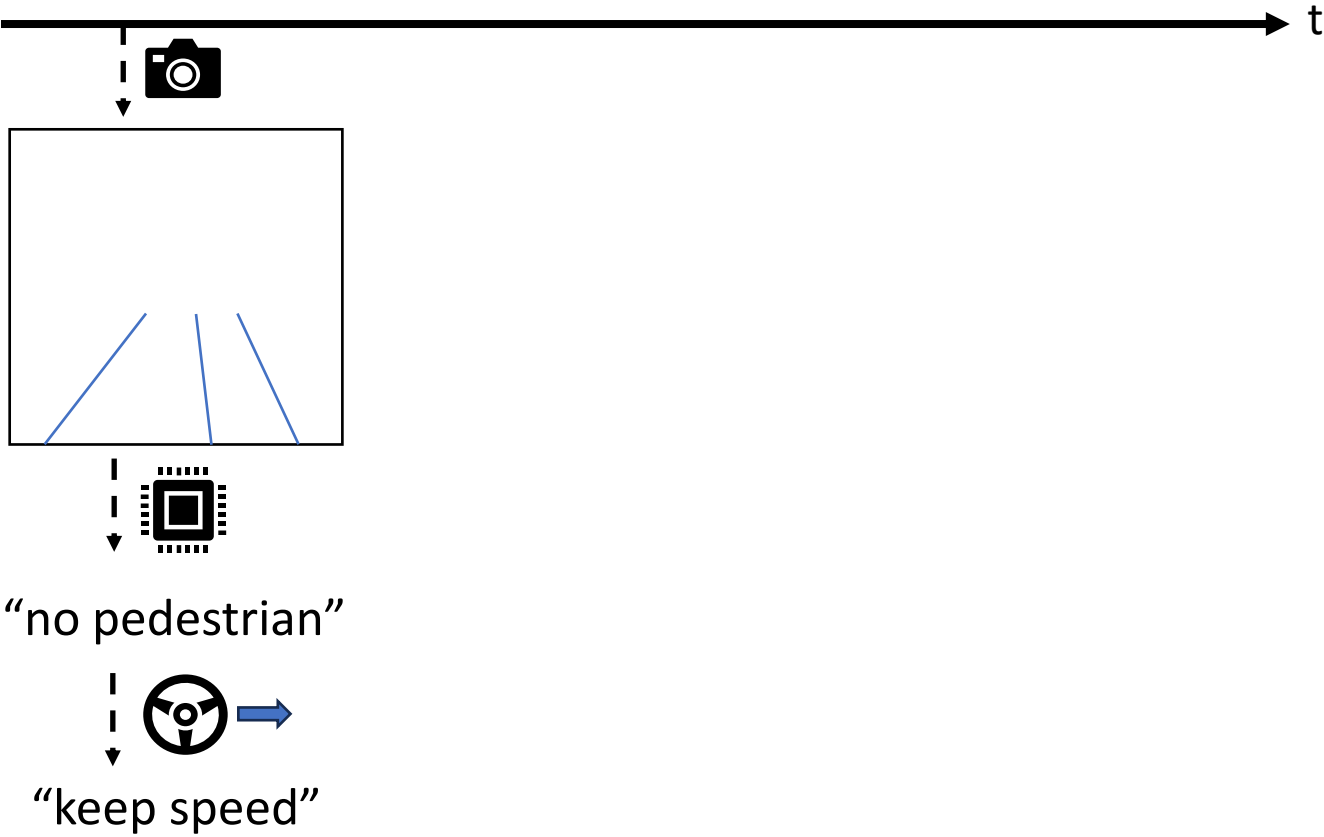
Pedestrian
detection

Why Safety-critical

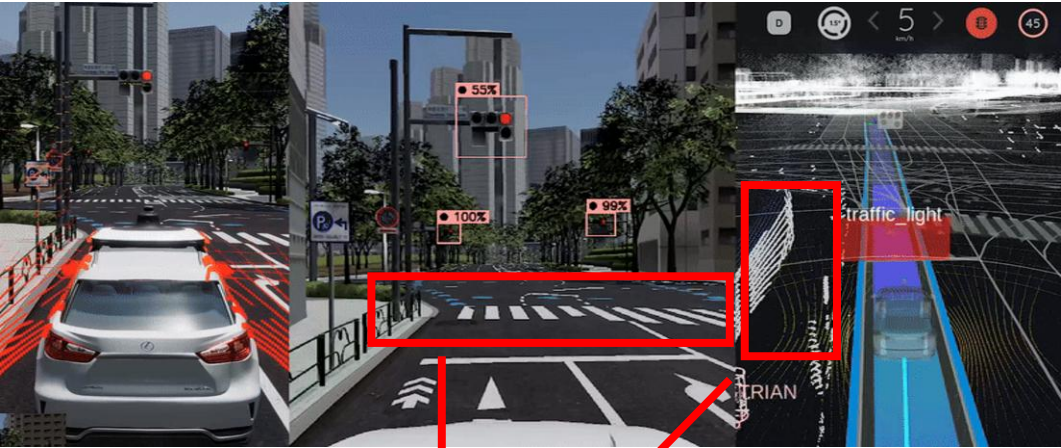


Pedestrian
detection

If the accelerator does not finish on time:

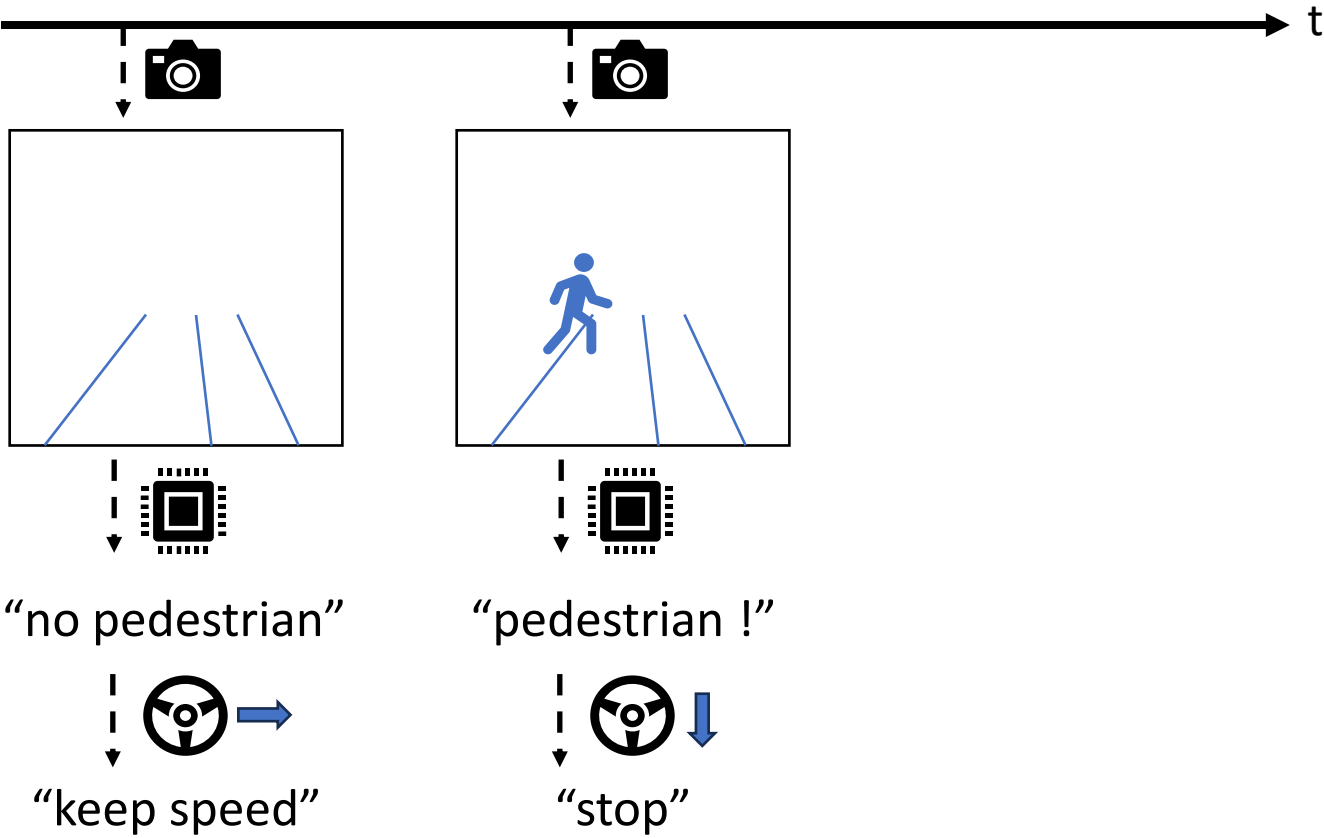


Why Safety-critical

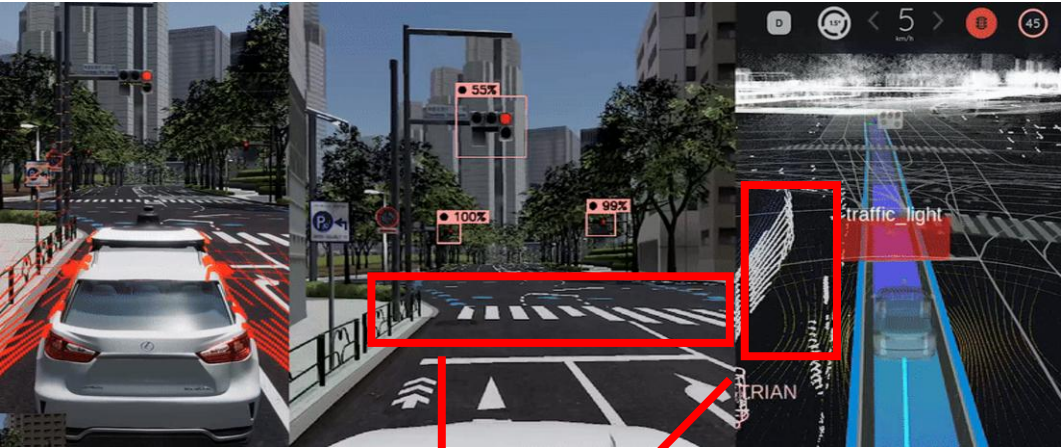


Pedestrian
detection

If the accelerator does not finish on time:

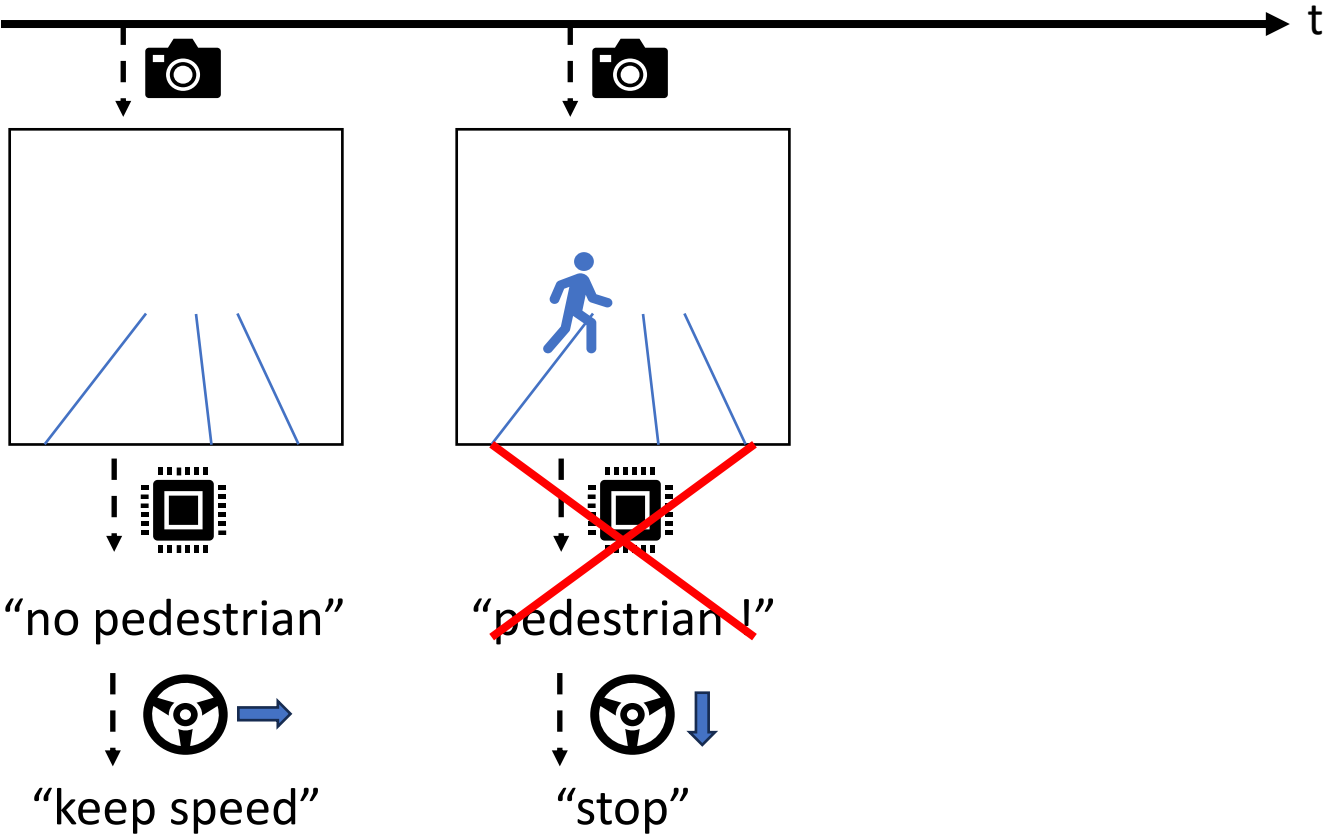


Why Safety-critical

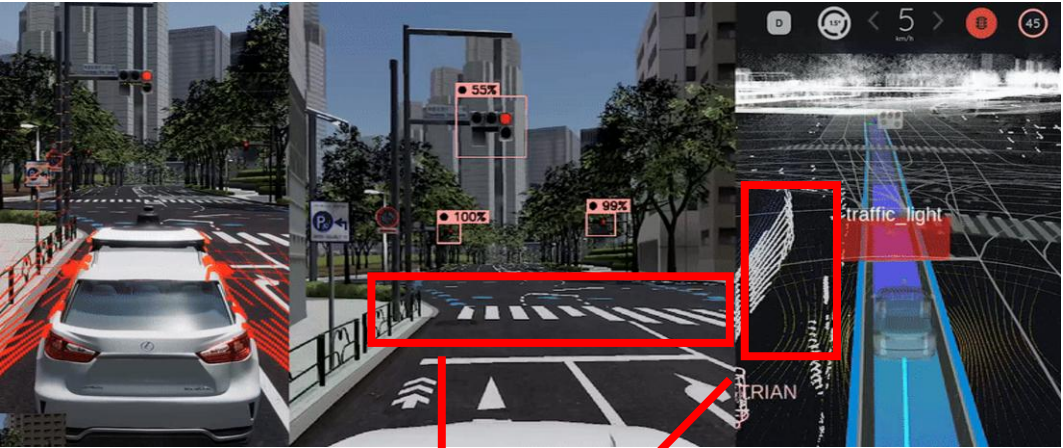


Pedestrian
detection

If the accelerator does not finish on time:

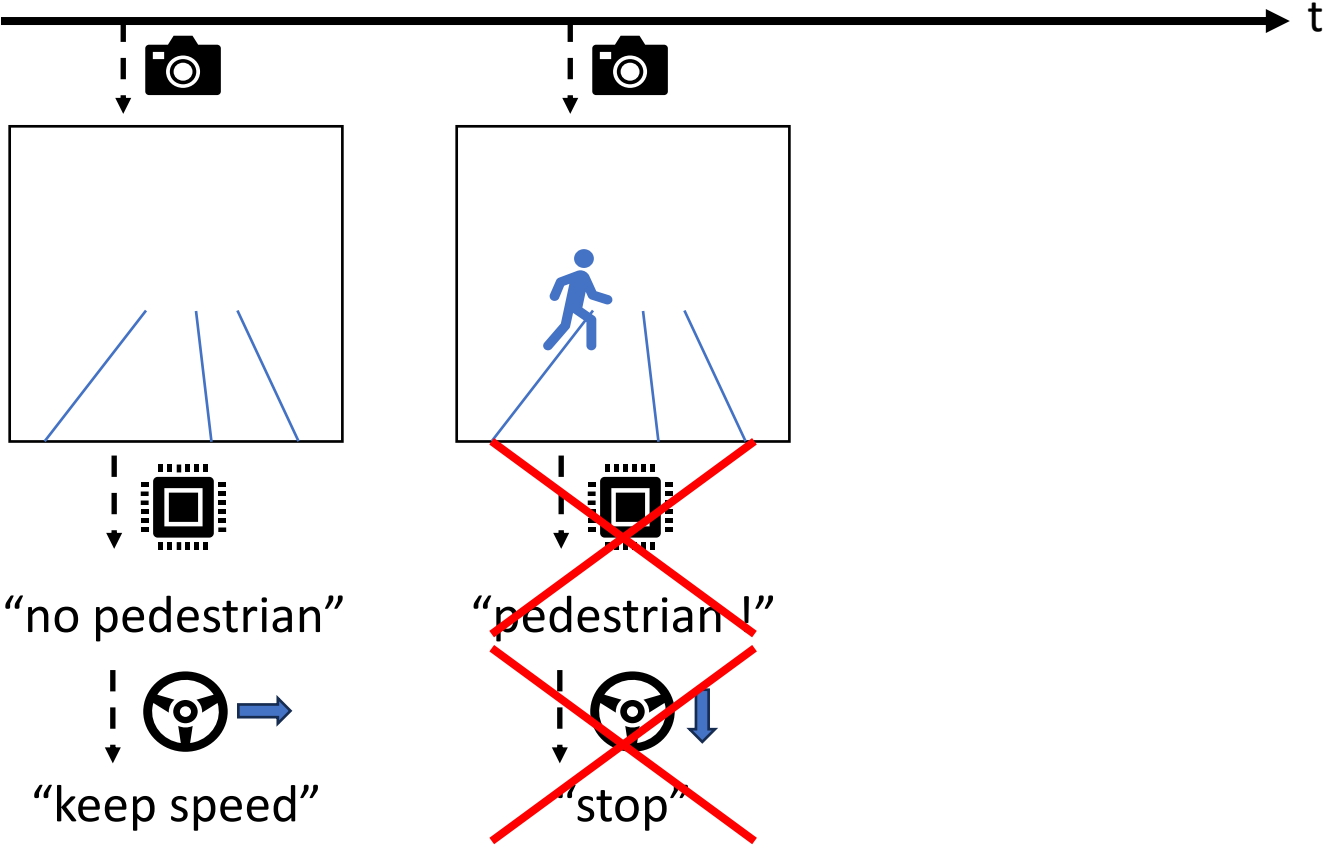


Why Safety-critical

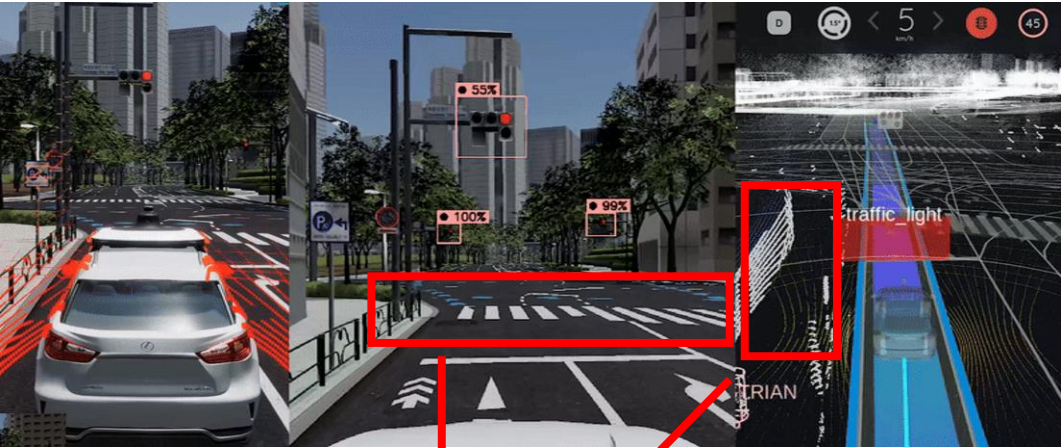


Pedestrian
detection

If the accelerator does not finish on time:

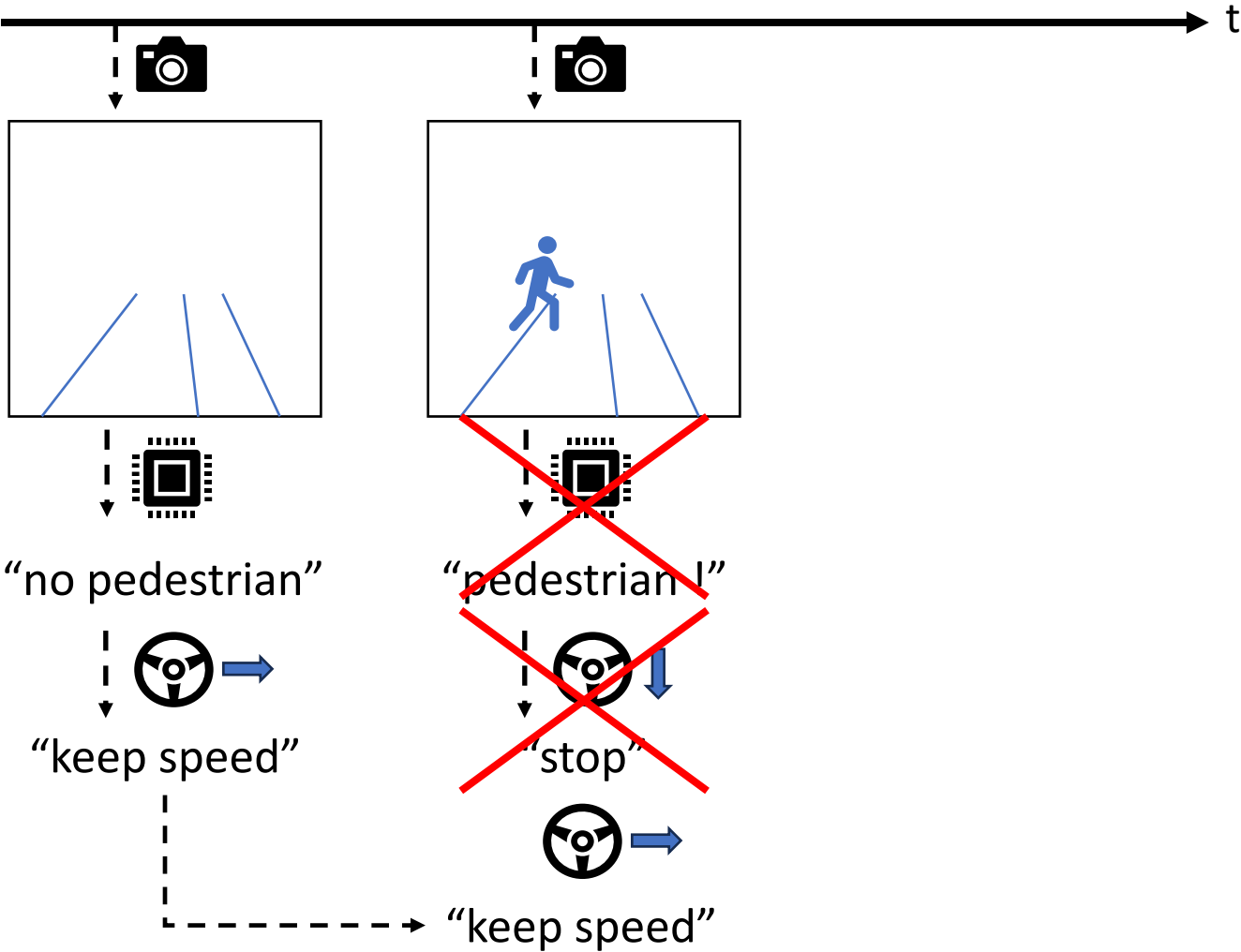


Why Safety-critical

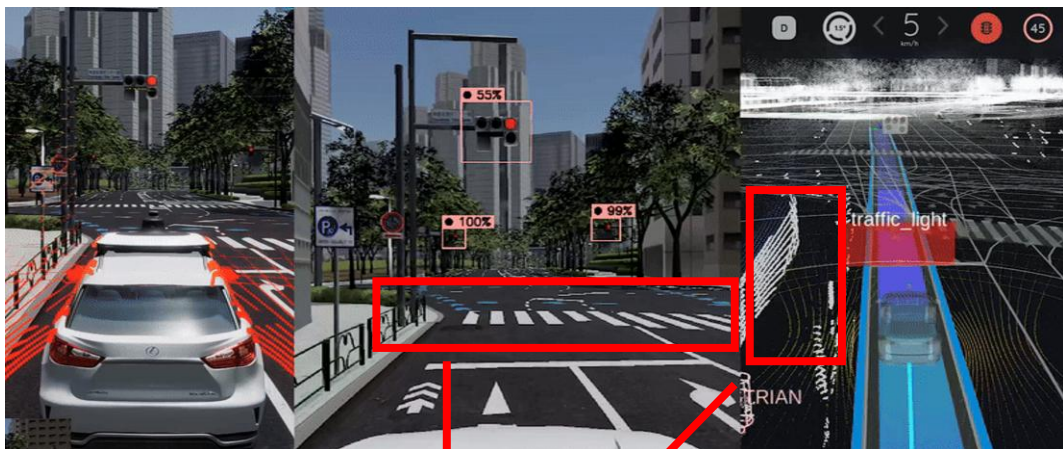


Pedestrian
detection

If the accelerator does not finish on time:

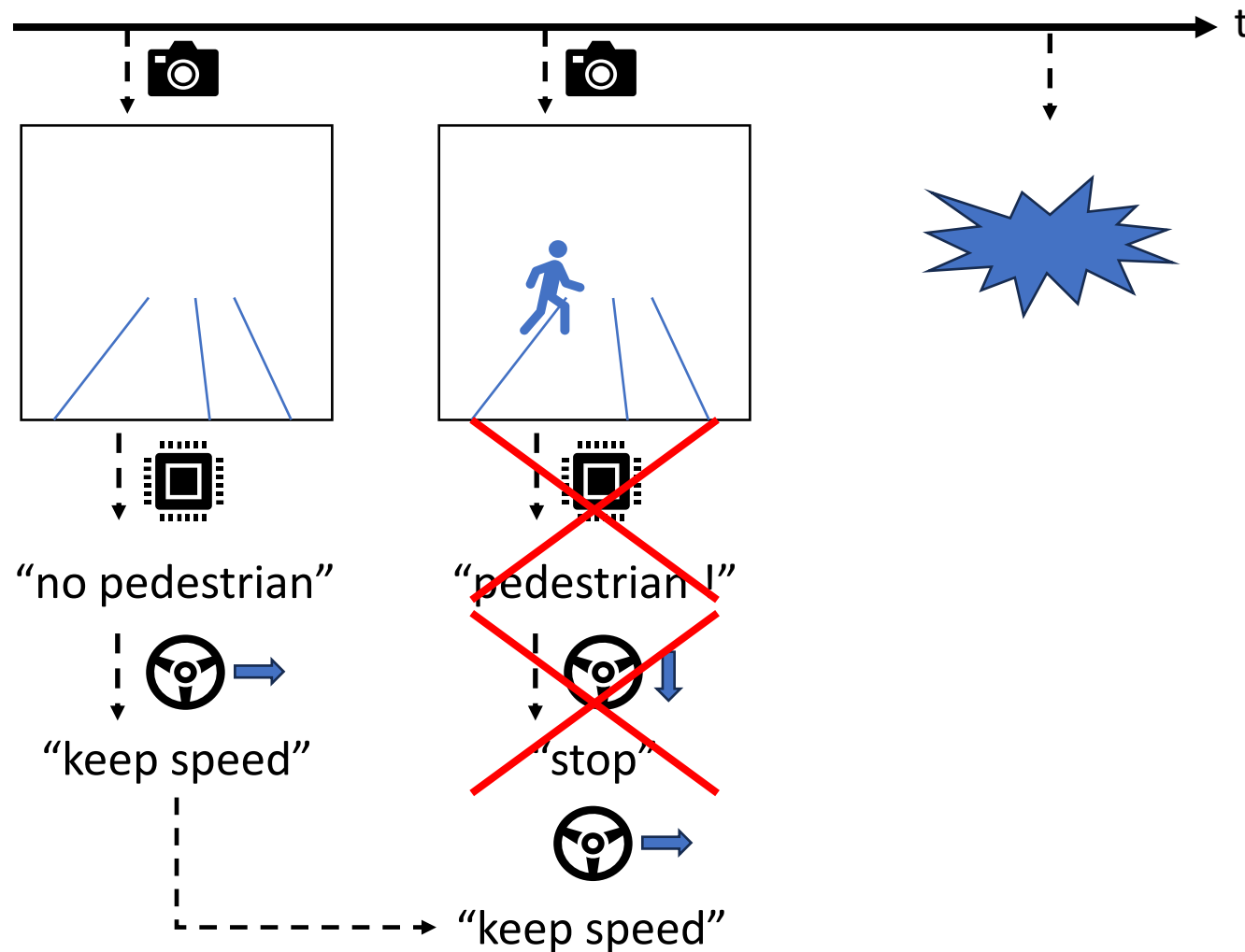


Why Safety-critical

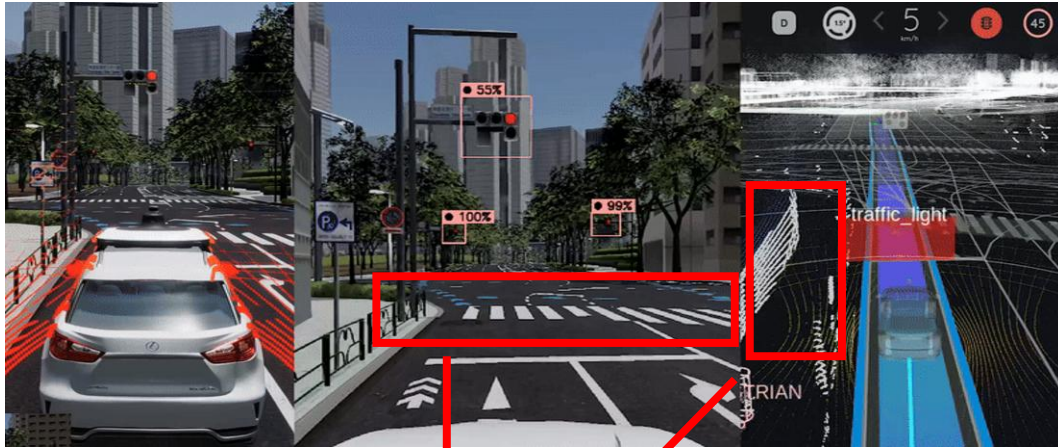


Pedestrian
detection

If the accelerator does not finish on time:



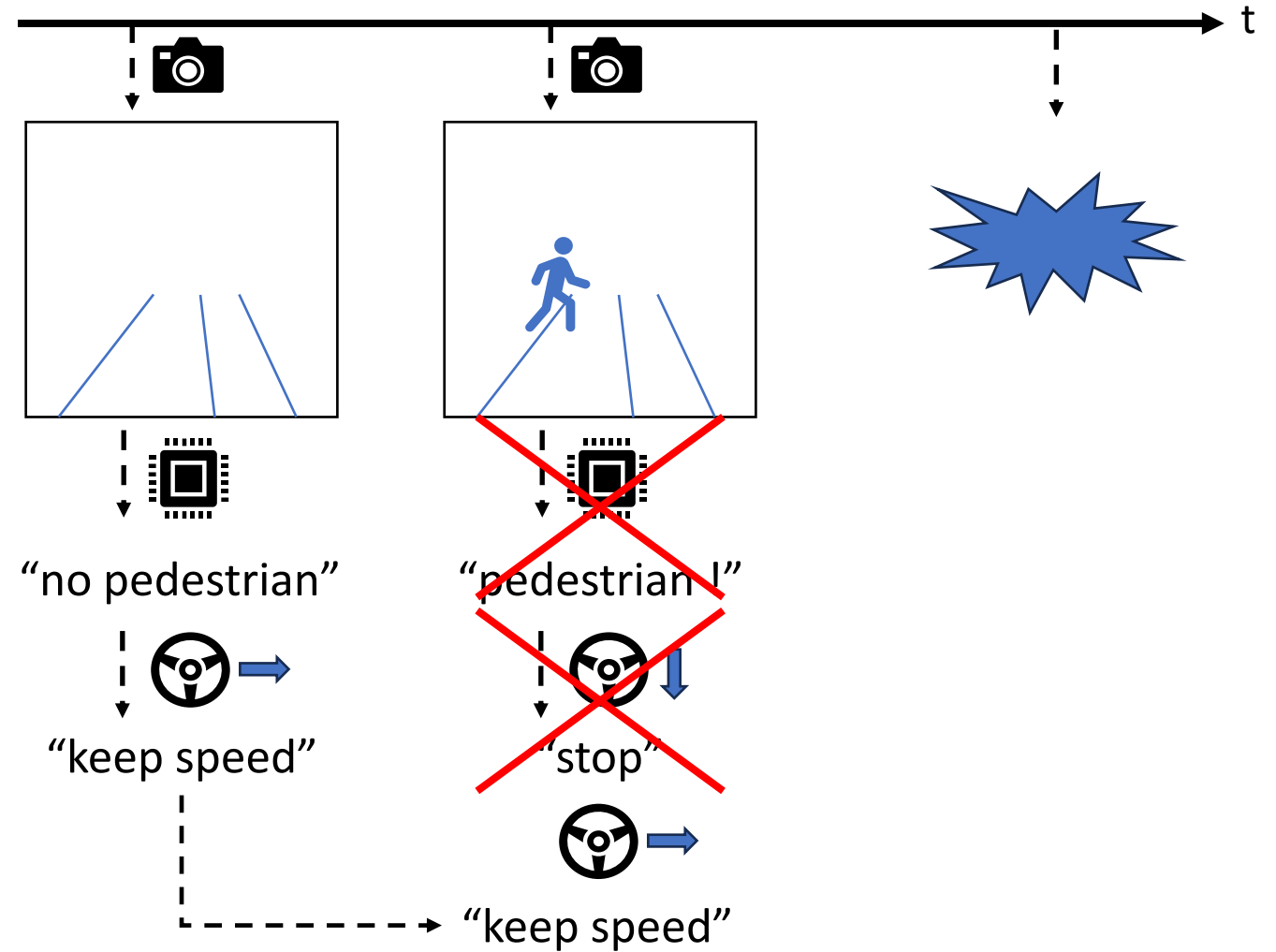
Why Safety-critical



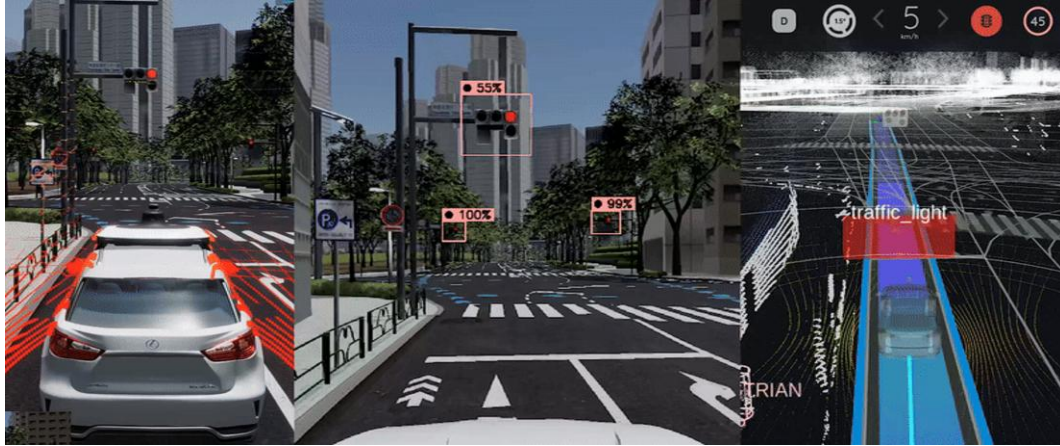
Pedestrian
detection

**Real-time safety critical system:
Each frame/job matters**

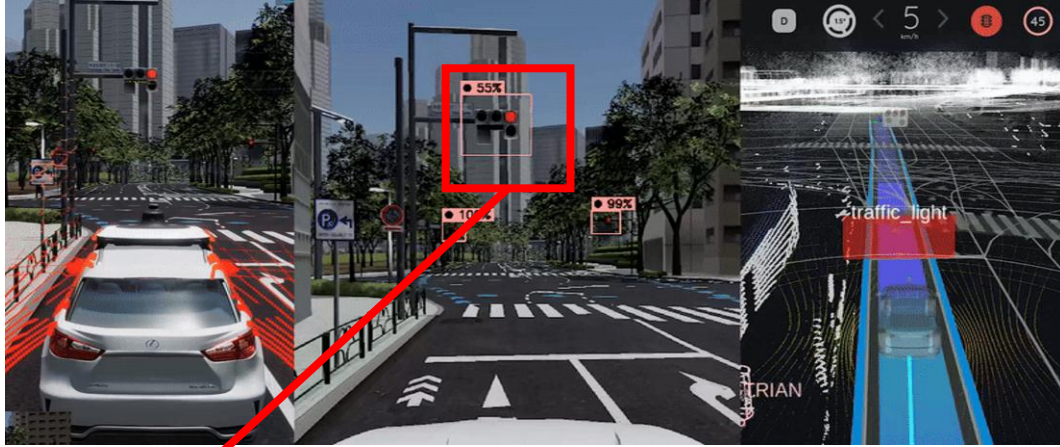
If the accelerator does not finish on time:



Real-time System Modeling



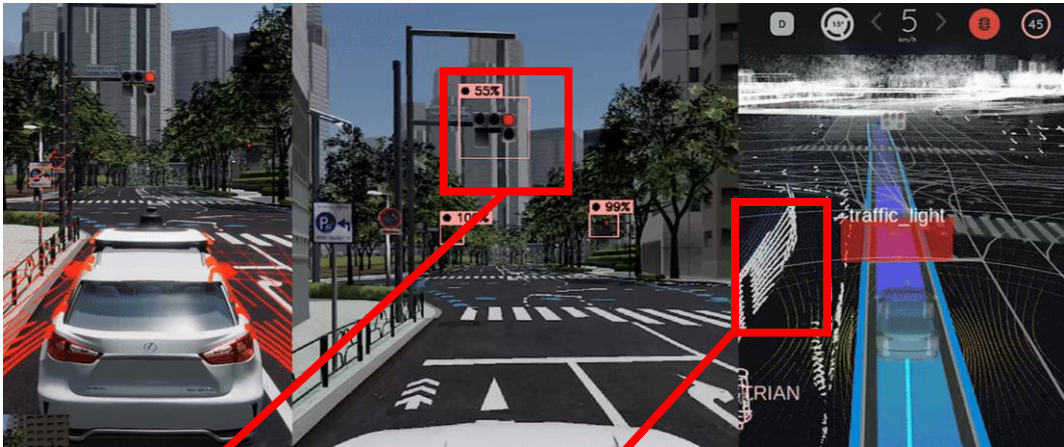
Real-time System Modeling



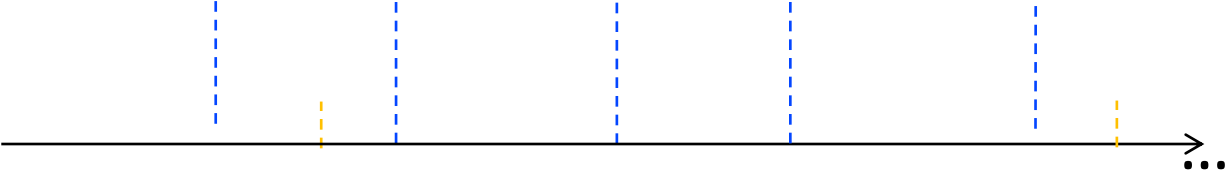
Traffic light
detection



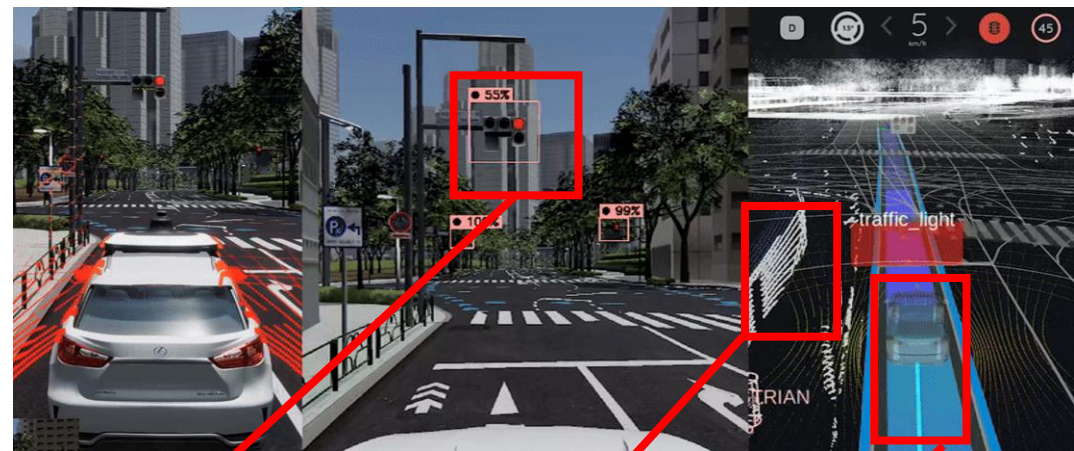
Real-time System Modeling




 Traffic light
detection  Pedestrian
detection





Real-time System Modeling



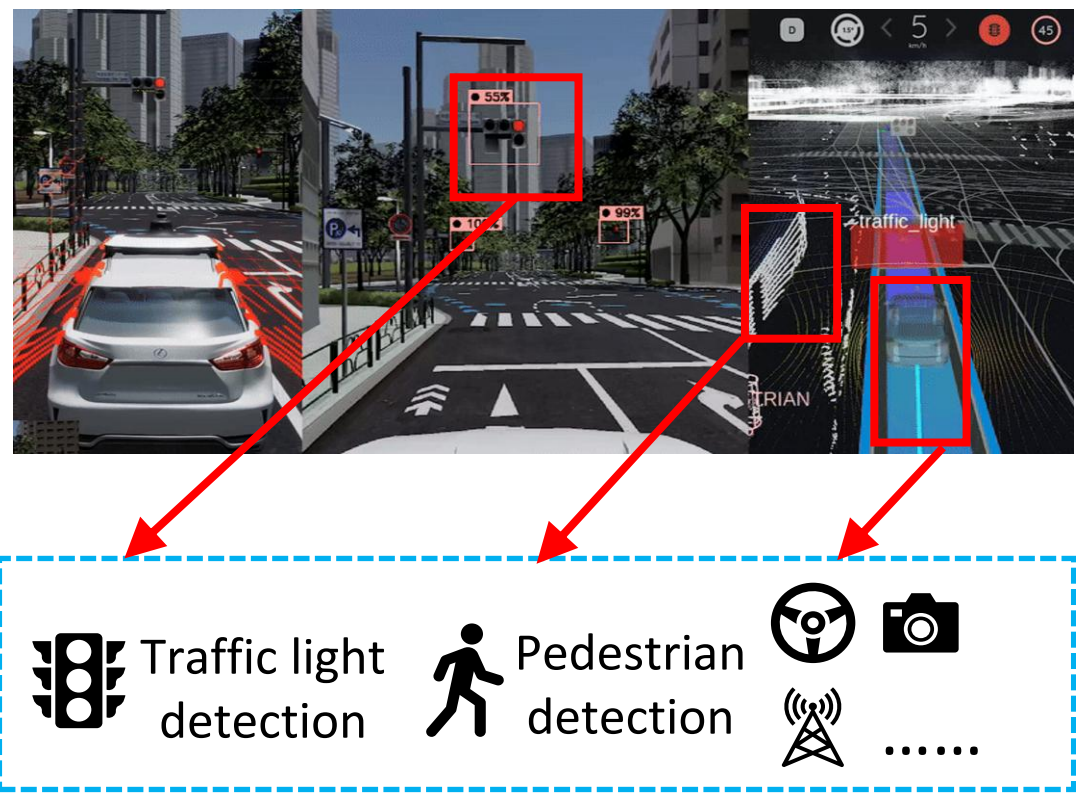
 Traffic light
detection

 Pedestrian
detection

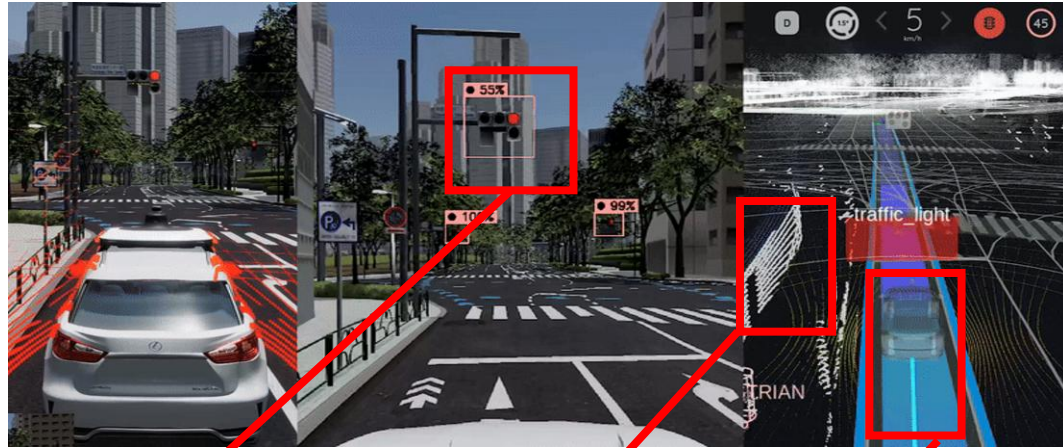





Real-time System Modeling



Real-time System Modeling



Traffic light
detection



Pedestrian
detection

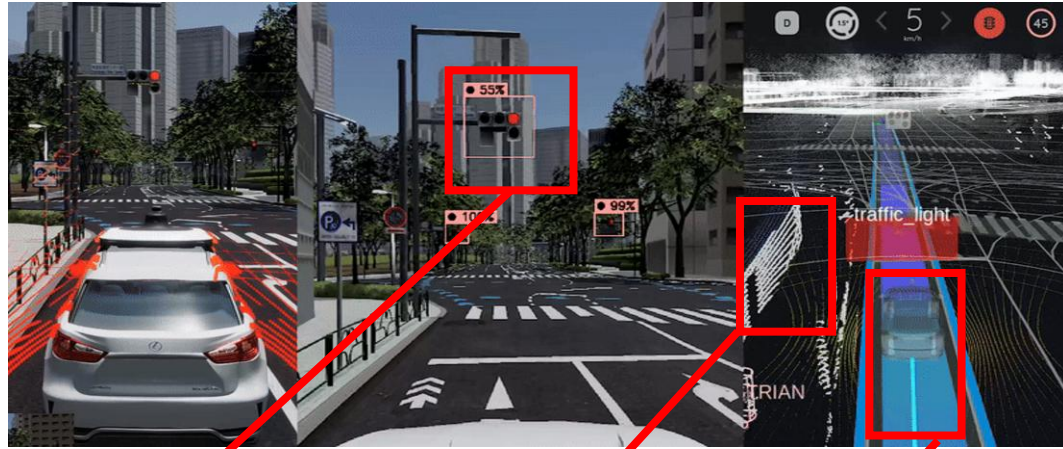


.....



- A task set: multiple tasks

Real-time System Modeling



Traffic light
detection



Pedestrian
detection



.....

Task1



Task2



Job1



Job2

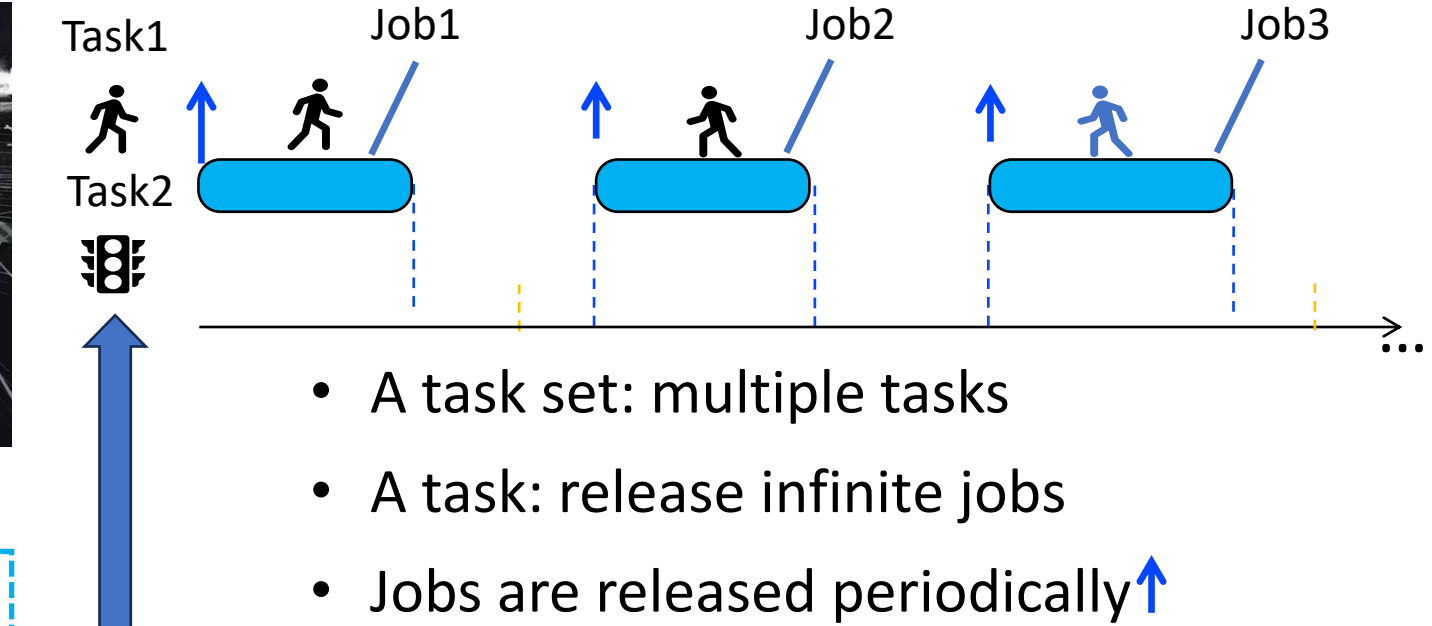
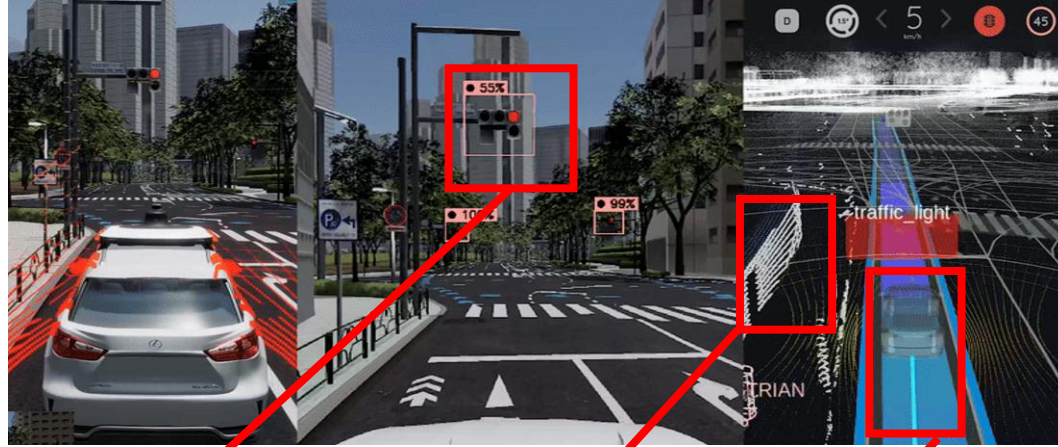


Job3

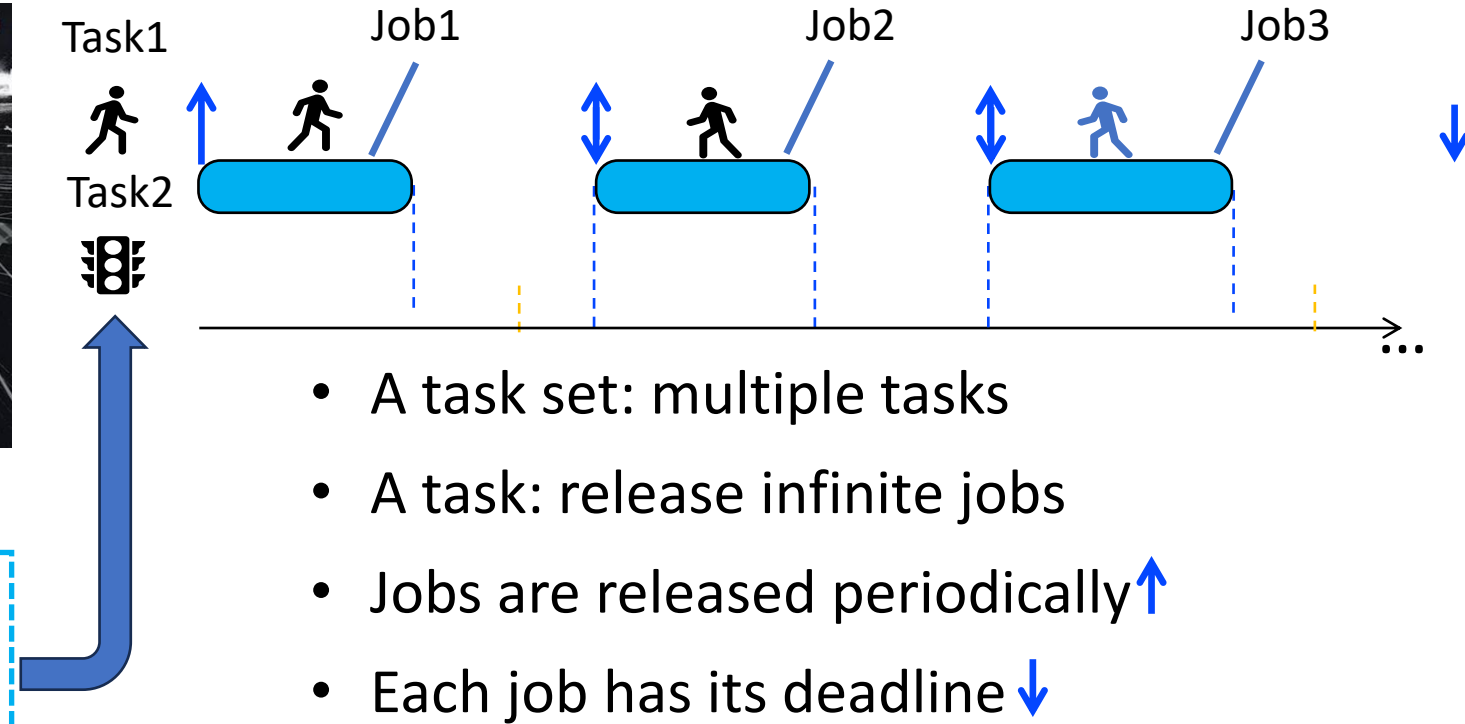
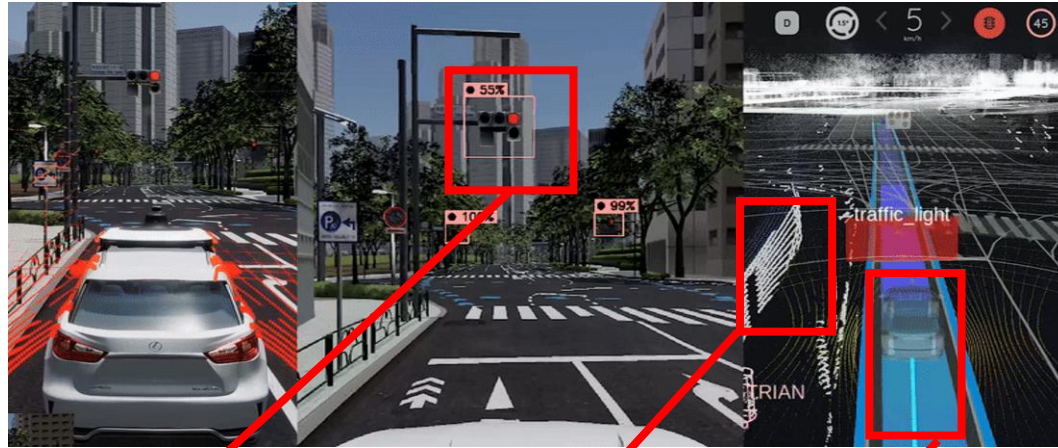


- A task set: multiple tasks
- A task: release infinite jobs

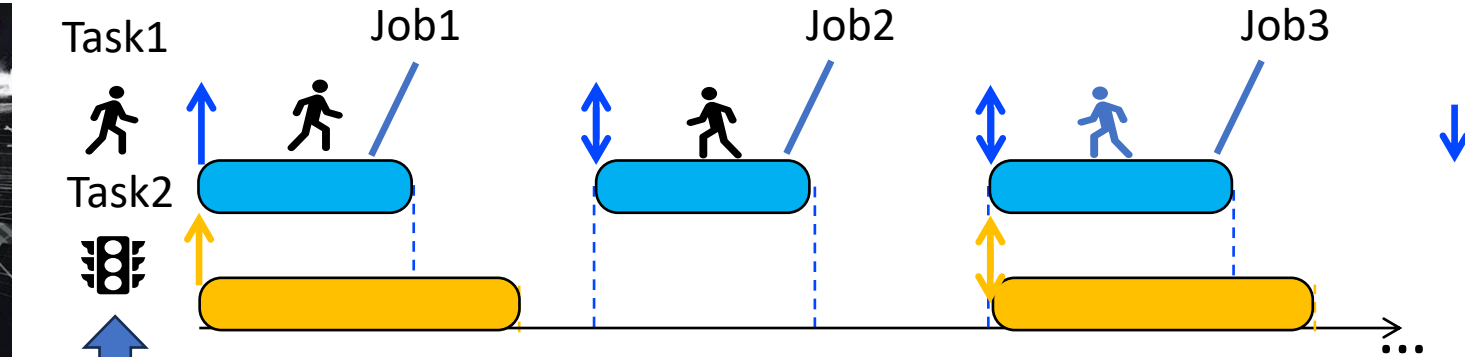
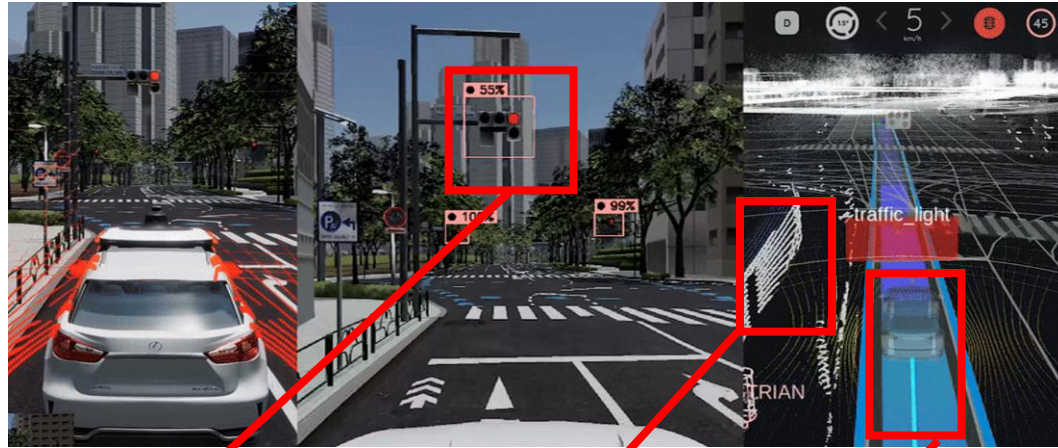
Real-time System Modeling



Real-time System Modeling

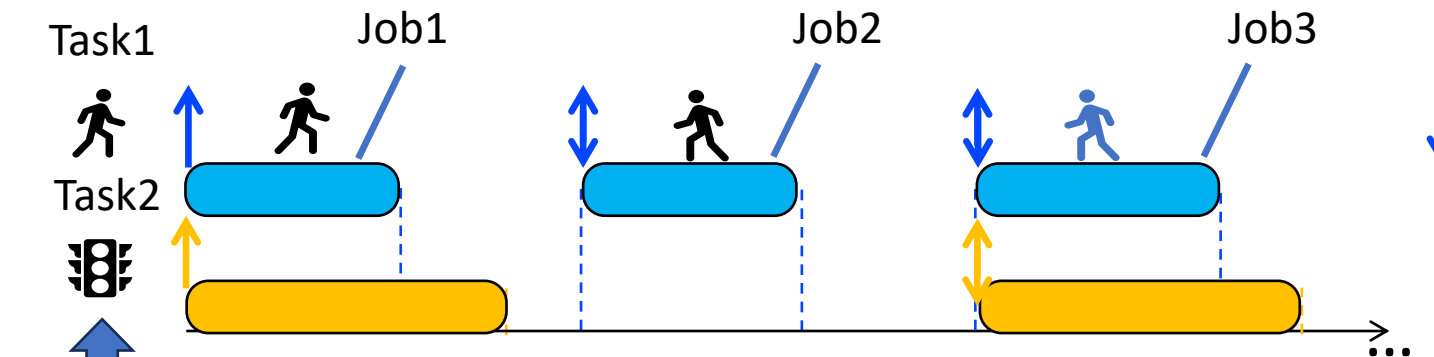
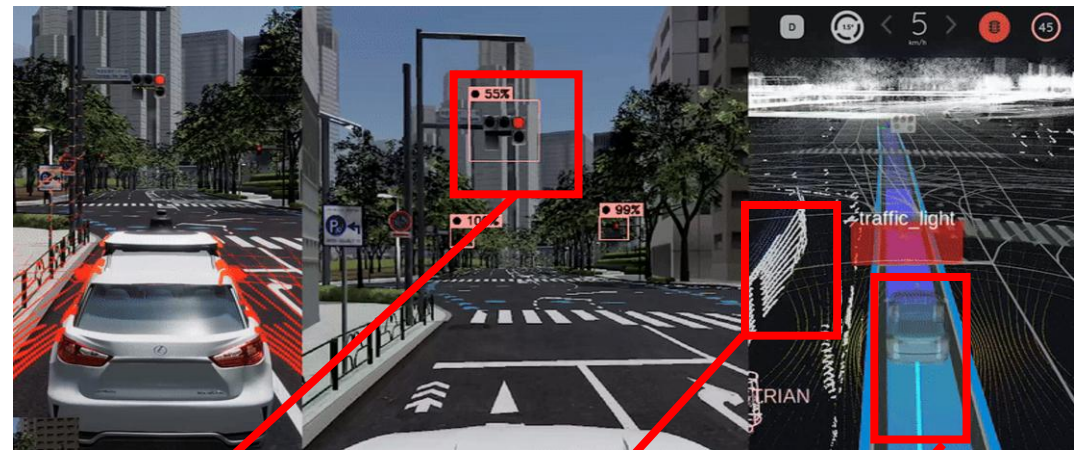


Real-time System Modeling

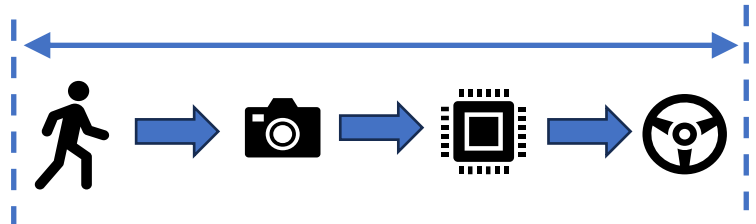


- A task set: multiple tasks
- A task: release infinite jobs
- Jobs are released periodically ↑
- Each job has its deadline ↓

Real-time System Modeling



- A task set: multiple tasks
- A task: release infinite jobs
- Jobs are released periodically ↑
- Each job has its deadline ↓



Latency bounded → System predictable

Motivation Example: 2 Tasks, 1 Single Acc

Task 1 : 10 ms execution time, 20 ms period

Task 2: 190 ms execution time, 400 ms period (assuming ddl = period)

Non-preemptive scheduling:

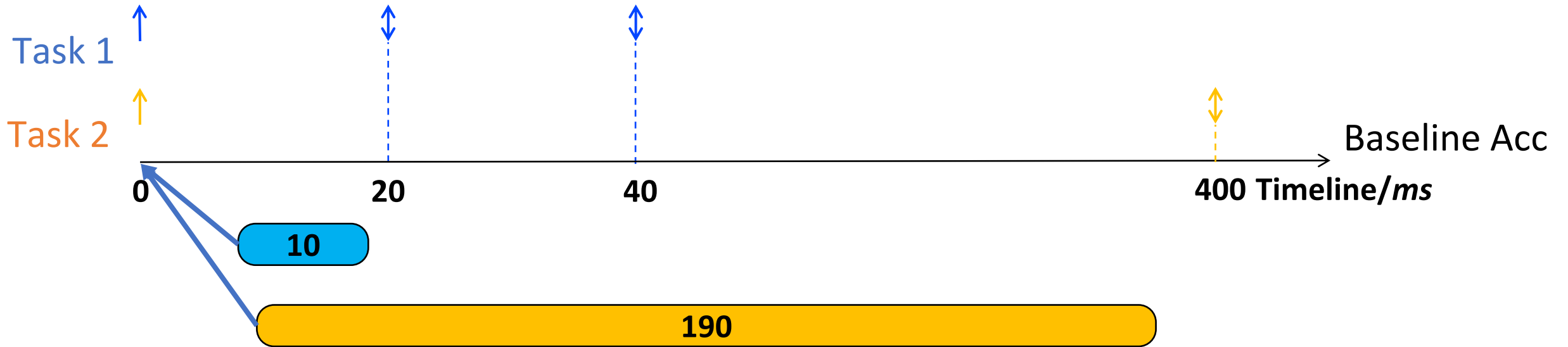


Motivation Example: 2 Tasks, 1 Single Acc

Task 1 : 10 ms execution time, 20 ms period

Task 2: 190 ms execution time, 400 ms period (assuming ddl = period)

Non-preemptive scheduling:

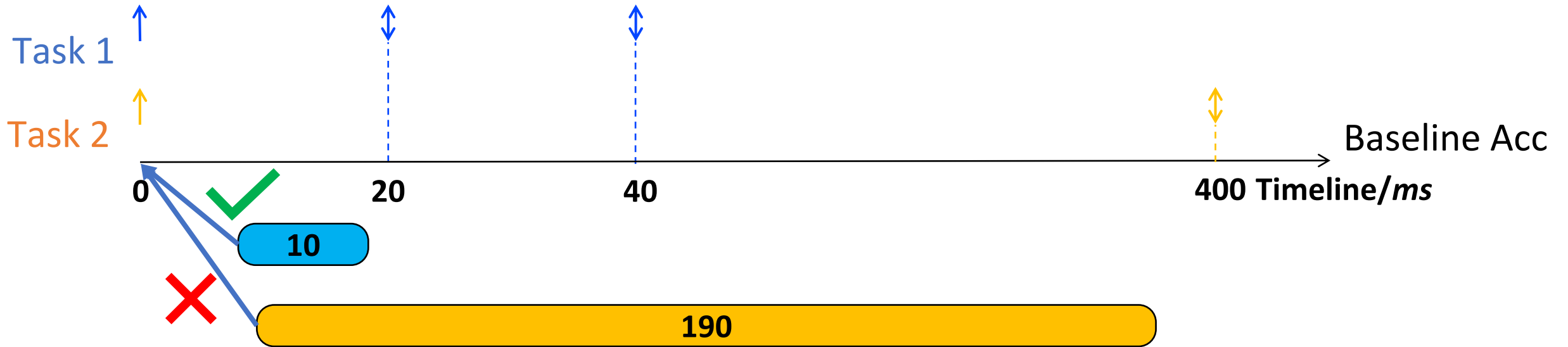


Motivation Example: 2 Tasks, 1 Single Acc

Task 1 : 10 ms execution time, 20 ms period

Task 2: 190 ms execution time, 400 ms period (assuming ddl = period)

Non-preemptive scheduling:

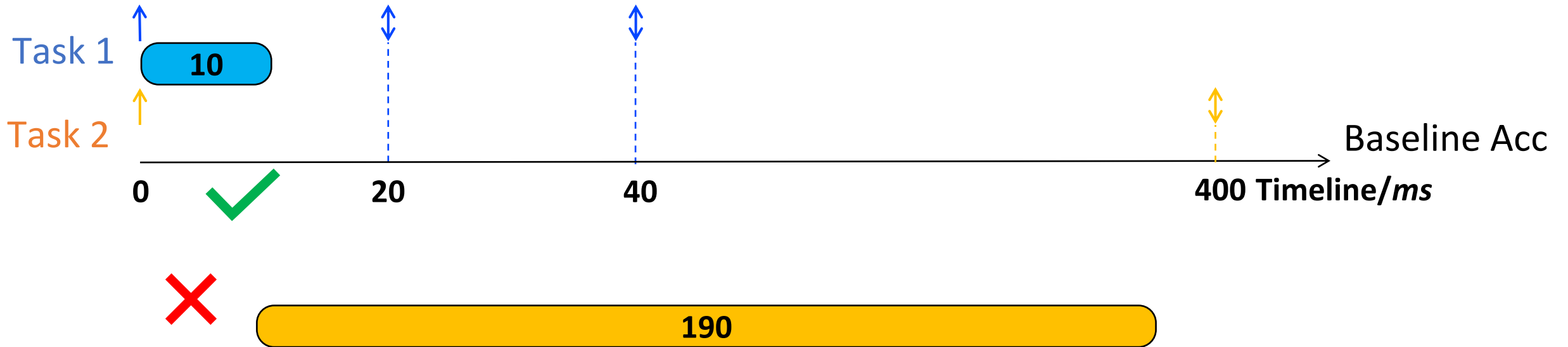


Motivation Example: 2 Tasks, 1 Single Acc

Task 1 : 10 ms execution time, 20 ms period

Task 2: 190 ms execution time, 400 ms period (assuming ddl = period)

Non-preemptive scheduling:

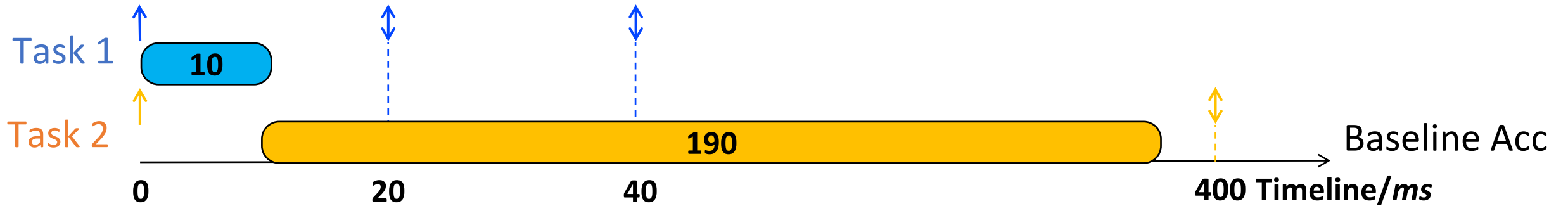


Motivation Example: 2 Tasks, 1 Single Acc

Task 1 : 10 ms execution time, 20 ms period

Task 2: 190 ms execution time, 400 ms period (assuming ddl = period)

Non-preemptive scheduling:



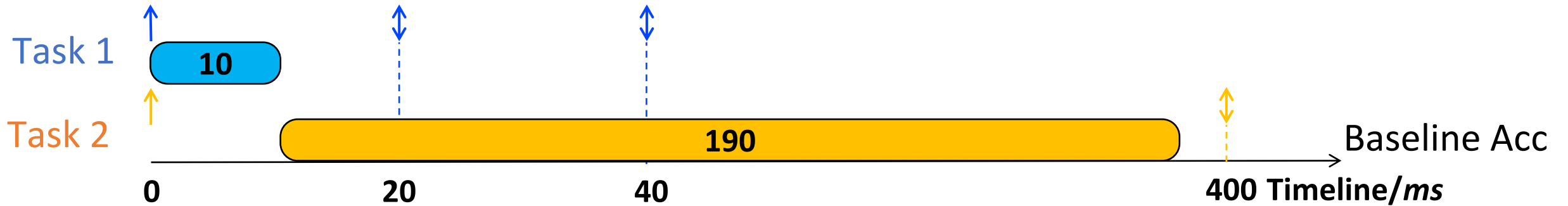
T=0: execute task 1 job 1 first , postpone task 2 job 1

Motivation Example: 2 Tasks, 1 Single Acc

Task 1: 10 ms execution time, 20 ms period

Task 2: 190 ms execution time, 400 ms period (assuming ddl = period)

Non-preemptive scheduling:

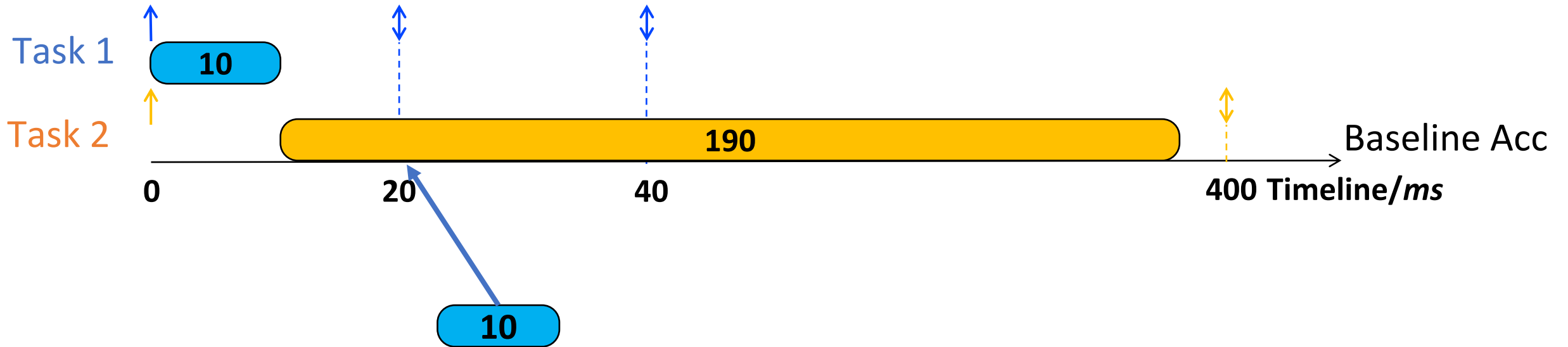


Motivation Example: 2 Tasks, 1 Single Acc

Task 1: 10 ms execution time, 20 ms period

Task 2: 190 ms execution time, 400 ms period (assuming ddl = period)

Non-preemptive scheduling:

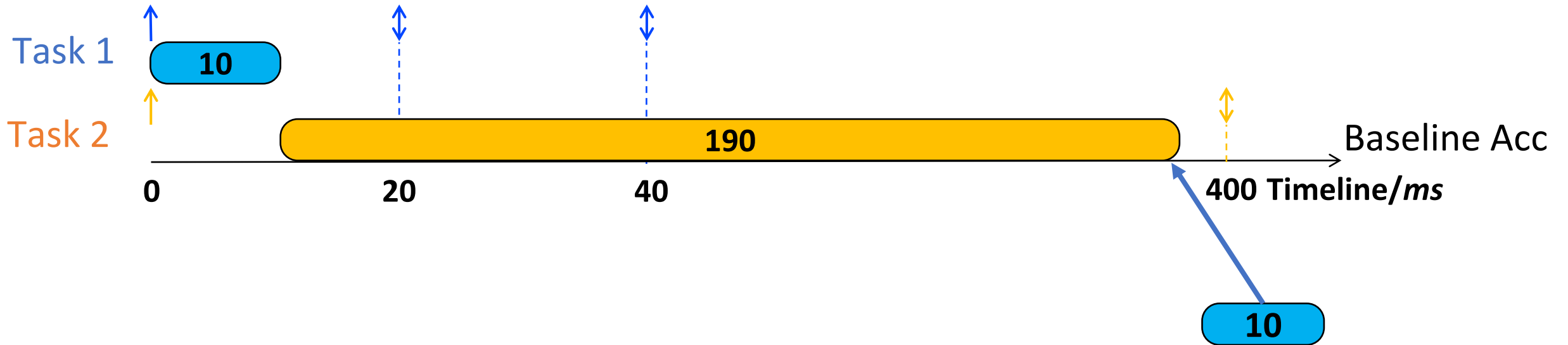


Motivation Example: 2 Tasks, 1 Single Acc

Task 1: 10 ms execution time, 20 ms period

Task 2: 190 ms execution time, 400 ms period (assuming ddl = period)

Non-preemptive scheduling:

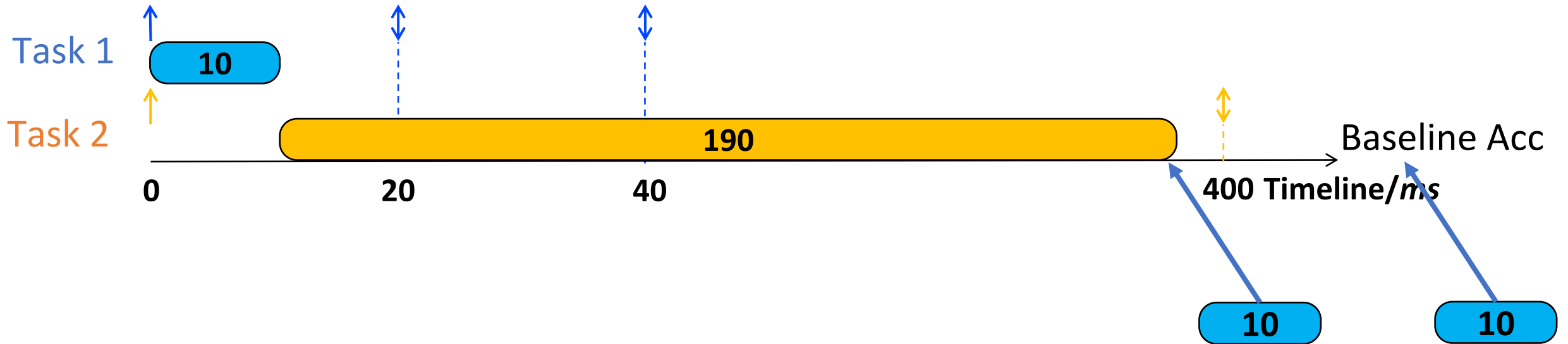


Motivation Example: 2 Tasks, 1 Single Acc

Task 1: 10 ms execution time, 20 ms period

Task 2: 190 ms execution time, 400 ms period (assuming ddl = period)

Non-preemptive scheduling:

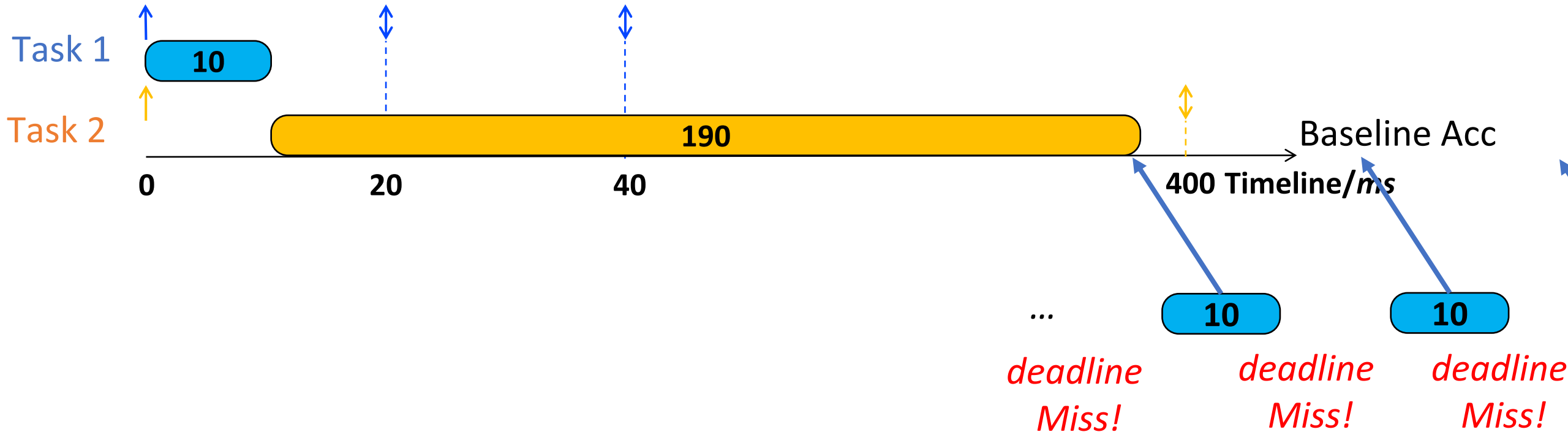


Motivation Example: 2 Tasks, 1 Single Acc

Task 1: 10 ms execution time, 20 ms period

Task 2: 190 ms execution time, 400 ms period (assuming ddl = period)

Non-preemptive scheduling:

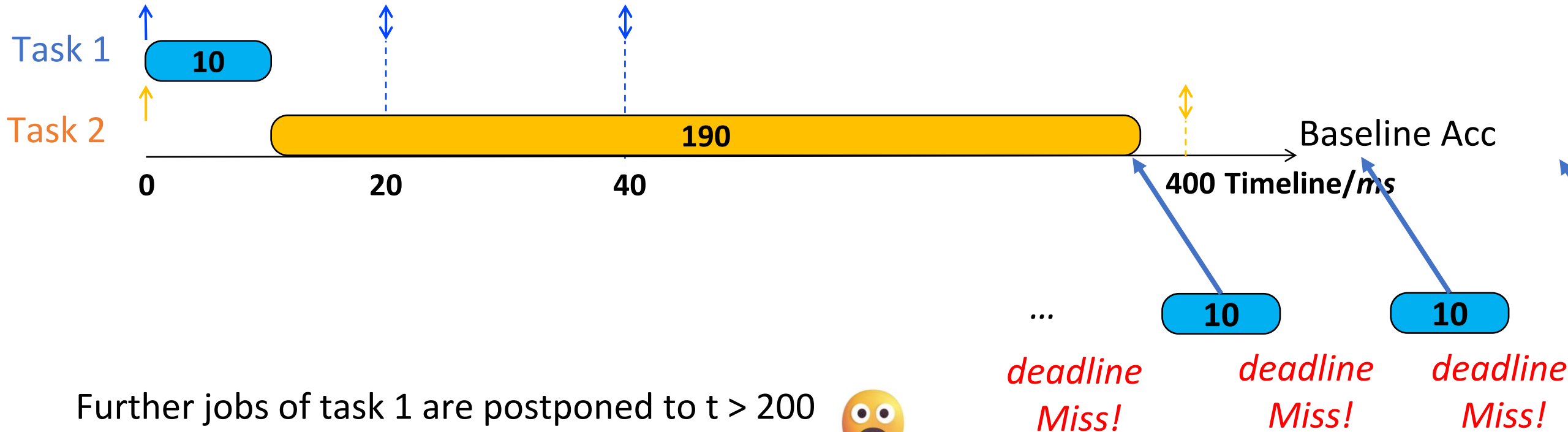


Motivation Example: 2 Tasks, 1 Single Acc

Task 1: 10 ms execution time, 20 ms period

Task 2: 190 ms execution time, 400 ms period (assuming ddl = period)

Non-preemptive scheduling:



Further jobs of task 1 are postponed to $t > 200$
System misses the deadline --> system fails



Motivation Example: 2 Tasks on 1 Single Acc: Will More Speedup Help?



- optimize the latency --> **5x** faster



Motivation Example: 2 Tasks on 1 Single Acc: Will More Speedup Help?



- optimize the latency --> **5x** faster ➡ **Still miss deadline!**



Motivation Example: 2 Tasks on 1 Single Acc: Will More Speedup Help?



- optimize the latency --> **5x** faster ➡ **Still miss deadline!**



Motivation Example: 2 Tasks on 1 Single Acc: Will More Speedup Help?



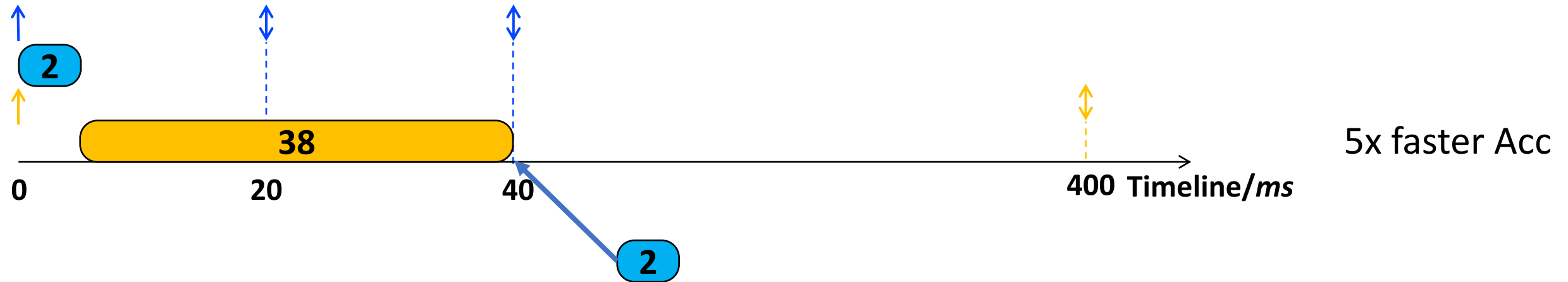
- optimize the latency --> **5x** faster ➡ **Still miss deadline!**



Motivation Example: 2 Tasks on 1 Single Acc: Will More Speedup Help?



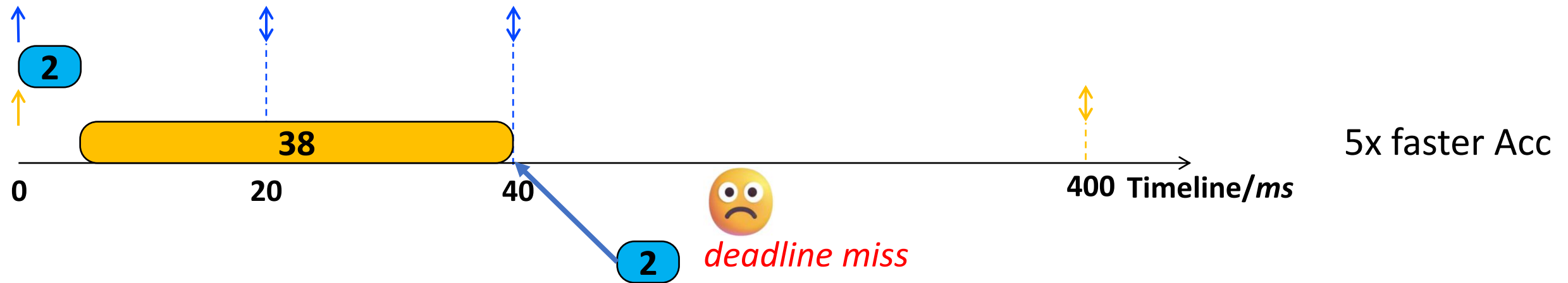
- optimize the latency --> **5x** faster ➡ **Still miss deadline!**



Motivation Example: 2 Tasks on 1 Single Acc: Will More Speedup Help?



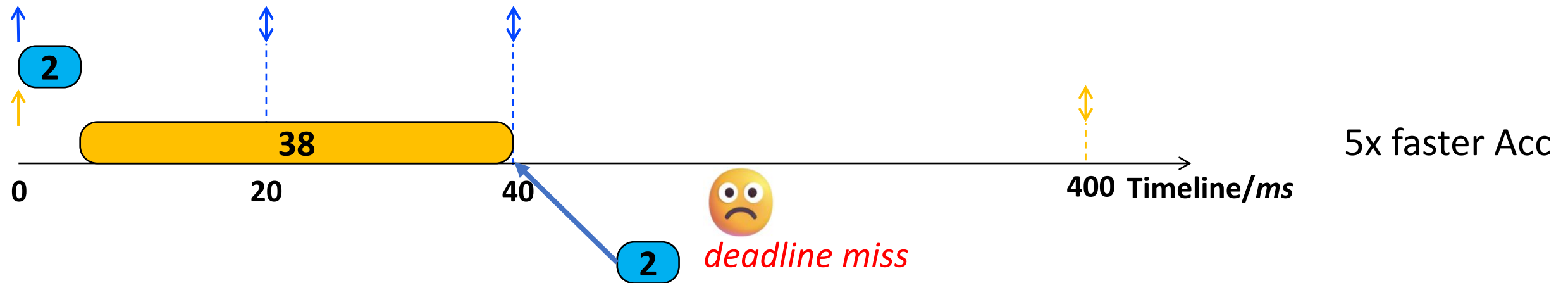
- optimize the latency --> **5x** faster ➡ **Still miss deadline!**



Motivation Example: 2 Tasks on 1 Single Acc: Will More Speedup Help?



- optimize the latency --> **5x** faster ➡ **Still miss deadline!**



- Endless latency acceleration is impractical
- more feasible solution needed

Motivation Example: 2 Tasks on 1 Single Acc: What if We Have Preemption?

- Assuming if we can **swap out** the ongoing tasks and **resume** its execution later



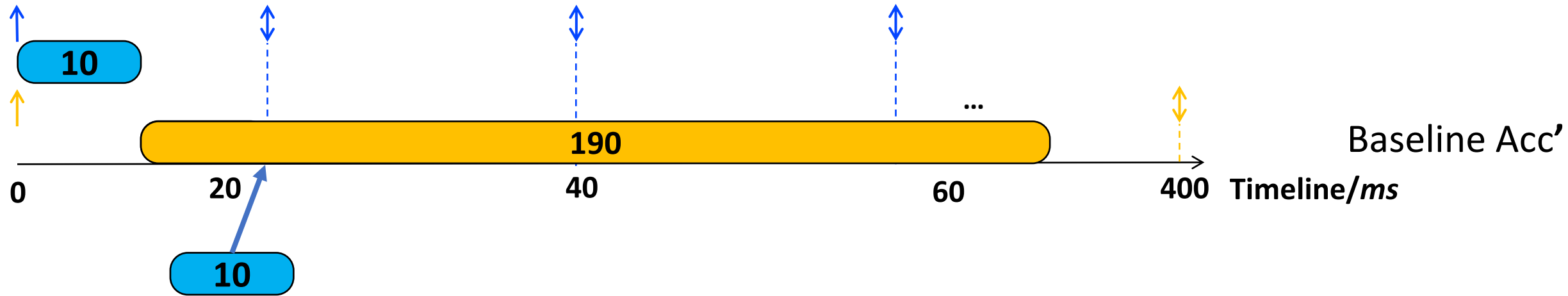
Motivation Example: 2 Tasks on 1 Single Acc: What if We Have Preemption?

- Assuming if we can **swap out** the ongoing tasks and **resume** its execution later



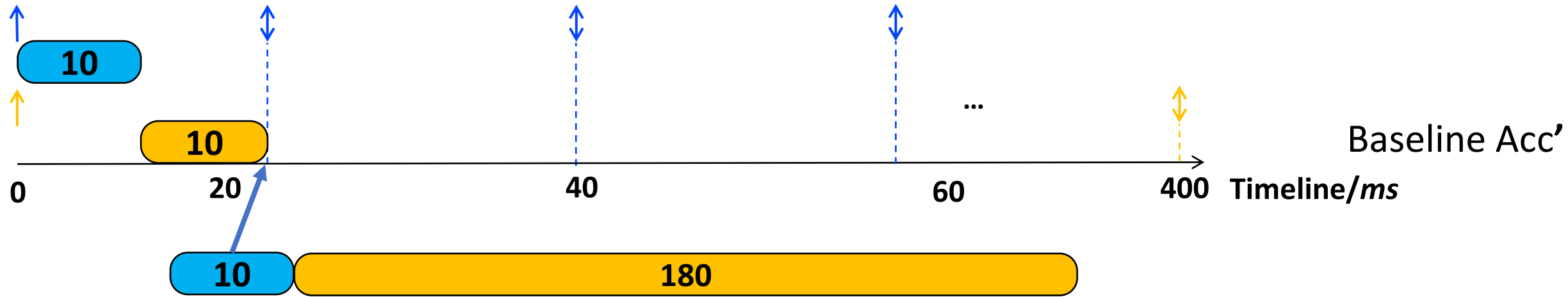
Motivation Example: 2 Tasks on 1 Single Acc: What if We Have Preemption?

- Assuming if we can **swap out** the ongoing tasks and **resume** its execution later



Motivation Example: 2 Tasks on 1 Single Acc: What if We Have Preemption?

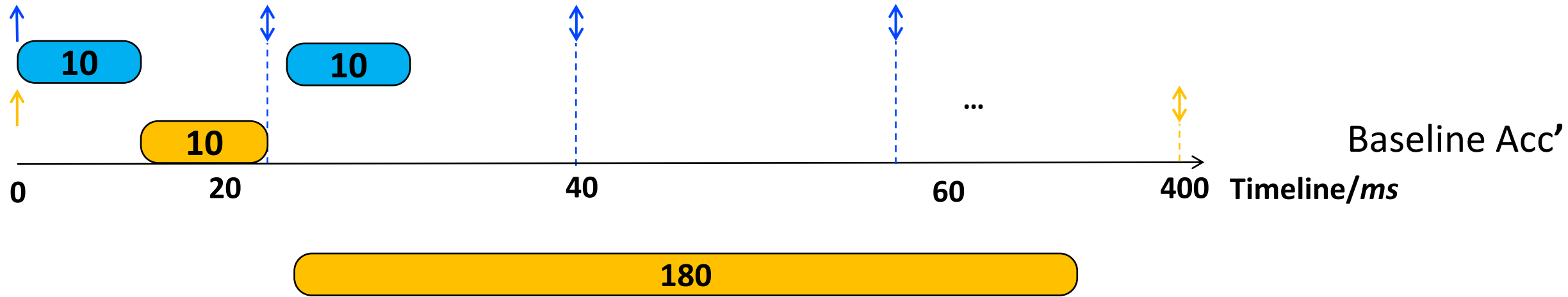
- Assuming if we can **swap out** the ongoing tasks and **resume** its execution later



- T=20: **swap out** the task 2 job 1

Motivation Example: 2 Tasks on 1 Single Acc: What if We Have Preemption?

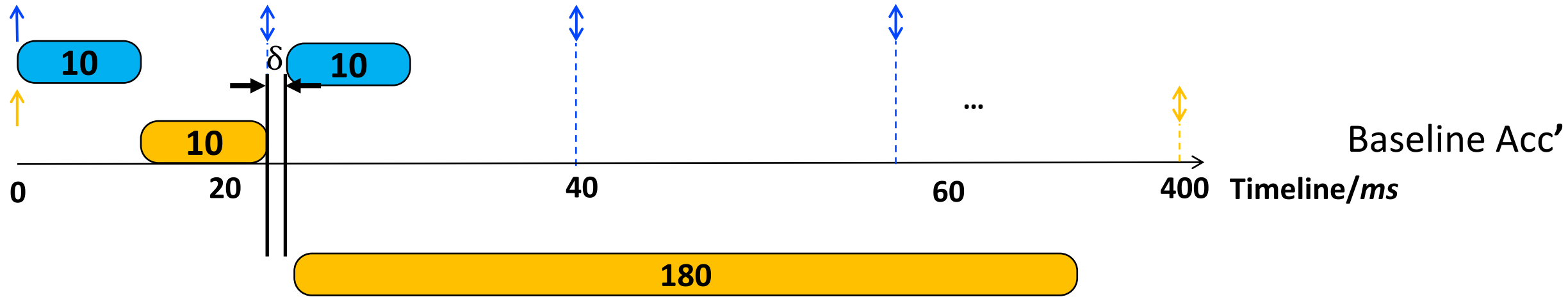
- Assuming if we can **swap out** the ongoing tasks and **resume** its execution later



- T=20: **swap out** the task 2 job 1

Motivation Example: 2 Tasks on 1 Single Acc: What if We Have Preemption?

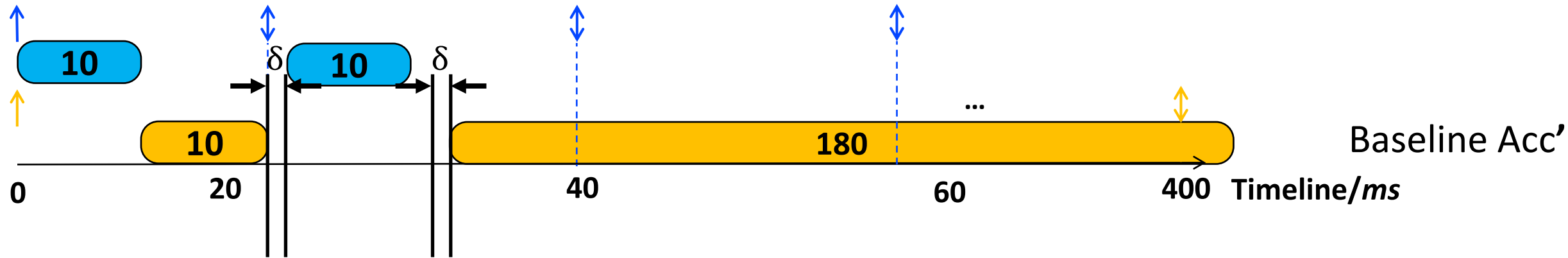
- Assuming if we can **swap out** the ongoing tasks and **resume** its execution later



- $T=20$: **swap out** the task 2 job 1

Motivation Example: 2 Tasks on 1 Single Acc: What if We Have Preemption?

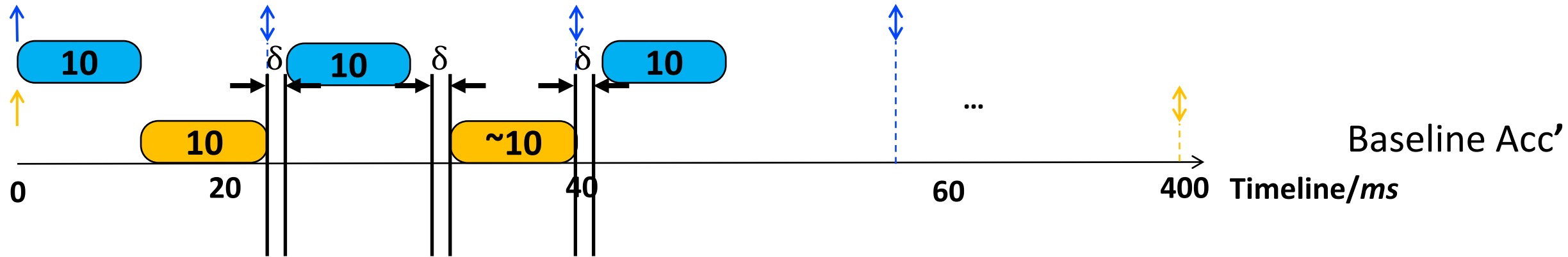
- Assuming if we can **swap out** the ongoing tasks and **resume** its execution later



- T=20: **swap out** the task 2 job 1
- T=30: **swap in** the task 2 job 1

Motivation Example: 2 Tasks on 1 Single Acc: What if We Have Preemption?

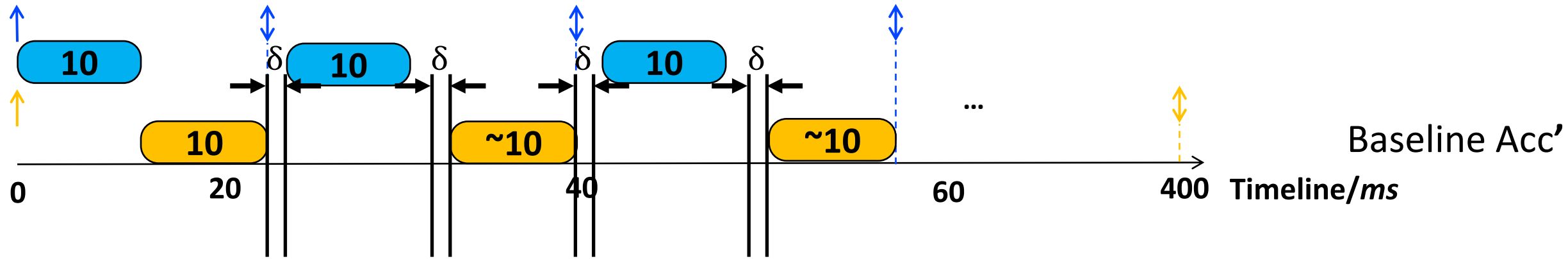
- Assuming if we can **swap out** the ongoing tasks and **resume** its execution later



- $T=20$: **swap out** the task 2 job 1
- $T=30$: **swap in** the task 2 job 1

Motivation Example: 2 Tasks on 1 Single Acc: What if We Have Preemption?

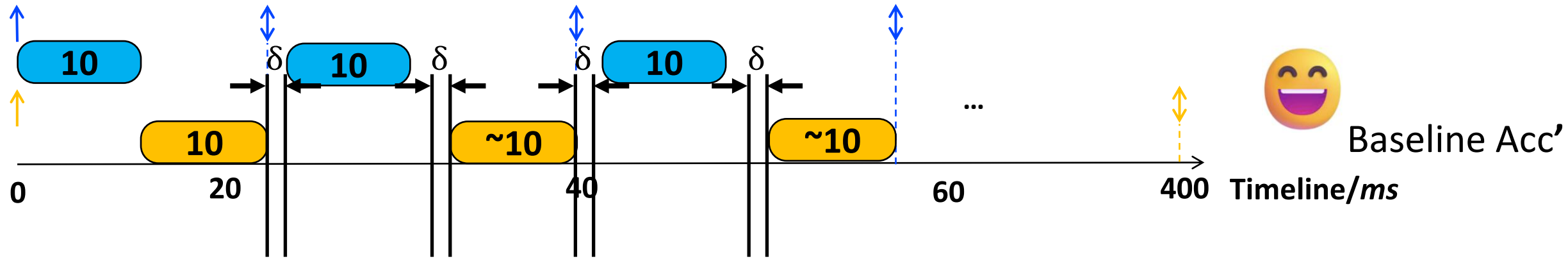
- Assuming if we can **swap out** the ongoing tasks and **resume** its execution later



- $T=20$: **swap out** the task 2 job 1
- $T=30$: **swap in** the task 2 job 1

Motivation Example: 2 Tasks on 1 Single Acc: What if We Have Preemption?

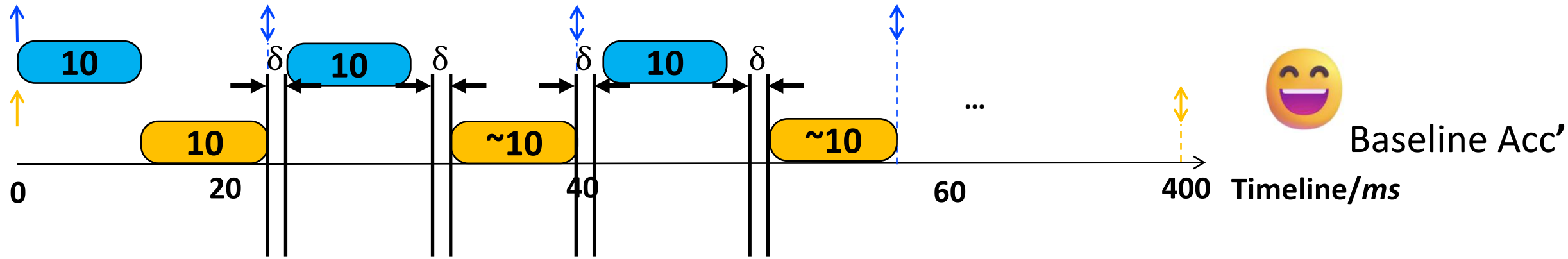
- Assuming if we can **swap out** the ongoing tasks and **resume** its execution later



- T=20: **swap out** the task 2 job 1
- T=30: **swap in** the task 2 job 1
- The tasks can continue running, and the system won't fail

Motivation Example: 2 Tasks on 1 Single Acc: What if We Have Preemption?

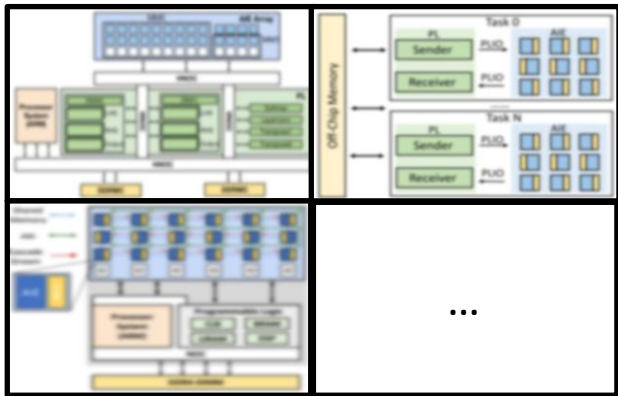
- Assuming if we can **swap out** the ongoing tasks and **resume** its execution later



- T=20: **swap out** the task 2 job 1
- T=30: **swap in** the task 2 job 1
- The tasks can continue running, and the system won't fail

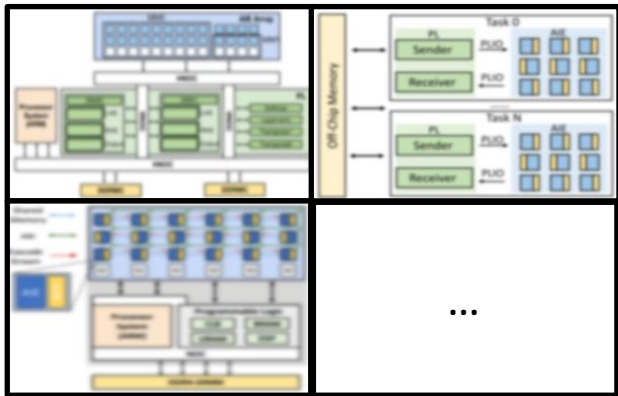
Preemption (swap out + swap in) is a viable solution!

How to Implement Preemption in Accelerator Hardware



Existing FPGA Accelerators

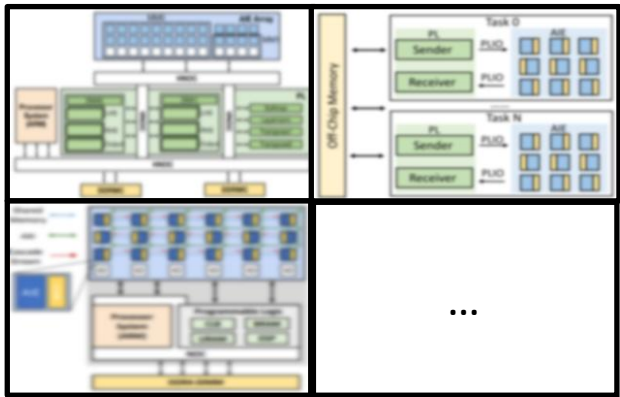
How to Implement Preemption in Accelerator Hardware



Low latency, high throughput ✓


Existing FPGA Accelerators

How to Implement Preemption in Accelerator Hardware

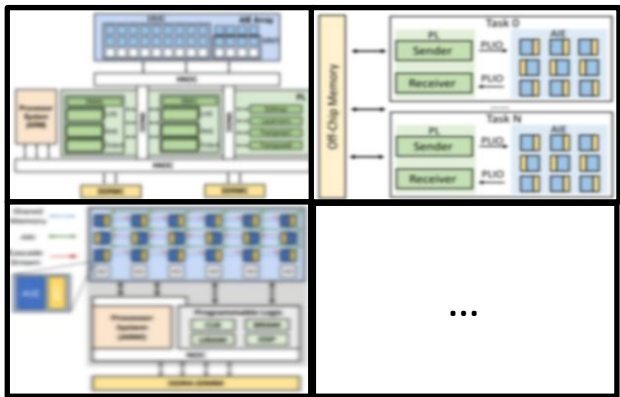


Existing FPGA Accelerators

➡ Low latency, high throughput ✓

➡ Can not be applied to real-time, safety-critical systems ❌ ➡ 

How to Implement Preemption in Accelerator Hardware



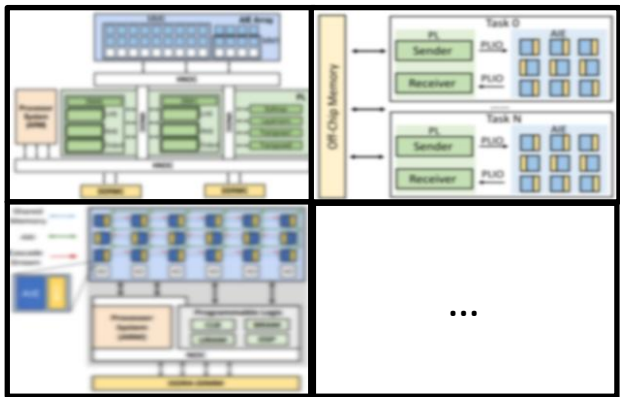
Existing FPGA Accelerators

➡ Low latency, high throughput ✓

➡ Can not be applied to real-time, safety-critical systems ❌ ➡ 

? How can we enable them to support real-time safety-critical systems

How to Implement Preemption in Accelerator Hardware



Existing FPGA Accelerators



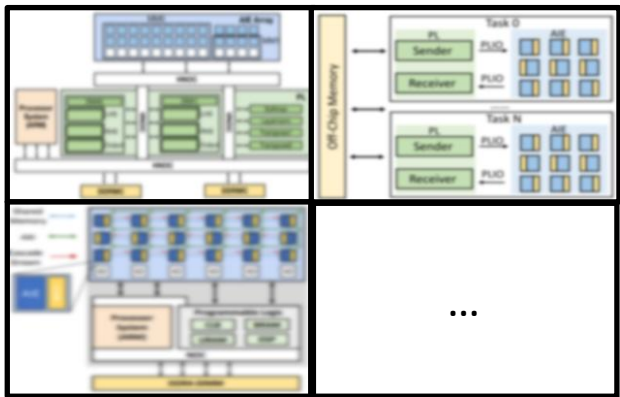
Fixed Acc
shape

➡ Low latency, high throughput ✓

➡ Can not be applied to real-time, safety-critical systems ❌ ➡ 

? How can we enable them to support real-time safety-critical systems

How to Implement Preemption in Accelerator Hardware




Existing FPGA Accelerators



Fixed Acc
shape Configs

➡ Low latency, high throughput ✓

➡ Can not be applied to real-time, safety-critical systems ❌ ➡ 

? How can we enable them to support real-time safety-critical systems

How to Implement Preemption in Accelerator Hardware

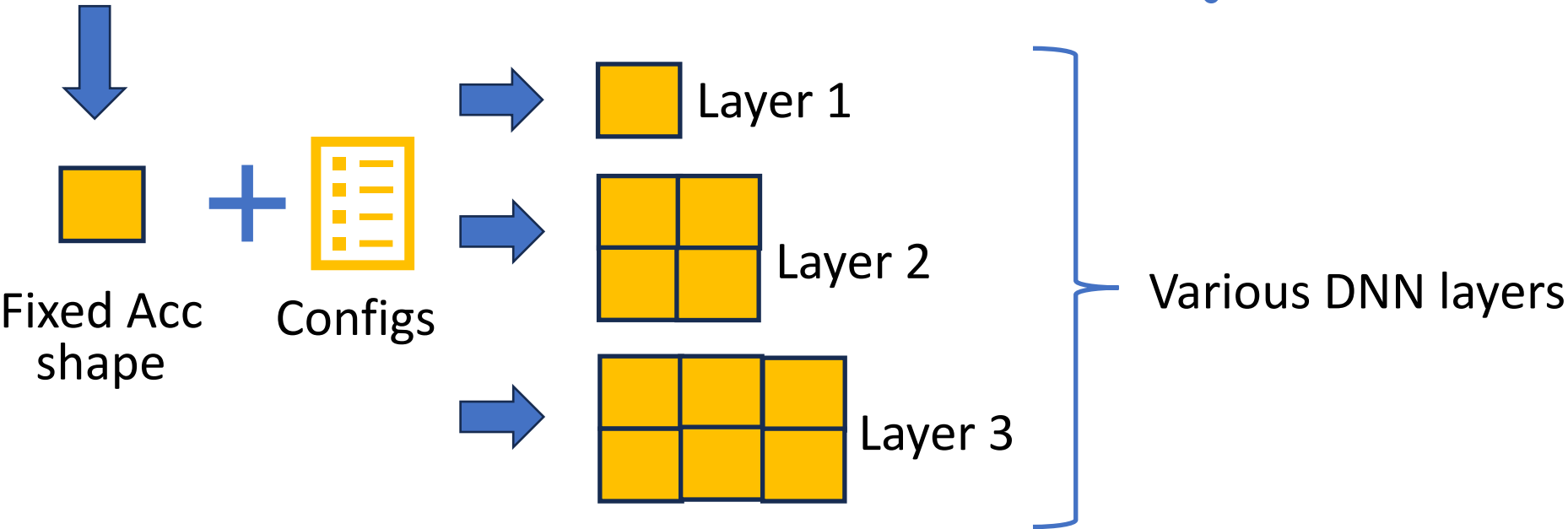


Existing FPGA Accelerators

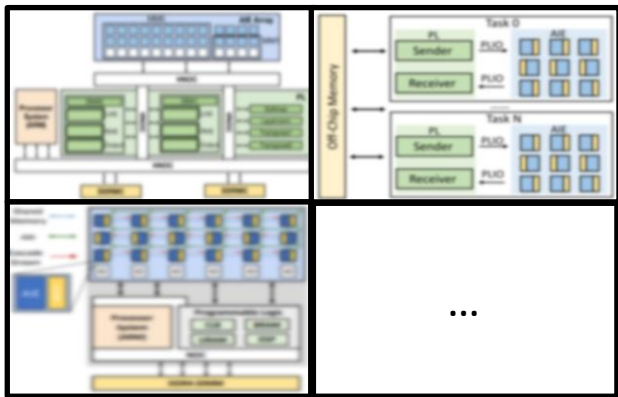
➡ Low latency, high throughput ✓

➡ Can not be applied to real-time, safety-critical systems ❌ ➡ 

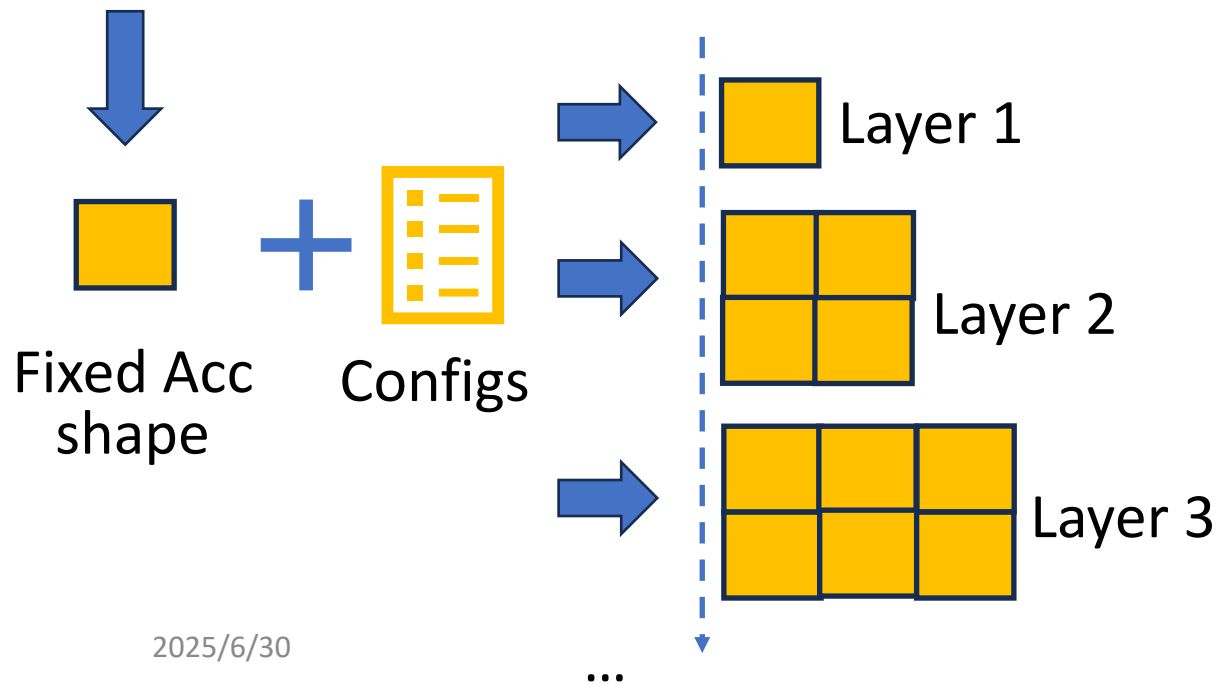
? How can we enable them to support real-time safety-critical systems



How to Implement Preemption in Accelerator Hardware



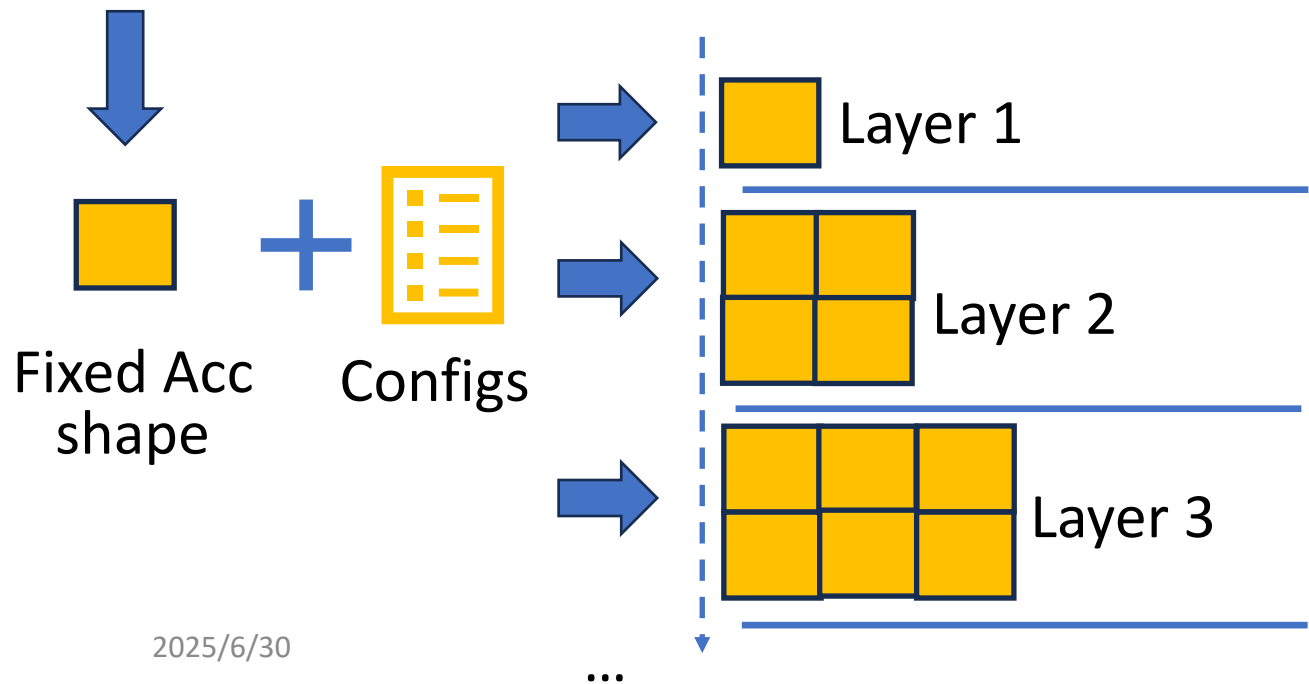
Existing FPGA Accelerators



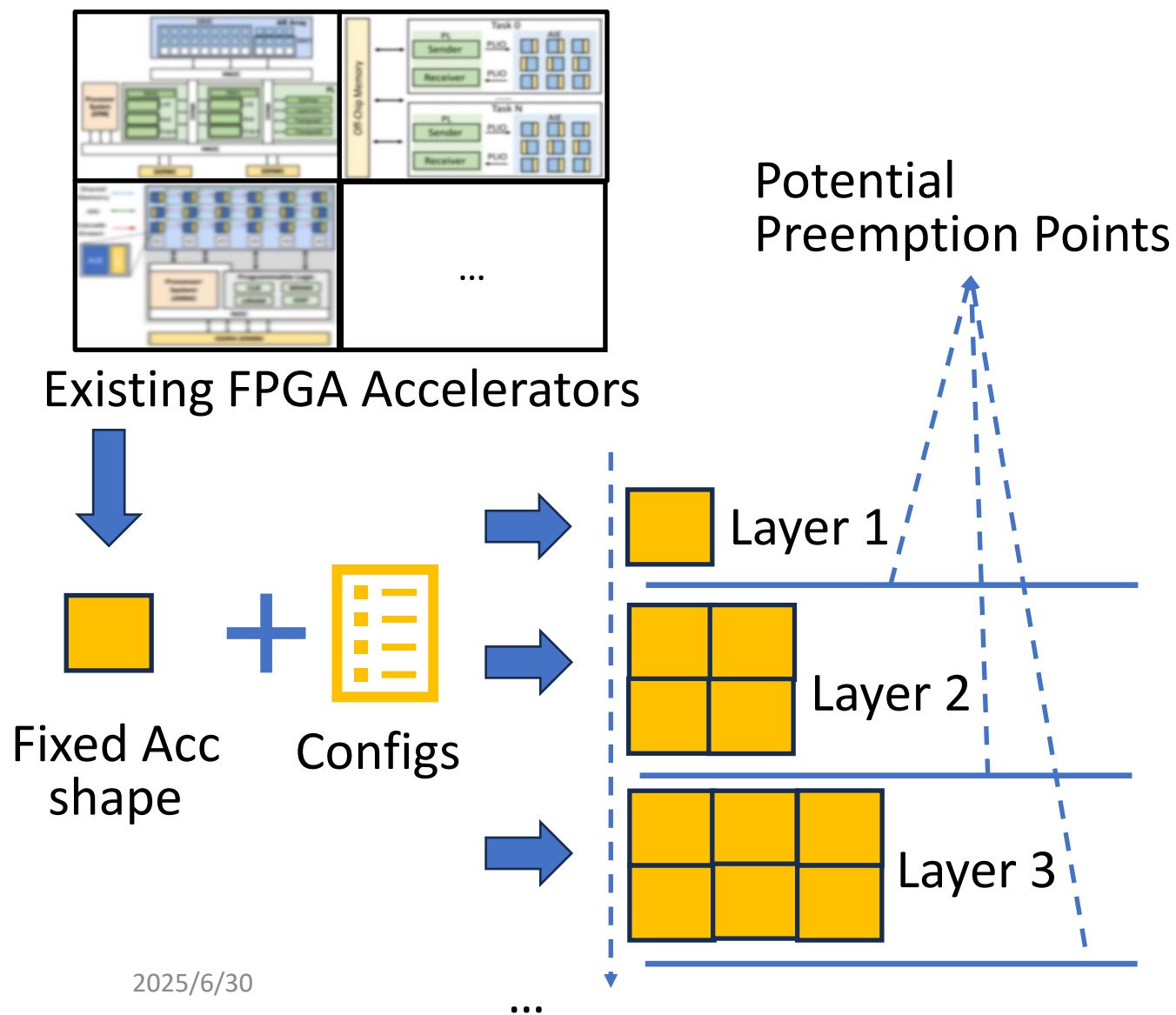
How to Implement Preemption in Accelerator Hardware



Existing FPGA Accelerators

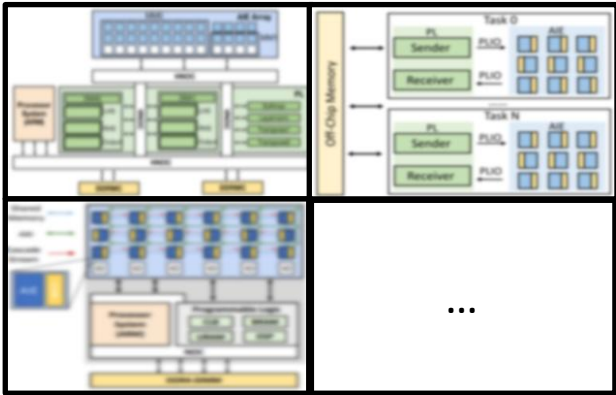


How to Implement Preemption in Accelerator Hardware

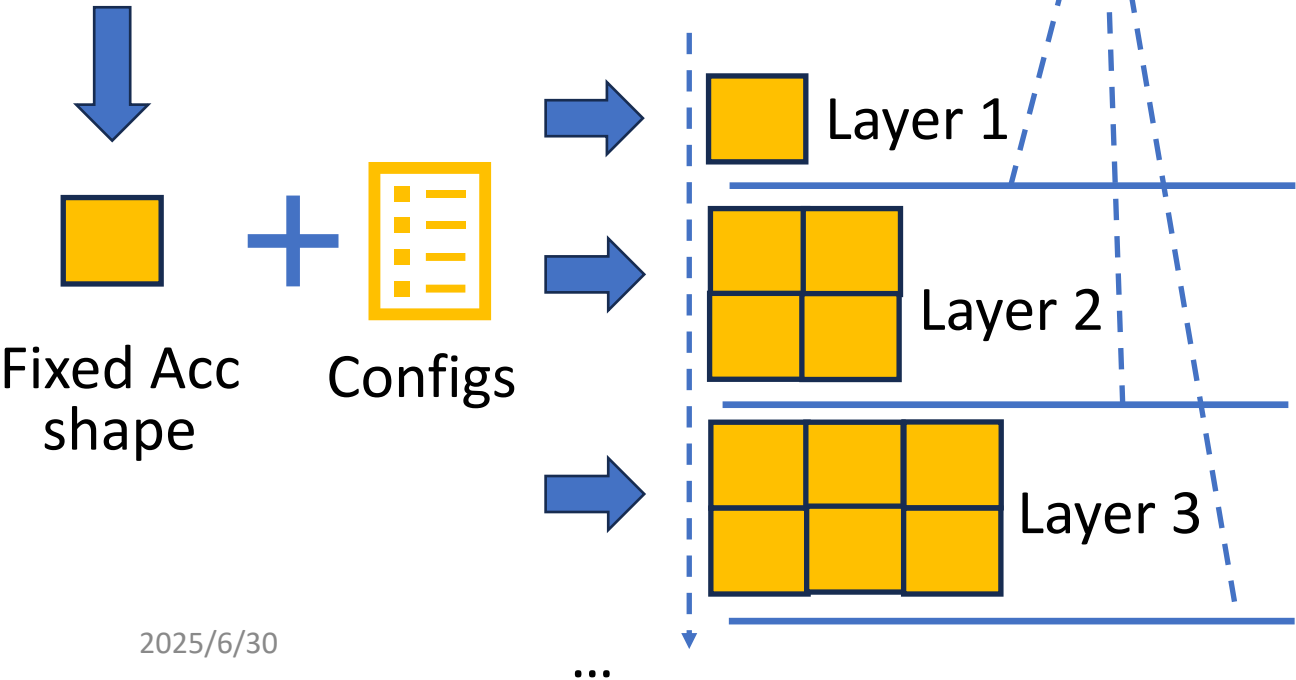


How to Implement Preemption in Accelerator Hardware

✓ Preemption ➡ Meet deadlines



Existing FPGA Accelerators

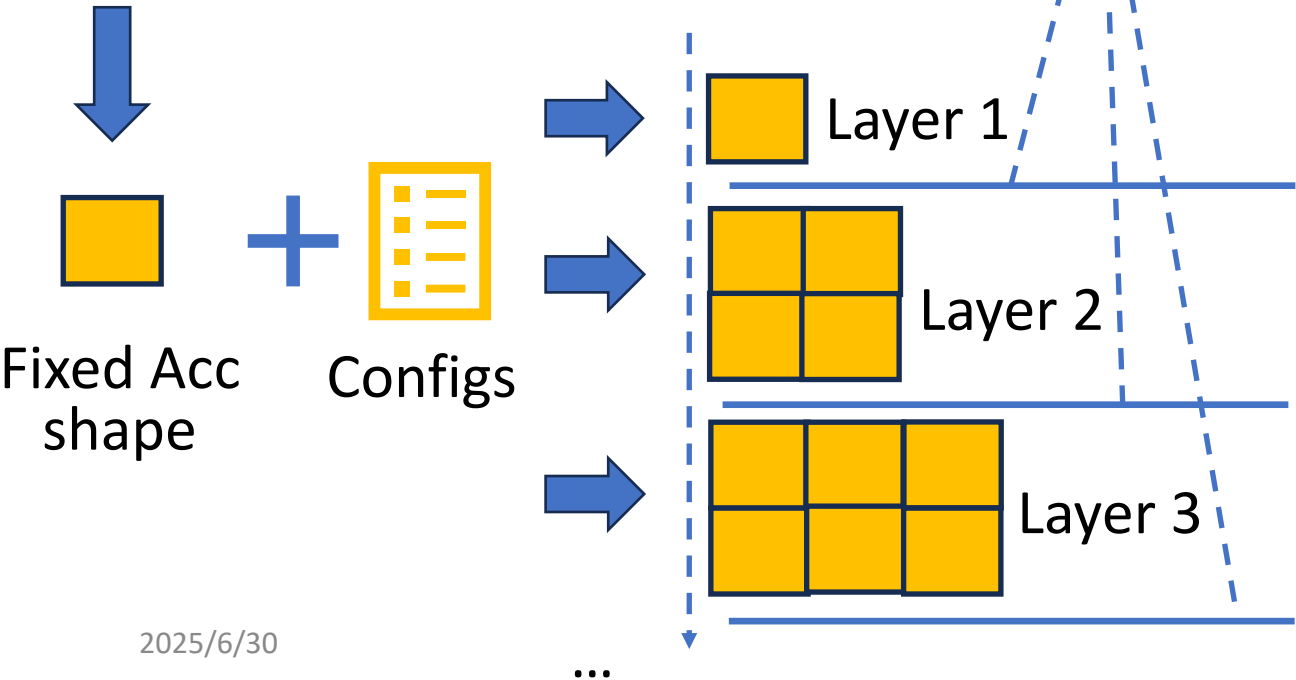


How to Implement Preemption in Accelerator Hardware

- ✓ Preemption → Meet deadlines
- ✓ Minimum invasive HW changes



Existing FPGA Accelerators

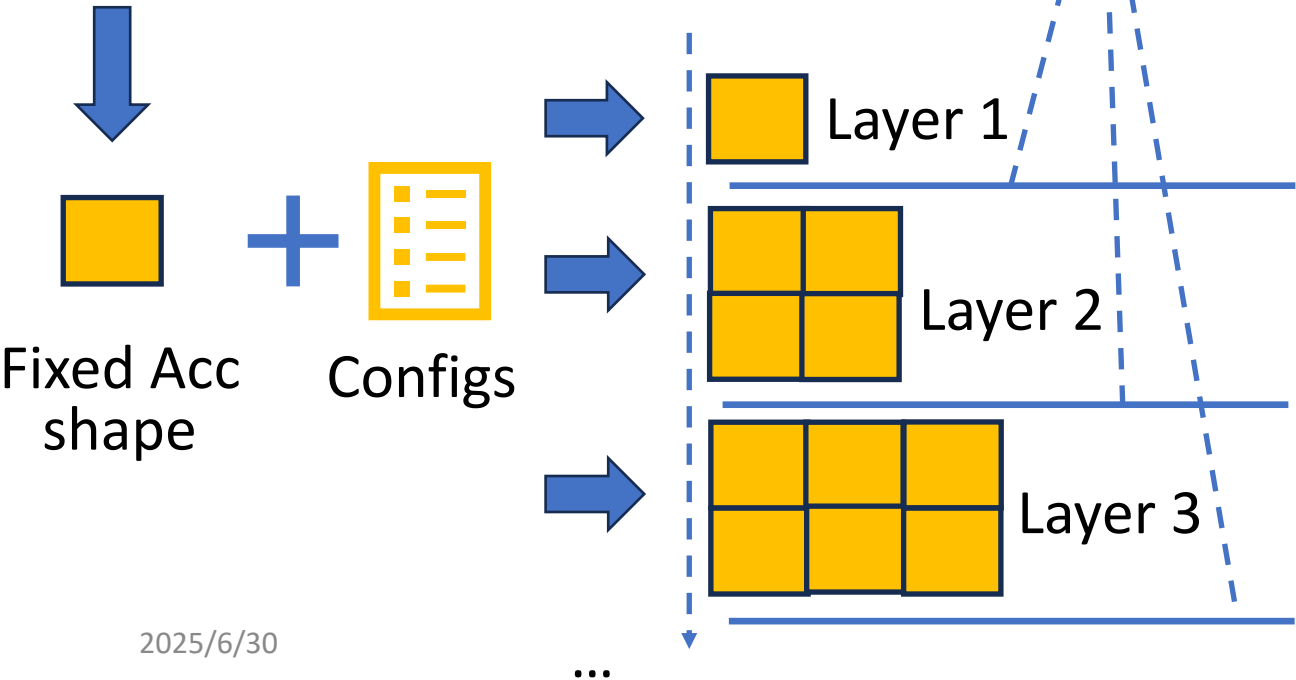


Potential
Preemption Points

How to Implement Preemption in Accelerator Hardware

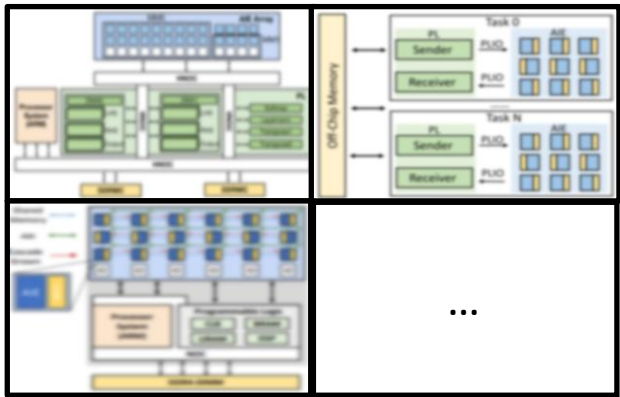


Existing FPGA Accelerators

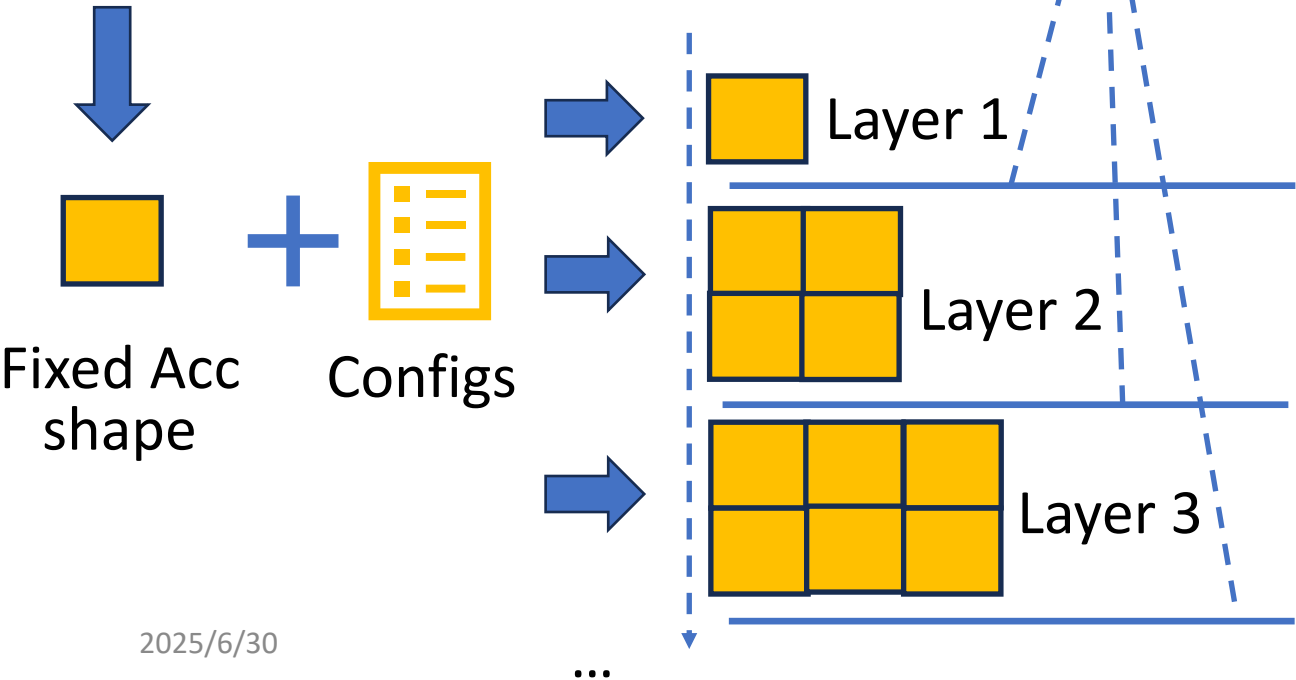


- ✓ Preemption ➡ Meet deadlines
- ✓ Minimum invasive HW changes
- ⚠ Have preemption overheads

How to Implement Preemption in Accelerator Hardware

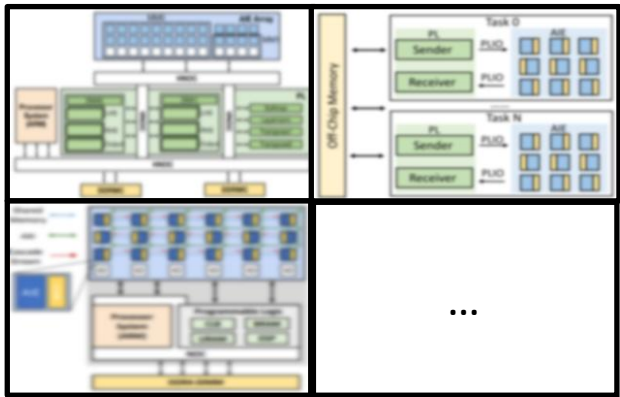


Existing FPGA Accelerators

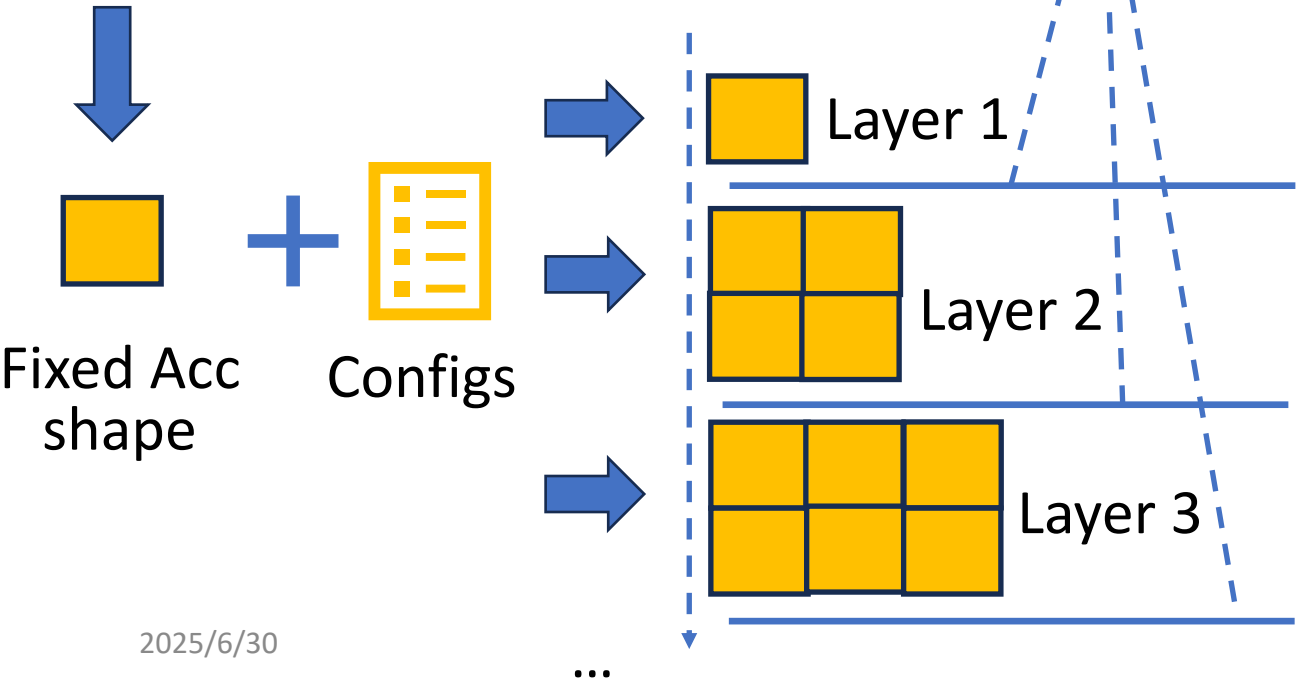


- ✓ Preemption → Meet deadlines
- ✓ Minimum invasive HW changes
- ⚠ Have preemption overheads
 - Low overhead achieved

How to Implement Preemption in Accelerator Hardware



Existing FPGA Accelerators



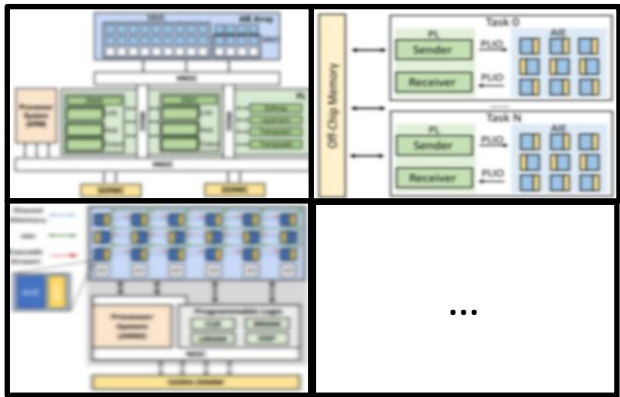
✓ Preemption → Meet deadlines

✓ Minimum invasive HW changes

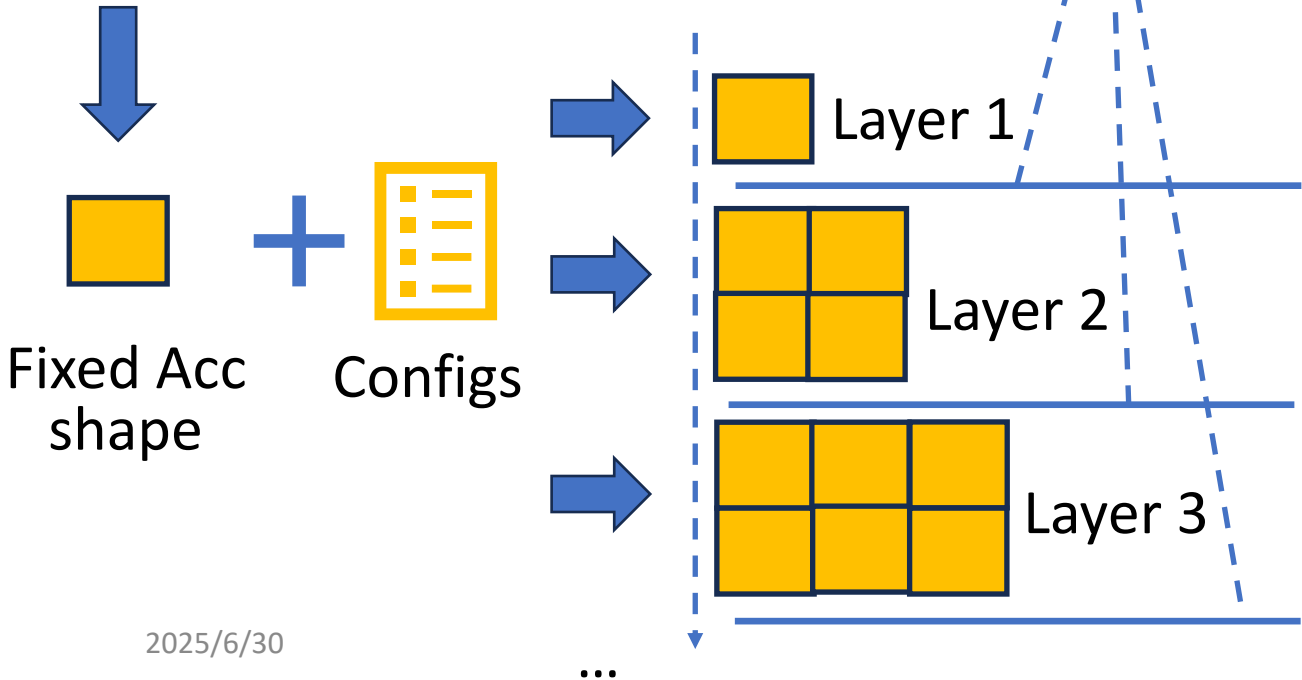
⚠ Have preemption overheads

- Low overhead achieved
- HLS Changes provided

How to Implement Preemption in Accelerator Hardware



Existing FPGA Accelerators

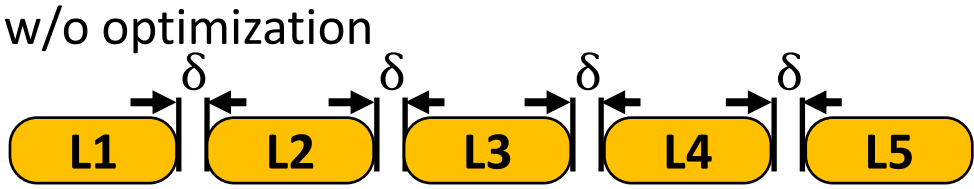


✓ Preemption → Meet deadlines

✓ Minimum invasive HW changes

⚠ Have preemption overheads

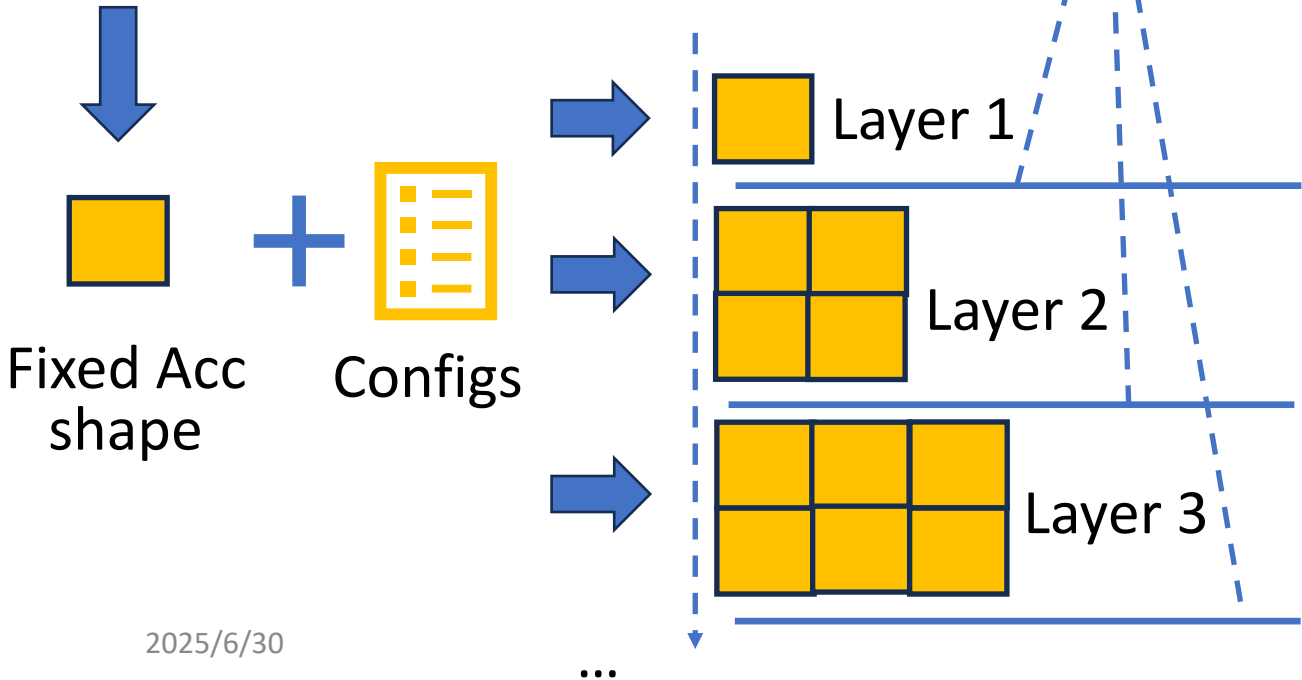
- Low overhead achieved
- HLS Changes provided



How to Implement Preemption in Accelerator Hardware



Existing FPGA Accelerators

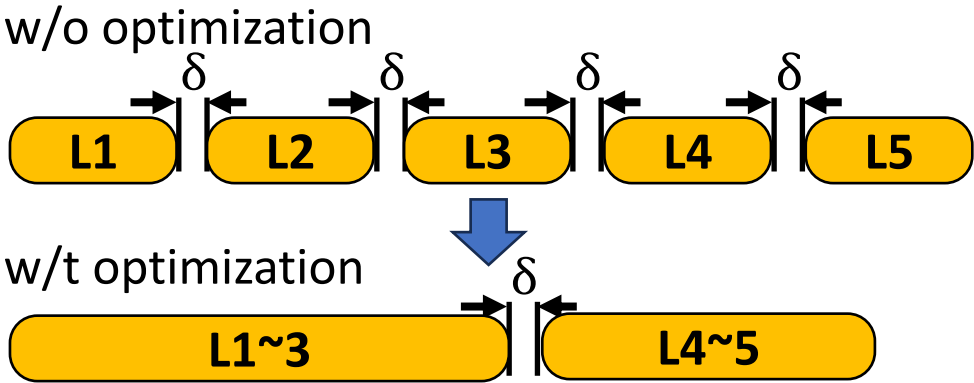


✓ Preemption → Meet deadlines

✓ Minimum invasive HW changes

⚠ Have preemption overheads

- Low overhead achieved
- HLS Changes provided



Challenges, Design Requirements and Contributions

Challenges, Design Requirements and Contributions

Existing accs can not
meet deadline

Challenges, Design Requirements and Contributions

Existing accs can not
meet deadline



Accelerator customization to
enable preemption

Challenges, Design Requirements and Contributions

Existing accs can not
meet deadline



Accelerator customization to
enable preemption

Dynamic Scheduling
Algorithm

Challenges, Design Requirements and Contributions

Existing accs can not
meet deadline



Accelerator customization to
enable preemption

Dynamic Scheduling
Algorithm

Preemption incurs
overhead

Challenges, Design Requirements and Contributions

Existing accs can not meet deadline



Accelerator customization to enable preemption




Dynamic Scheduling Algorithm



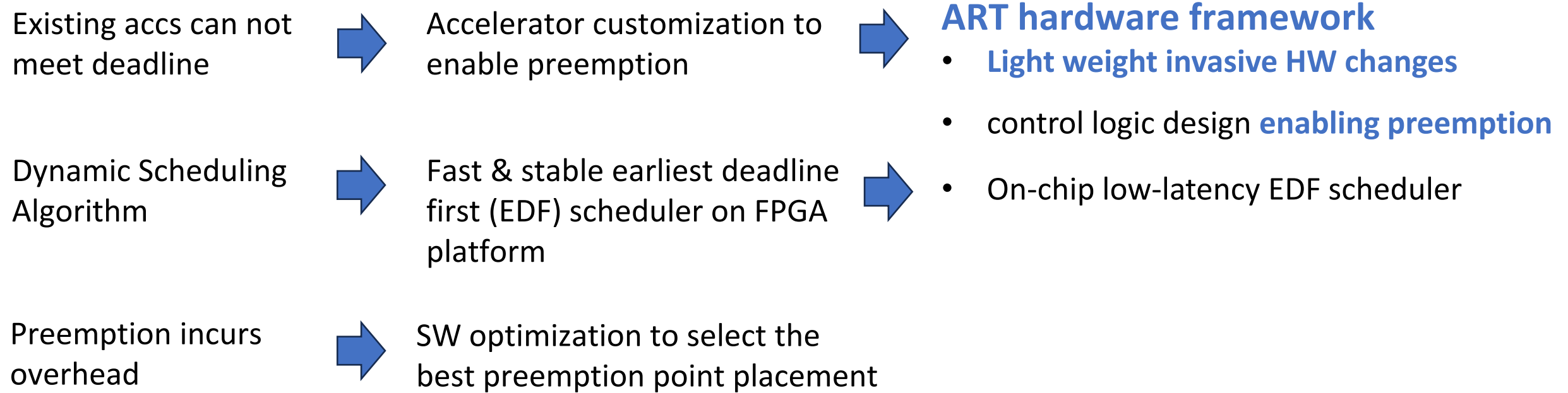
Fast & stable earliest deadline first (EDF) scheduler on FPGA platform

Preemption incurs overhead

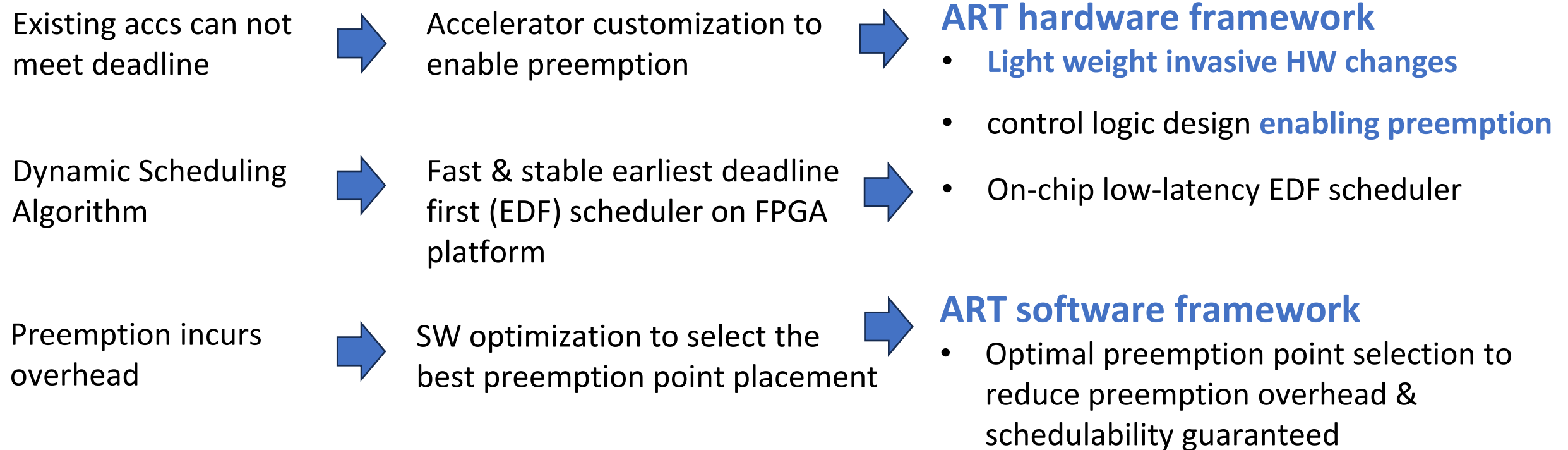
Challenges, Design Requirements and Contributions

Existing accs can not meet deadline		Accelerator customization to enable preemption
Dynamic Scheduling Algorithm		Fast & stable earliest deadline first (EDF) scheduler on FPGA platform
Preemption incurs overhead		SW optimization to select the best preemption point placement

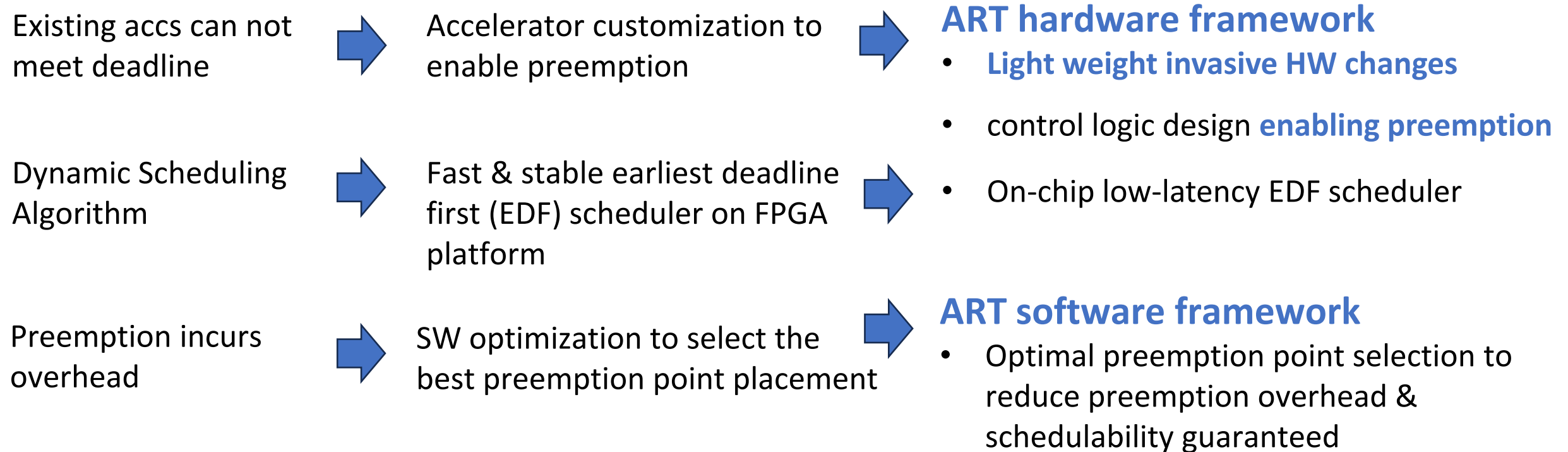
Challenges, Design Requirements and Contributions



Challenges, Design Requirements and Contributions



Challenges, Design Requirements and Contributions



ART enables existing FPGA HLS accelerators to be used in the real-time safety-critical scenario

ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features

ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features

Inputs



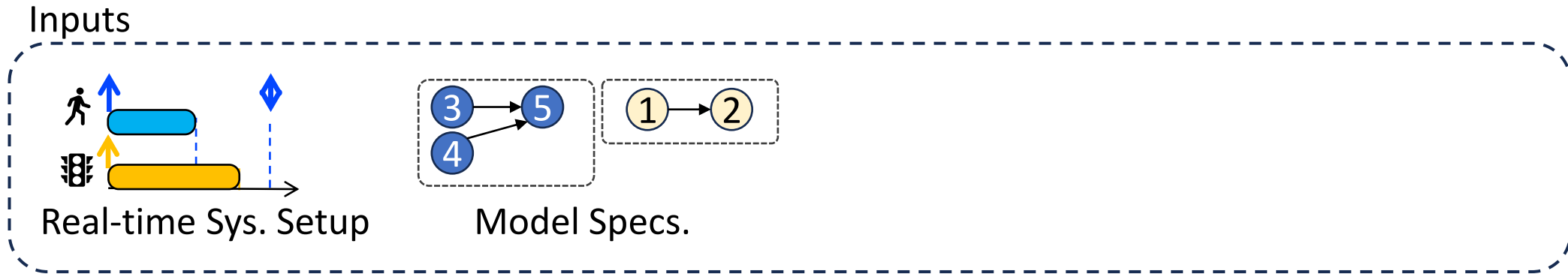
ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features



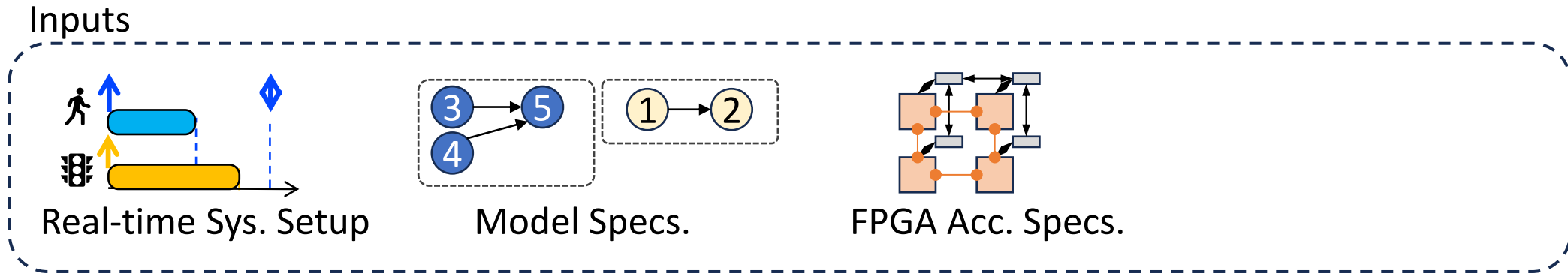
ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features



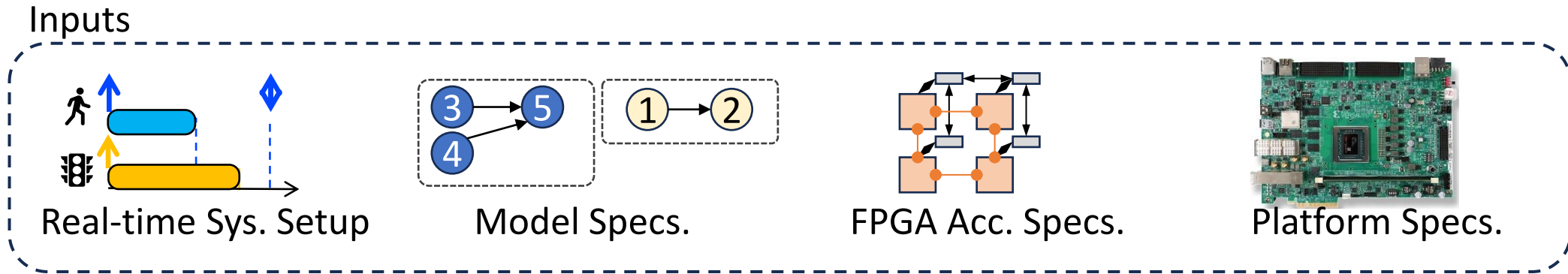
ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features



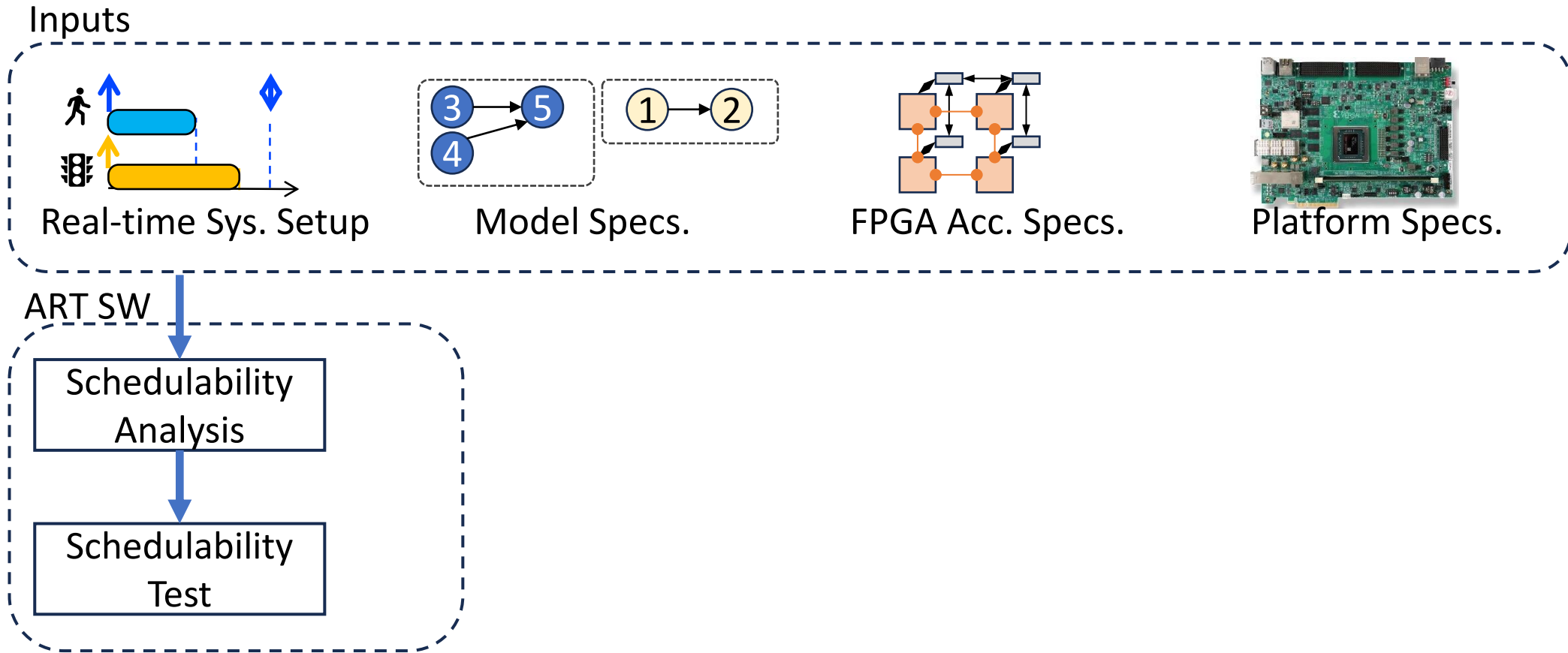
ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features



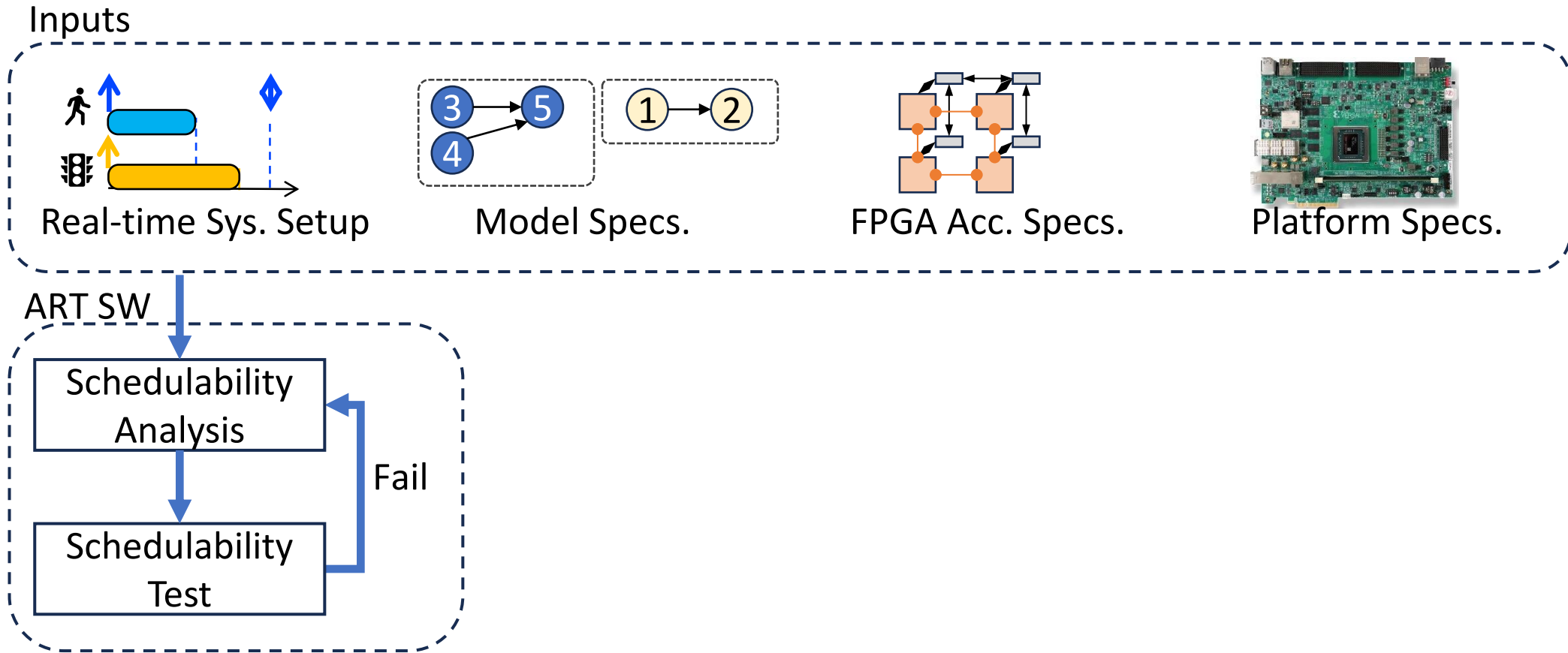
ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features



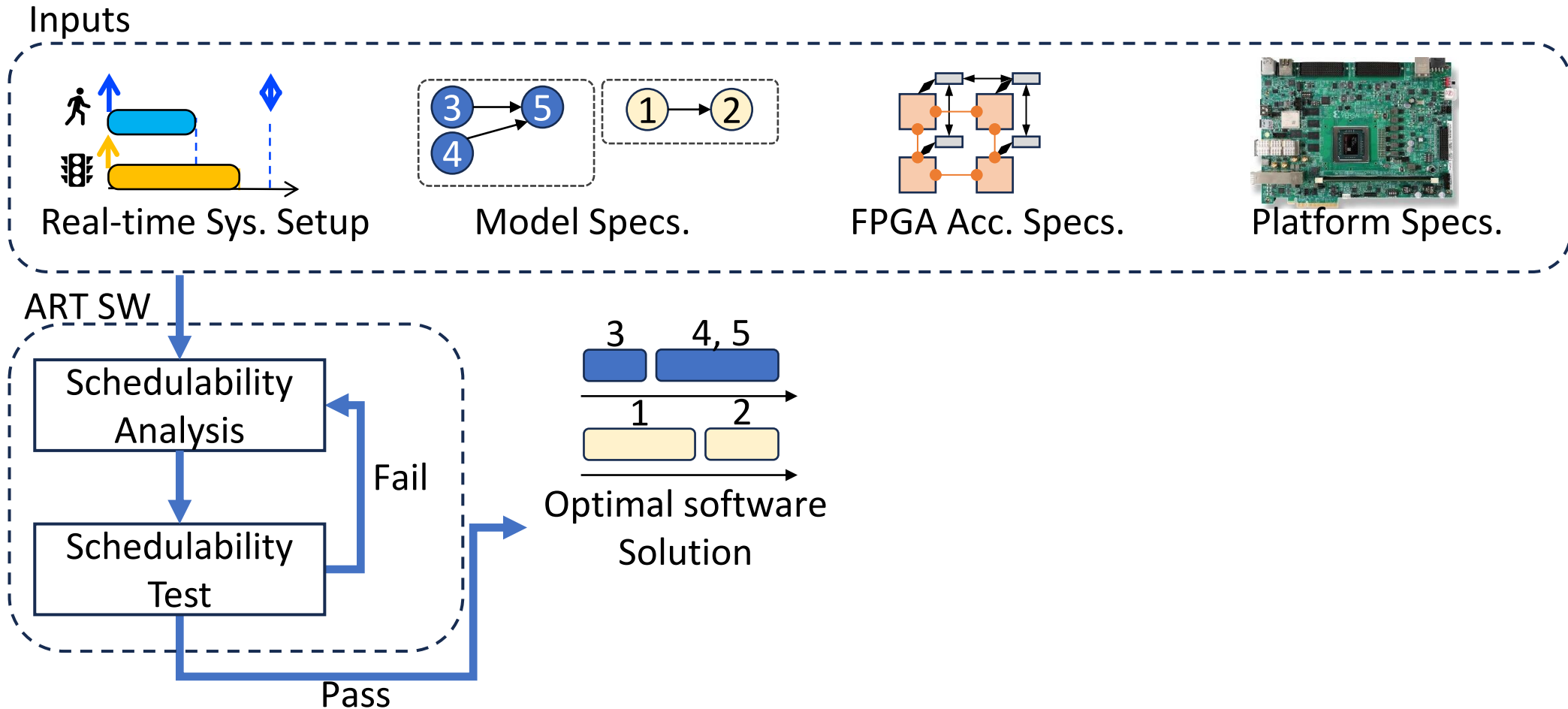
ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features



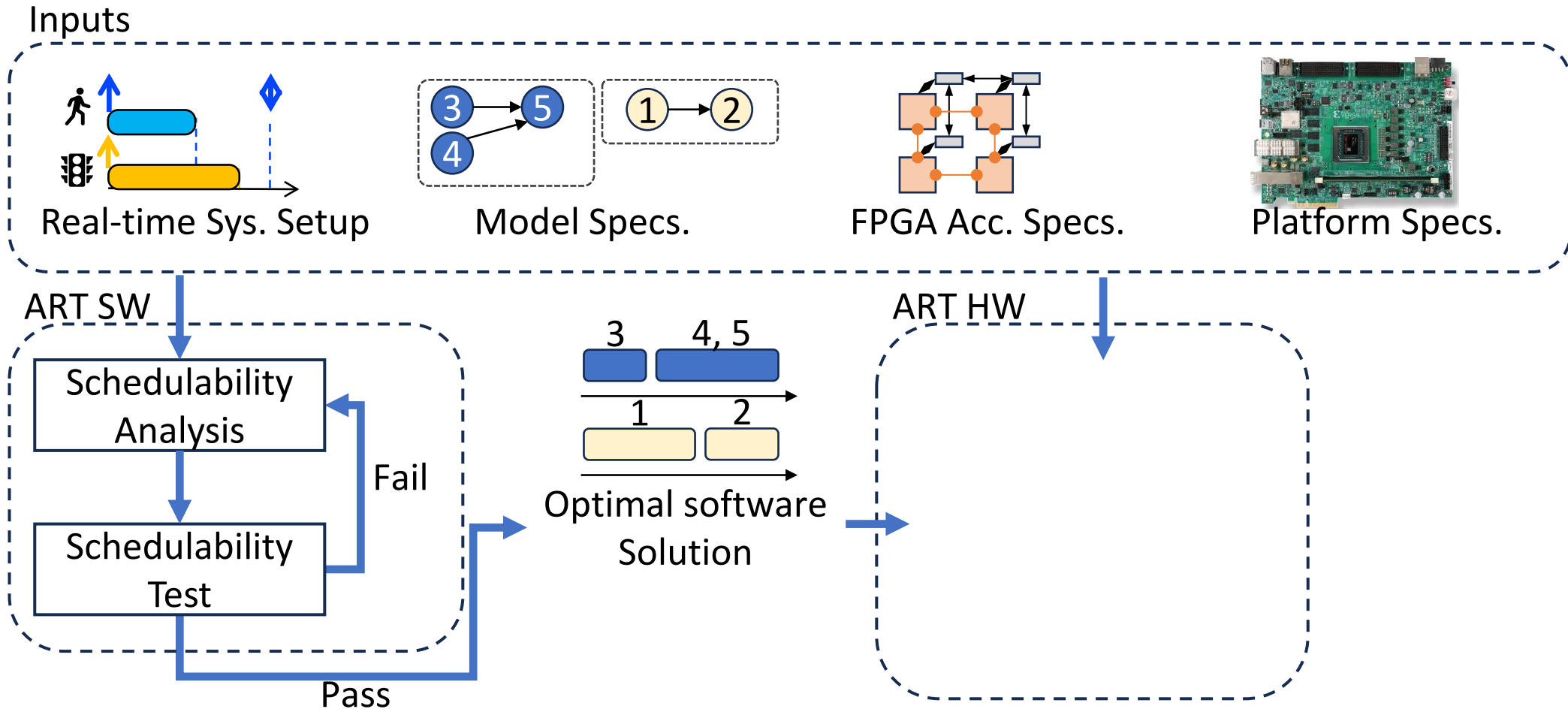
ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features



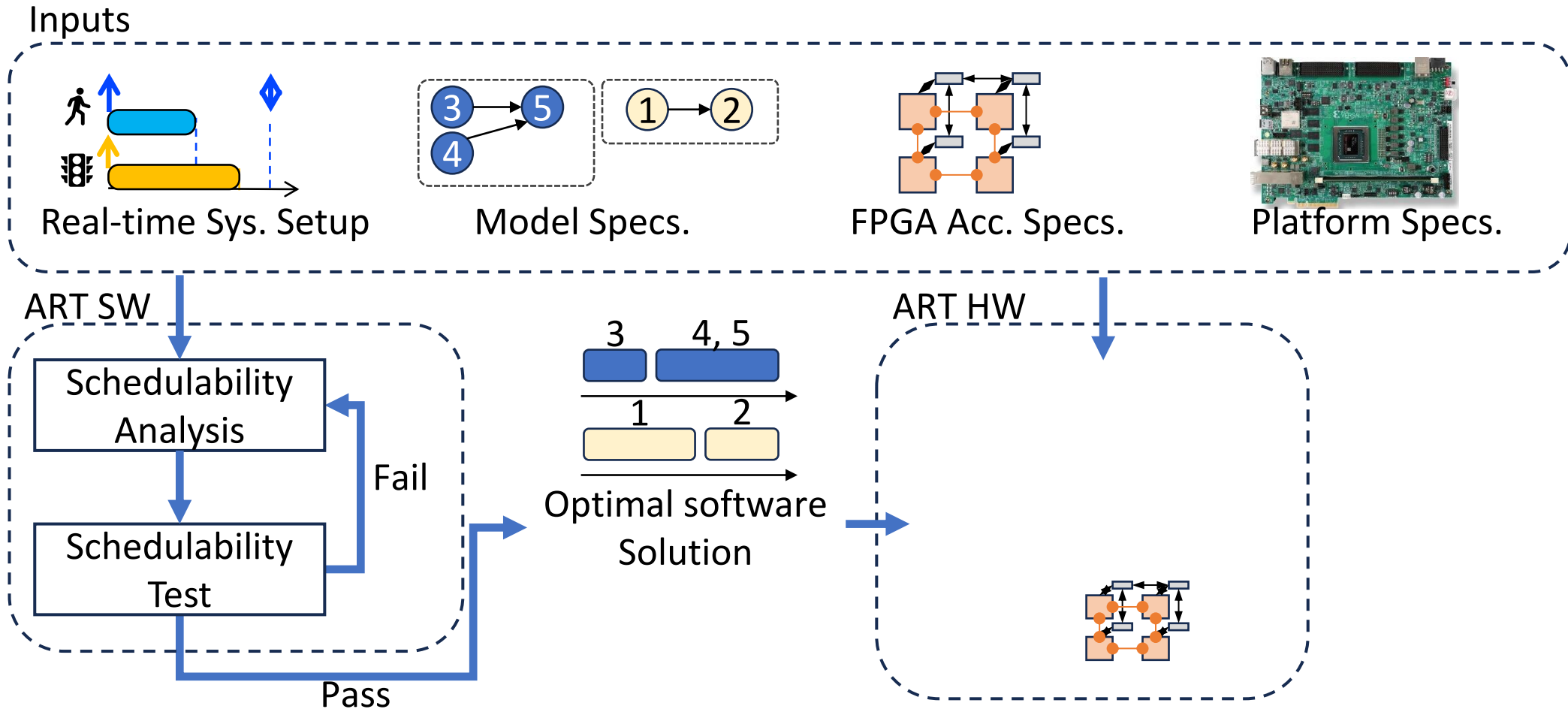
ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features



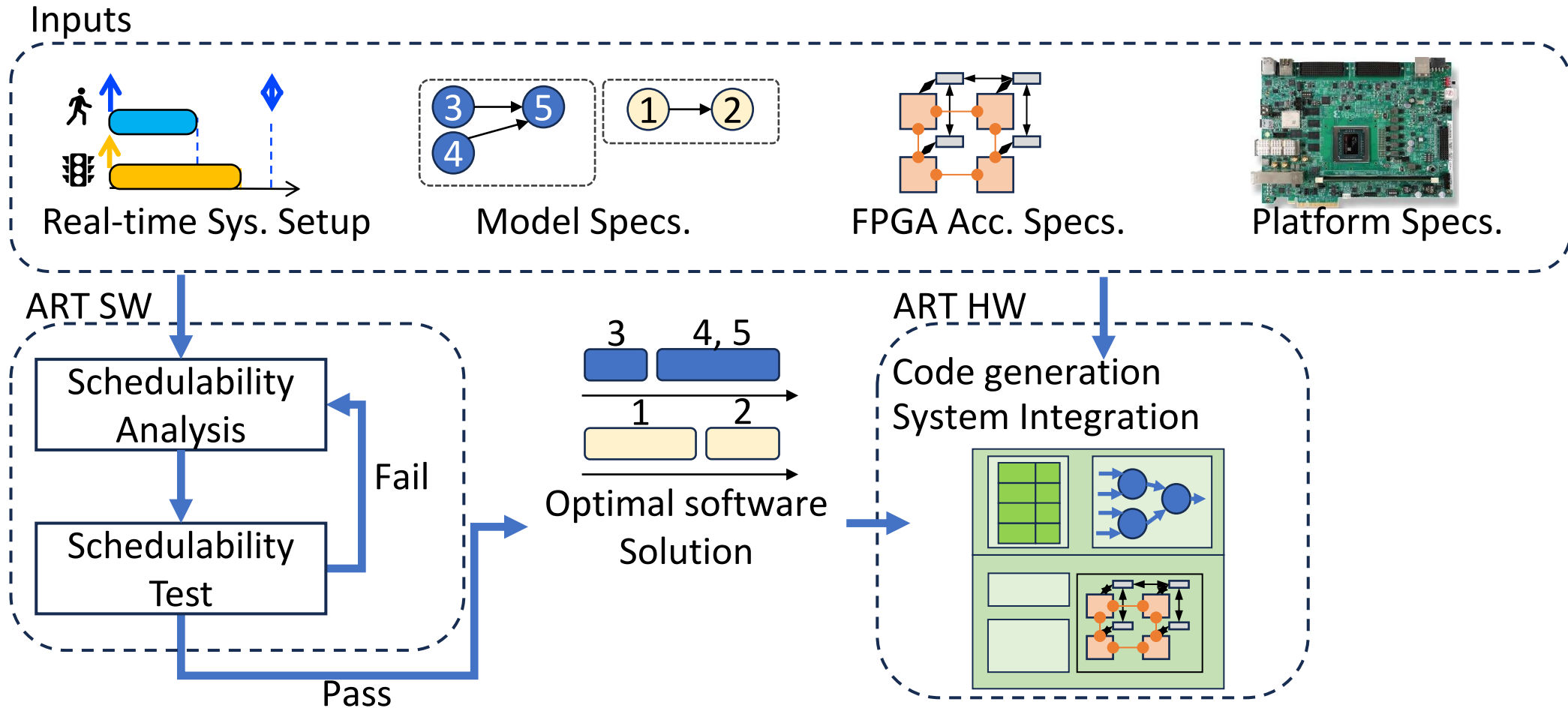
ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features



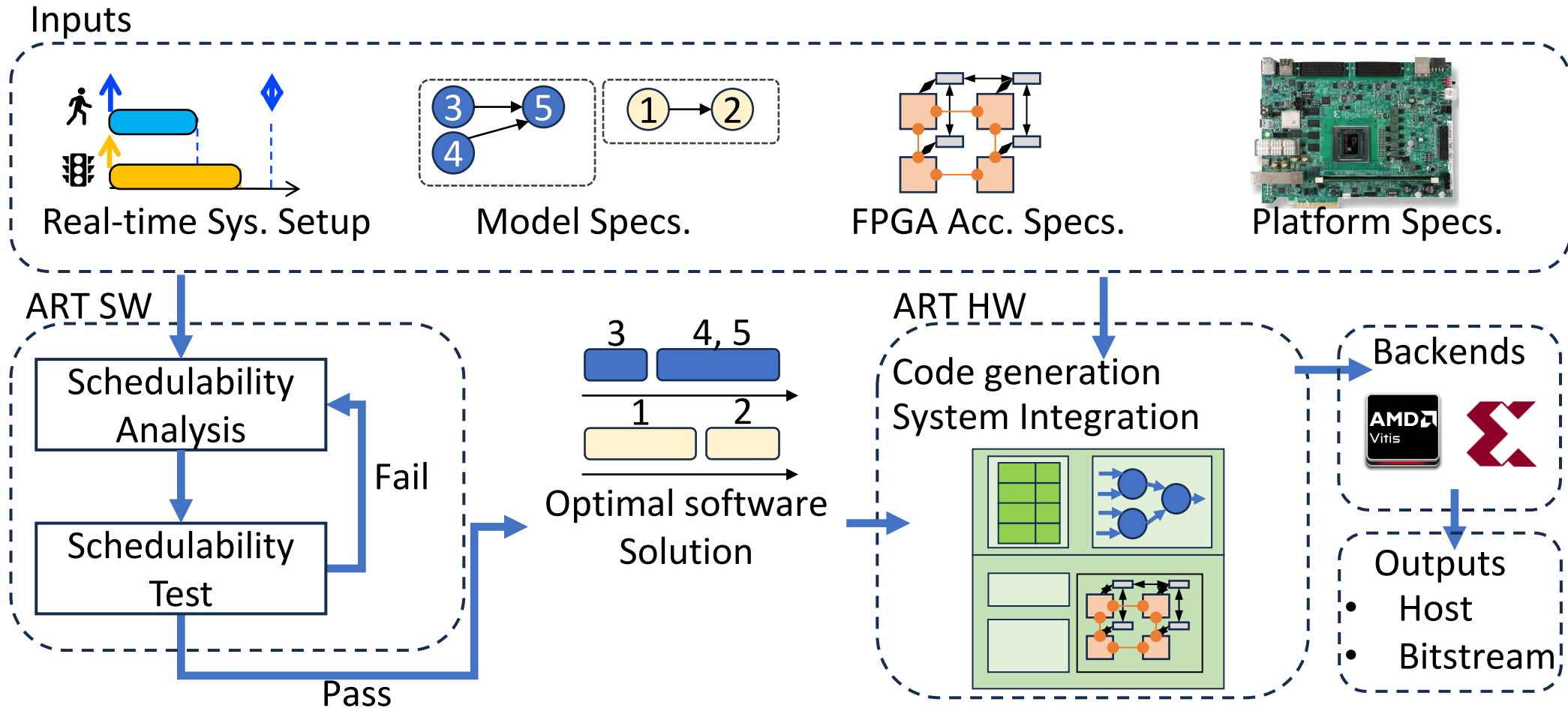
ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features

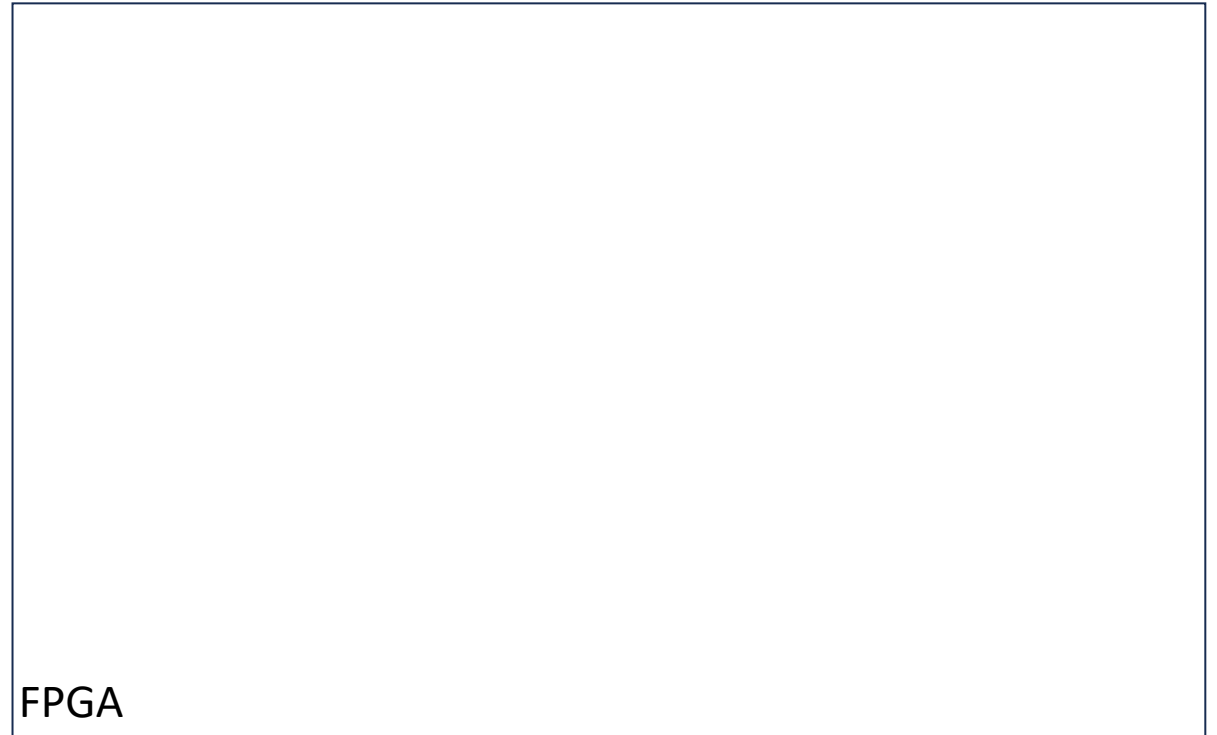


ART Framework: Overview

ART customizes a baseline FPGA HLS accelerator to support real-time features

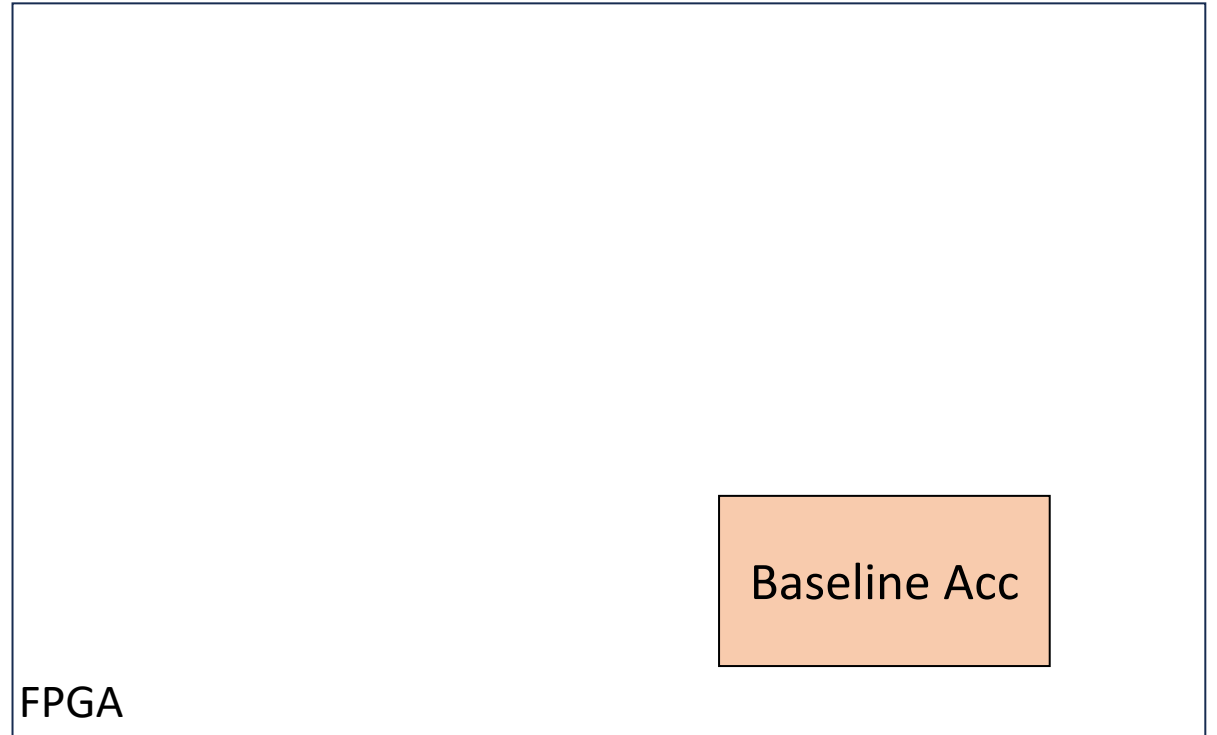


ART HW Framework: Accelerator Architecture



ART HW Framework: Accelerator Architecture

```
void baseline_acc(/*addr*/,/*specs*/){...}
```

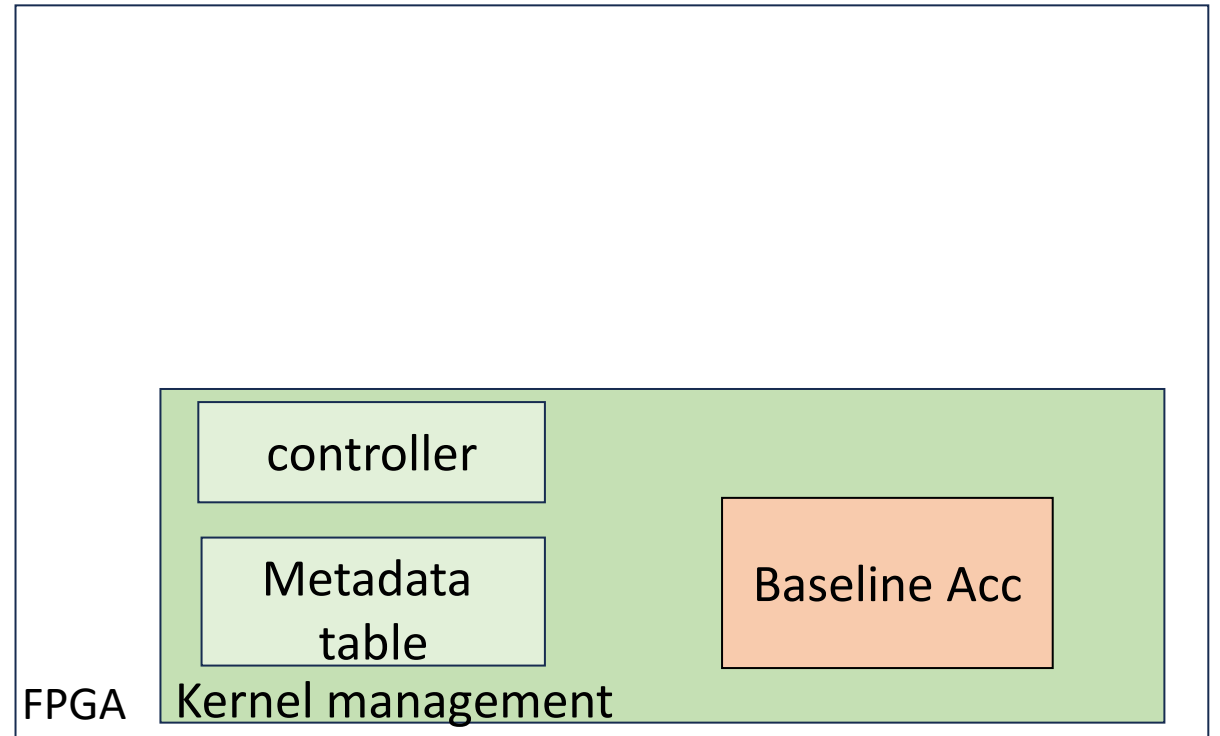


ART HW Framework: Accelerator Architecture

```
void baseline_acc(/*addr*/,/*specs*/){...}
```



```
void kernel_mgmt(...){  
    if (swap_in) {...}  
    if (swap_out) {...} //preemption support  
  
    get_addr(...);  
    get_specs(...);  
    baseline_acc(...); //Acc control  
}
```



ART HW Framework: Accelerator Architecture

```
void baseline_acc(/*addr*/,/*specs*/){...}
```



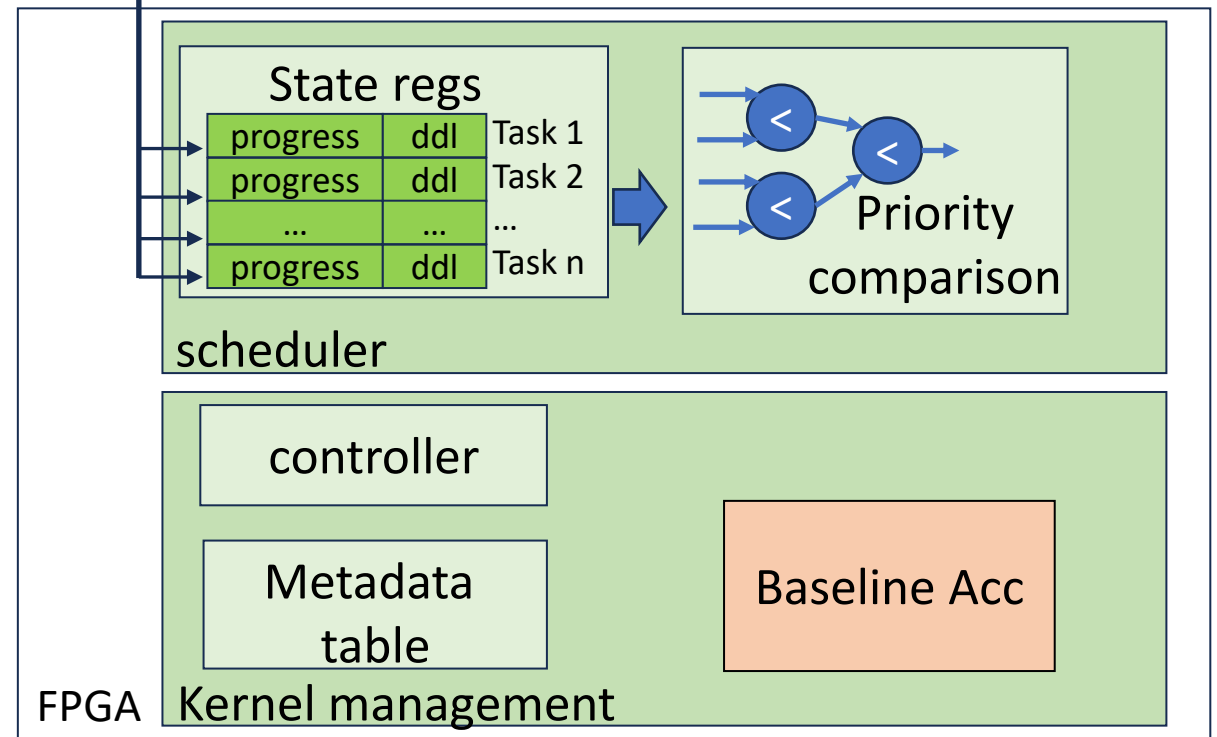
```
void kernel_mgmt(...){...}
```



(c-synth, system integration)

```
module top (...);  
  scheduler u_sche (...);  
  kernel_mgmt u_acc (...);  
  user_control u_control (...);  
endmodule
```

(task release)



ART HW Framework: Accelerator Architecture

```
void baseline_acc(/*addr*/,/*specs*/){...}
```



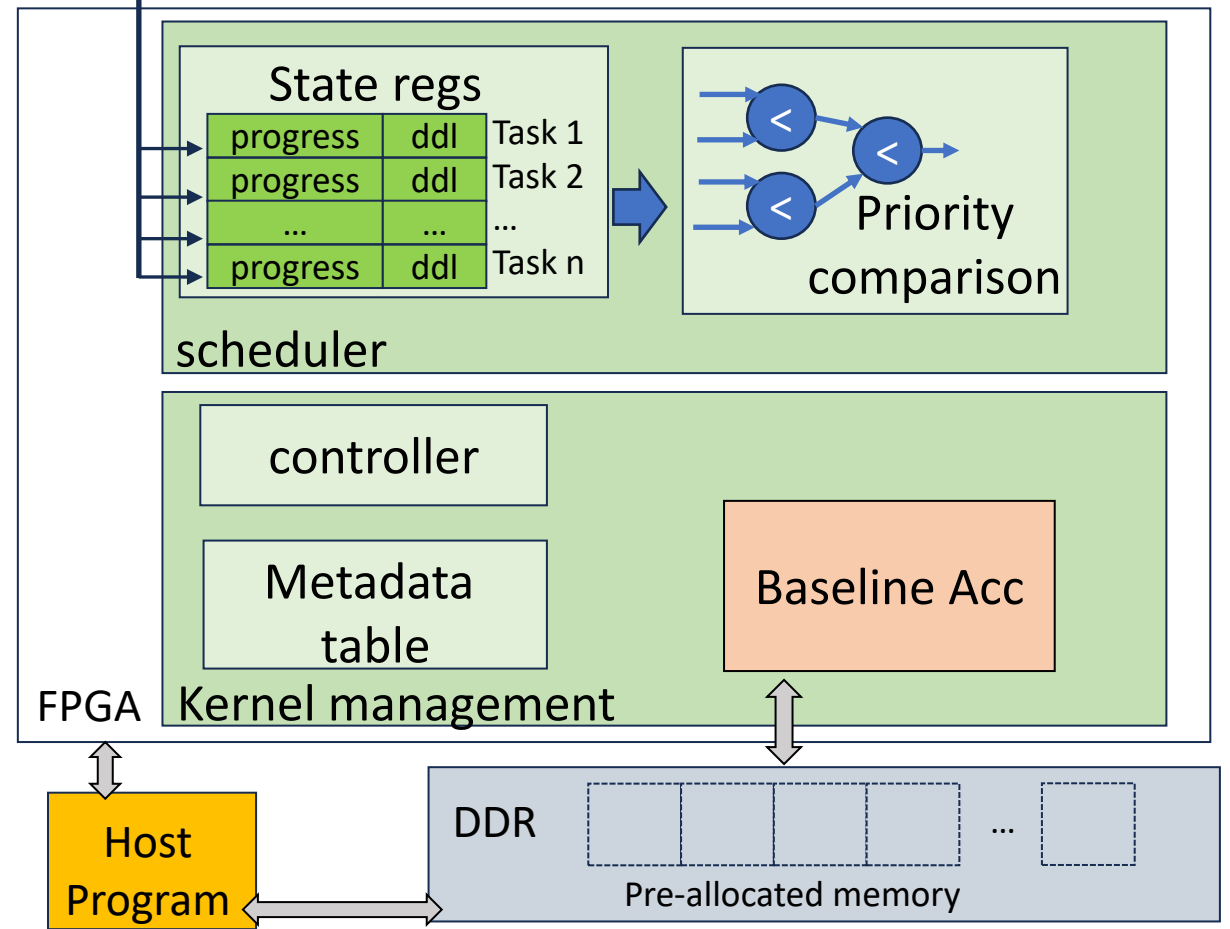
```
void kernel_mgmt(...){...}
```



(c-synth, system integration)

```
module top (...);  
  scheduler u_sche (...);  
  kernel_mgmt u_acc (...);  
  user_control u_control (...);  
endmodule
```

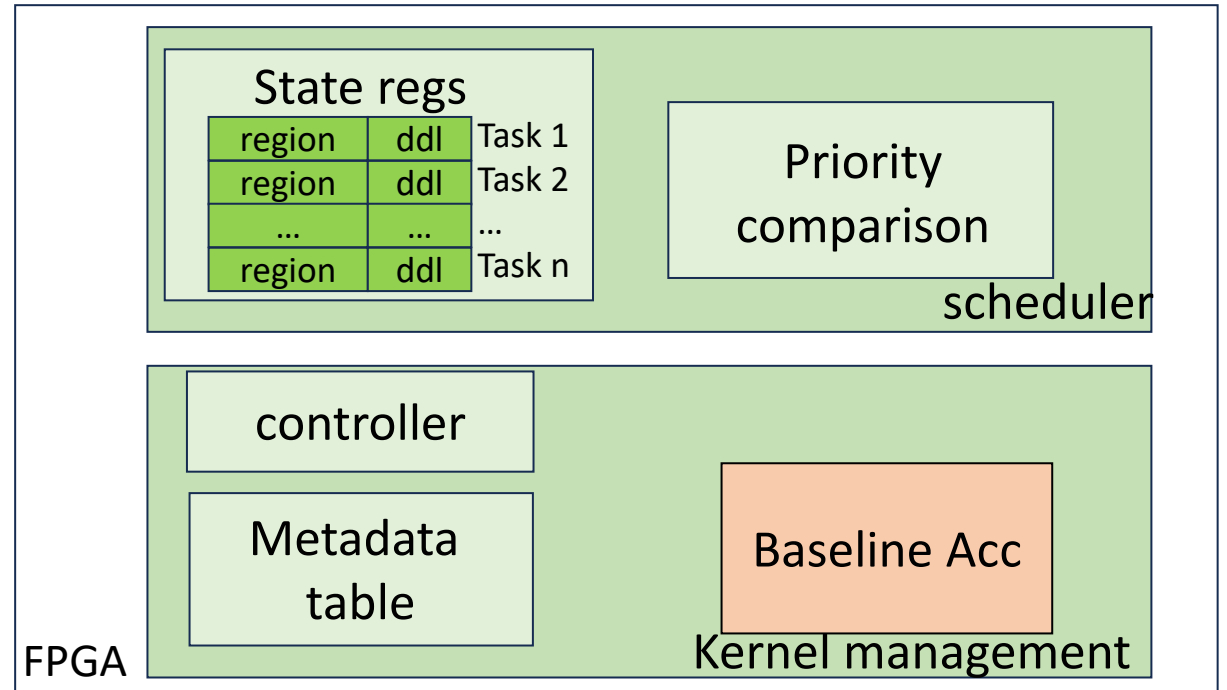
(task release)



ART HW Framework: Handling Various Tasks

ART uses overlay to handle different tasks within a task set

- Different DNN layers --> Same accelerator, different configuration files(metadata)



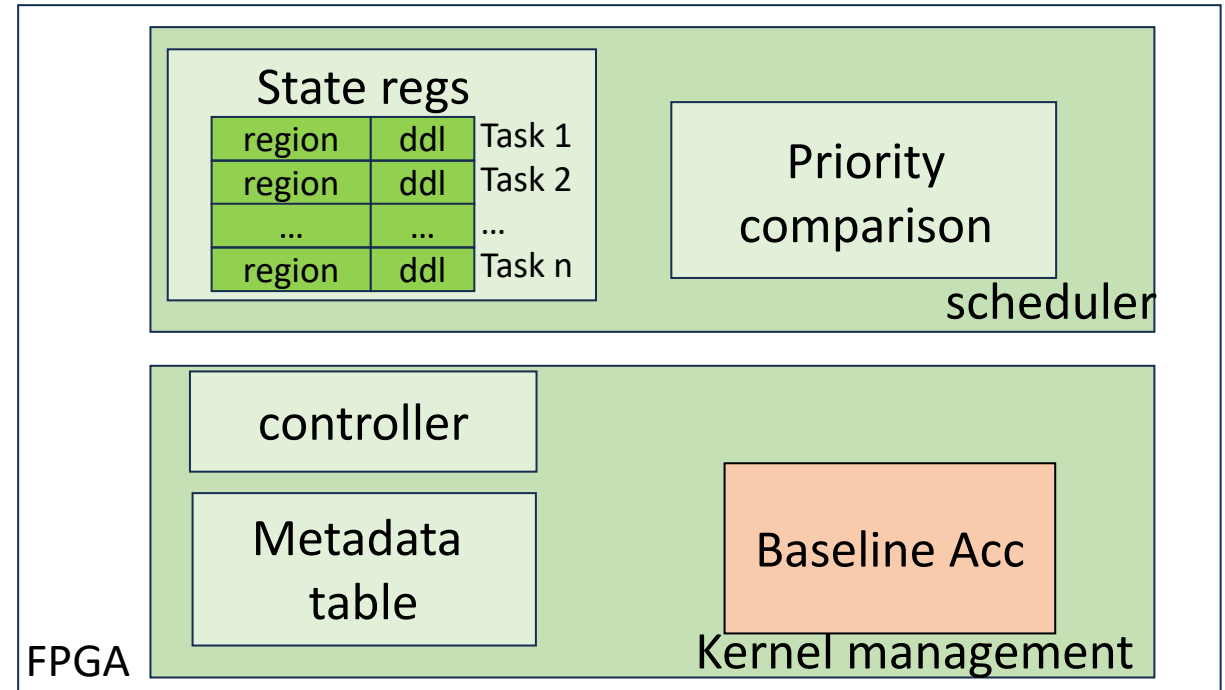
ART HW Framework: Handling Various Tasks

ART uses overlay to handle different tasks within a task set

- Different DNN layers --> Same accelerator, different configuration files(metadata)

ART segments the DNN models in unit of DNN layers

- Assume intermediate data is on DDR



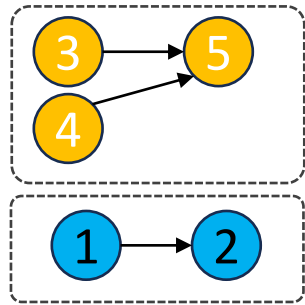
ART HW Framework: Handling Various Tasks

ART uses overlay to handle different tasks within a task set

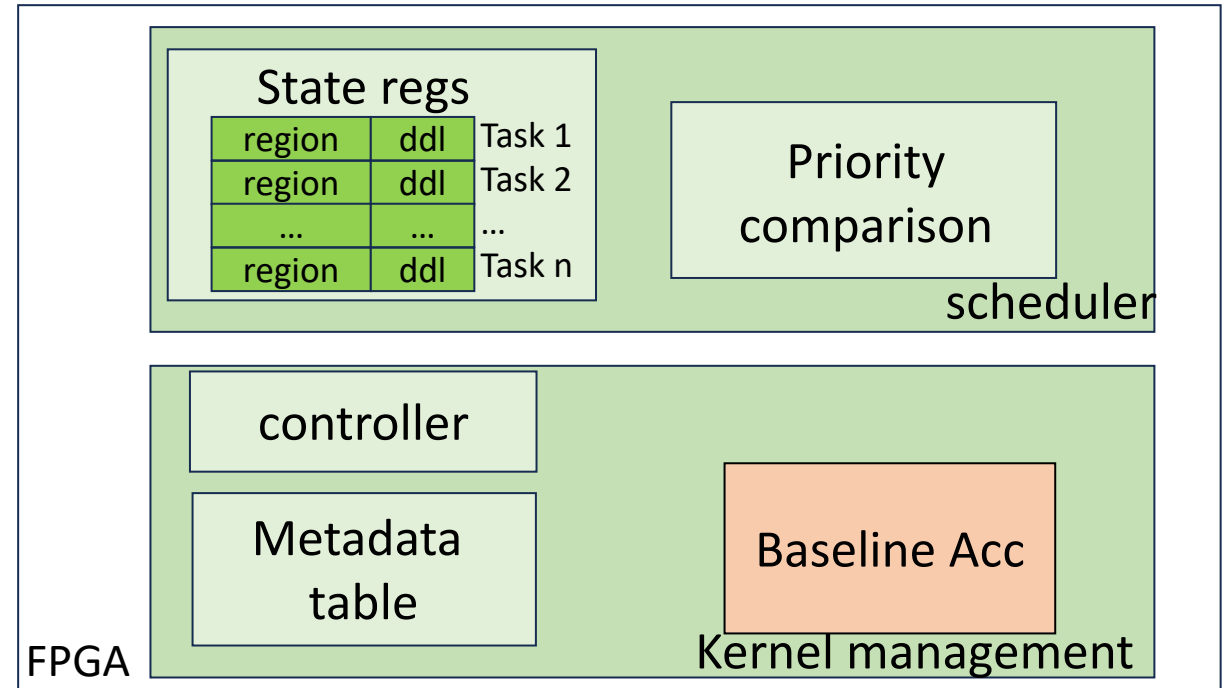
- Different DNN layers --> Same accelerator, different configuration files(metadata)

ART segments the DNN models in unit of DNN layers

- Assume intermediate data is on DDR



Task Configs



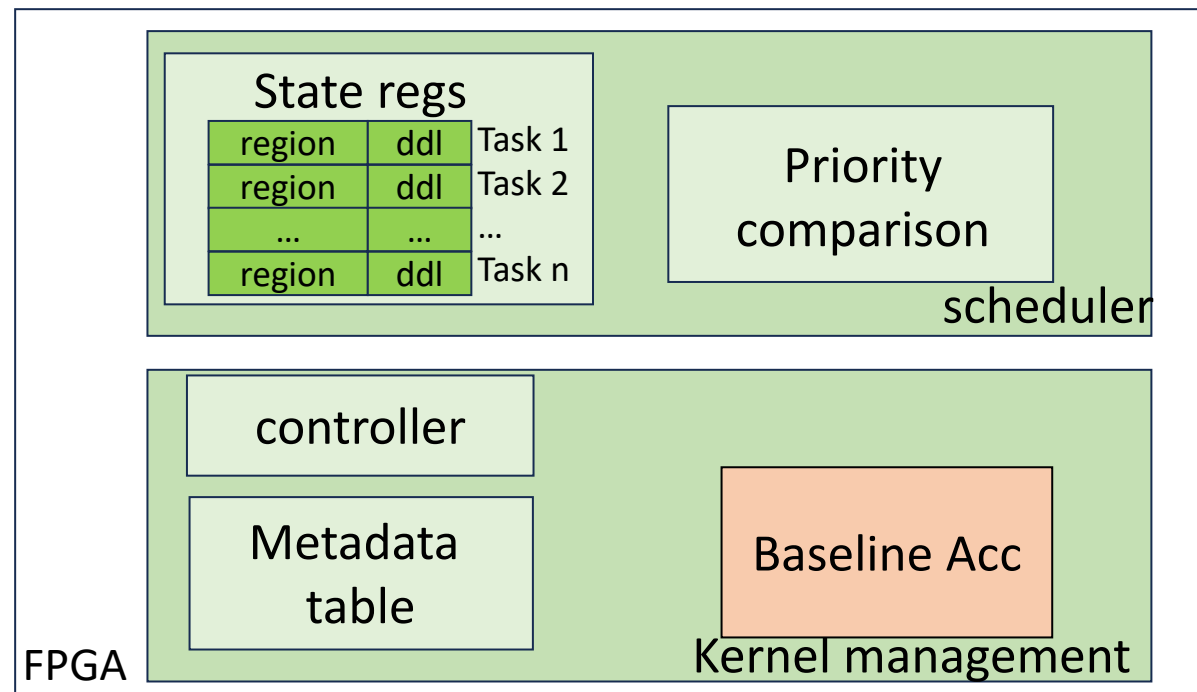
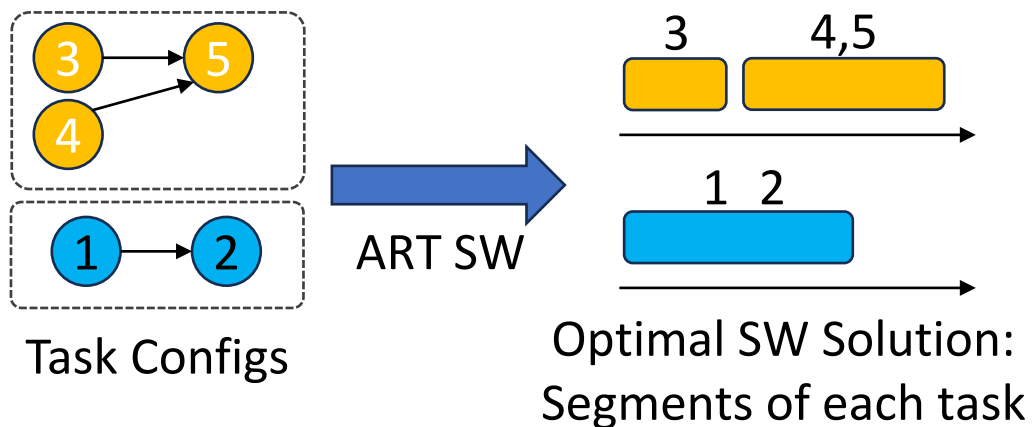
ART HW Framework: Handling Various Tasks

ART uses overlay to handle different tasks within a task set

- Different DNN layers --> Same accelerator, different configuration files(metadata)

ART segments the DNN models in unit of DNN layers

- Assume intermediate data is on DDR



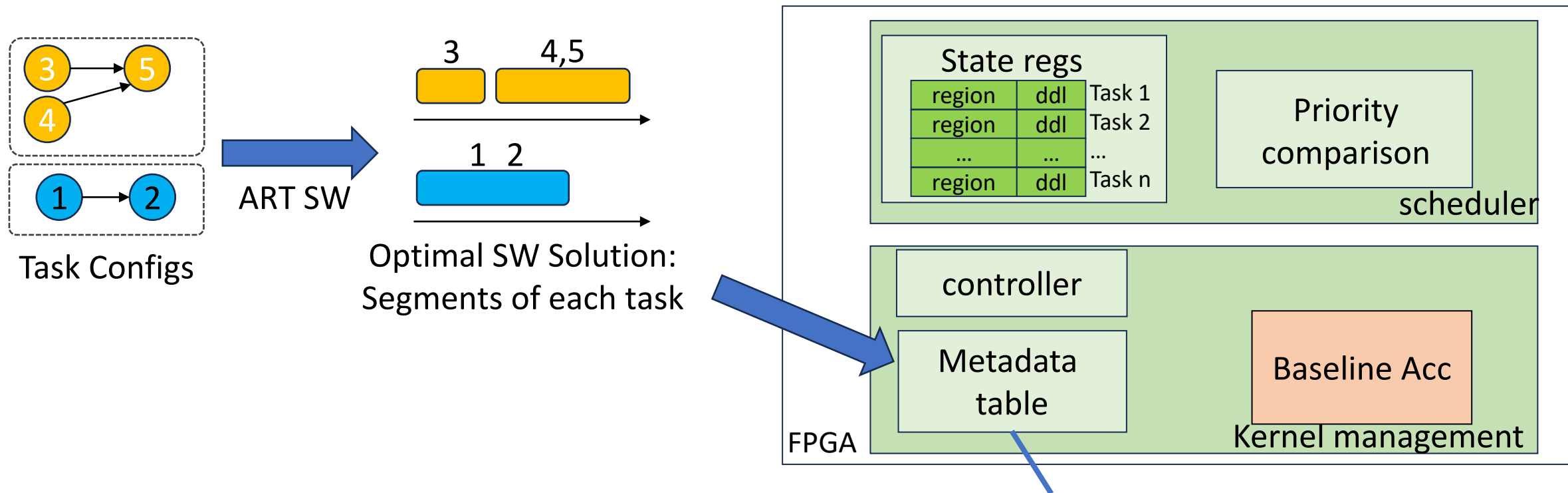
ART HW Framework: Handling Various Tasks

ART uses overlay to handle different tasks within a task set

- Different DNN layers --> Same accelerator, different configuration files(metadata)

ART segments the DNN models in unit of DNN layers

- Assume intermediate data is on DDR



- Iteration boundaries
- Address info

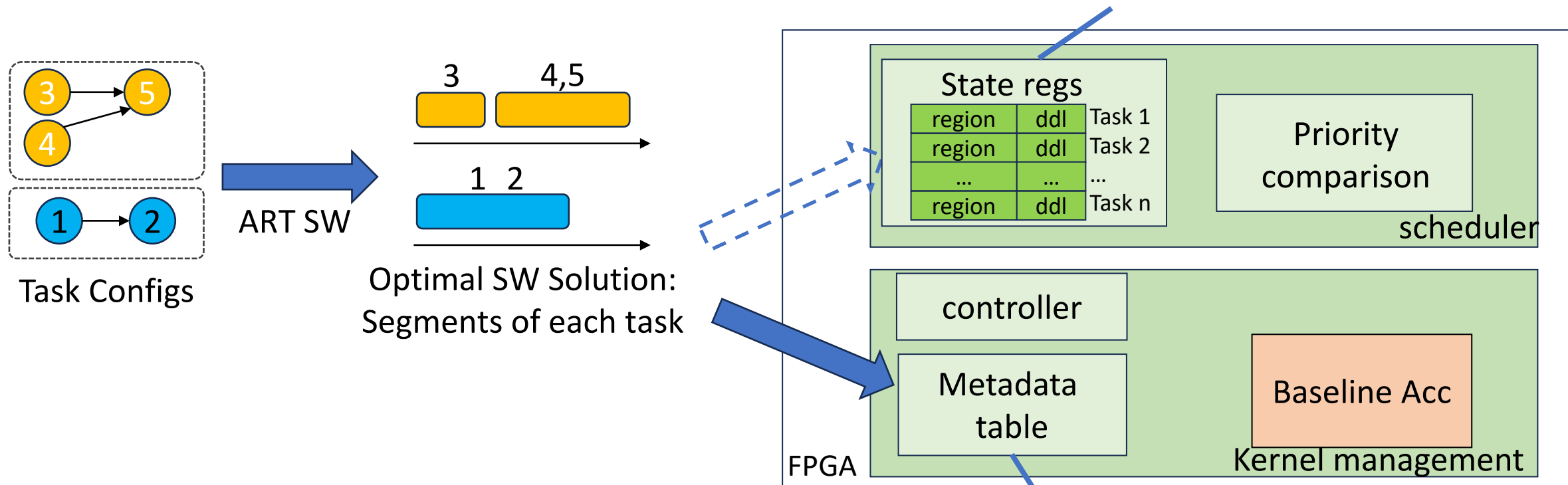
ART HW Framework: Handling Various Tasks

ART uses overlay to handle different tasks within a task set

- Different DNN layers --> Same accelerator, different configuration files(metadata)

ART segments the DNN models in unit of DNN layers

- Assume intermediate data is on DDR

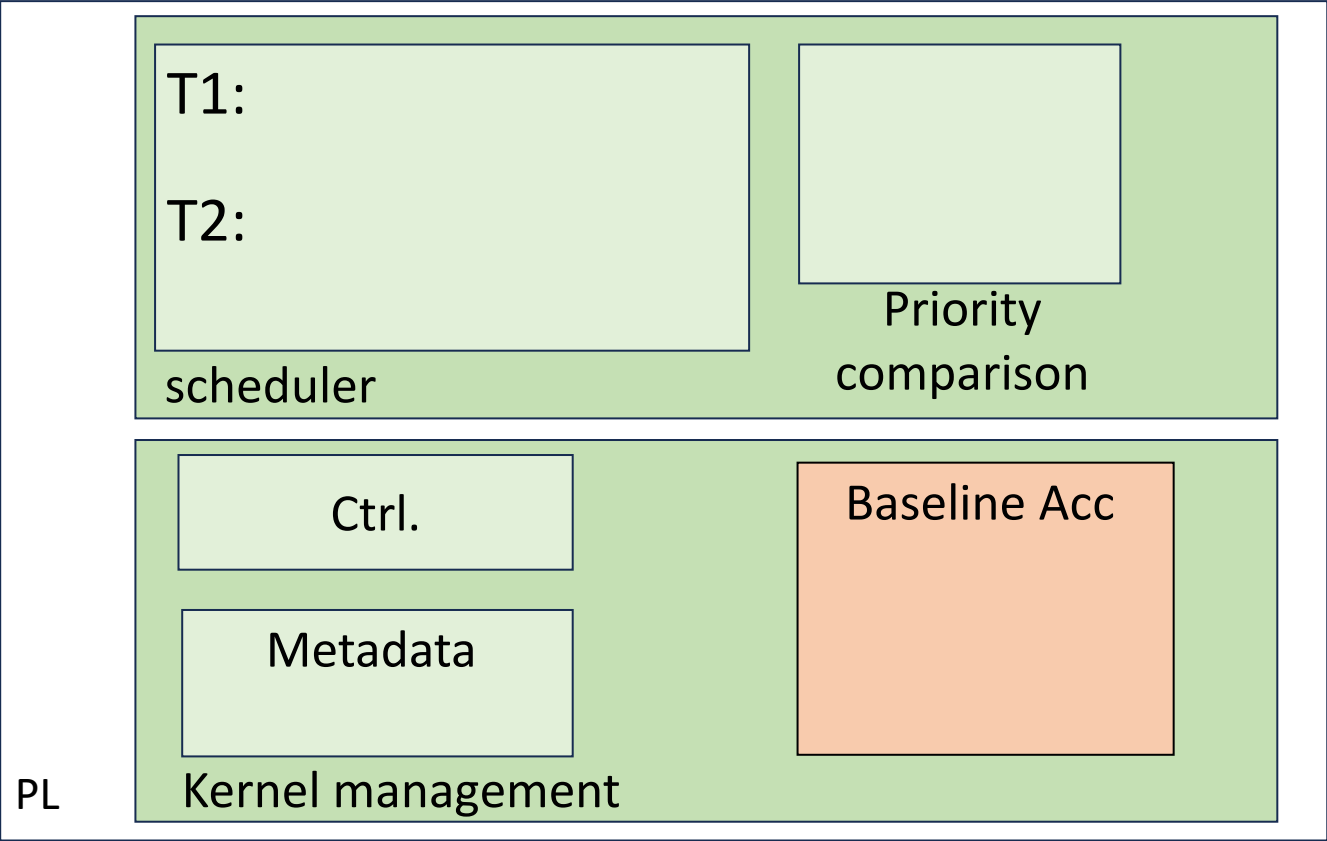


- Iteration boundaries
- Address info

Scheduling & Preemption Support: Running Example

T=0:
Task 1 release a job 1, ddl = 20

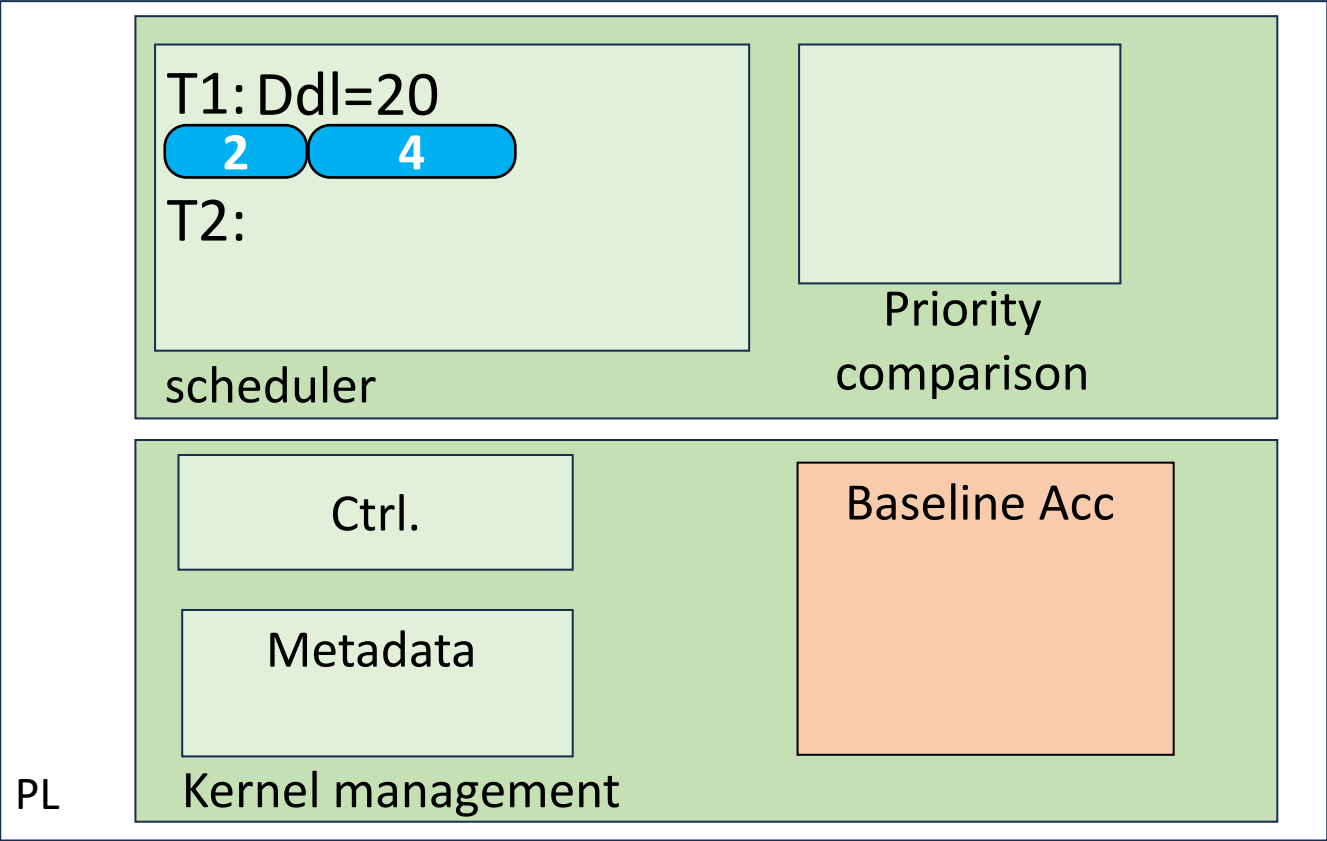
Task 1: 2 segments:



Scheduling & Preemption Support: Running Example

T=0:
Task 1 release a job 1, ddl = 20

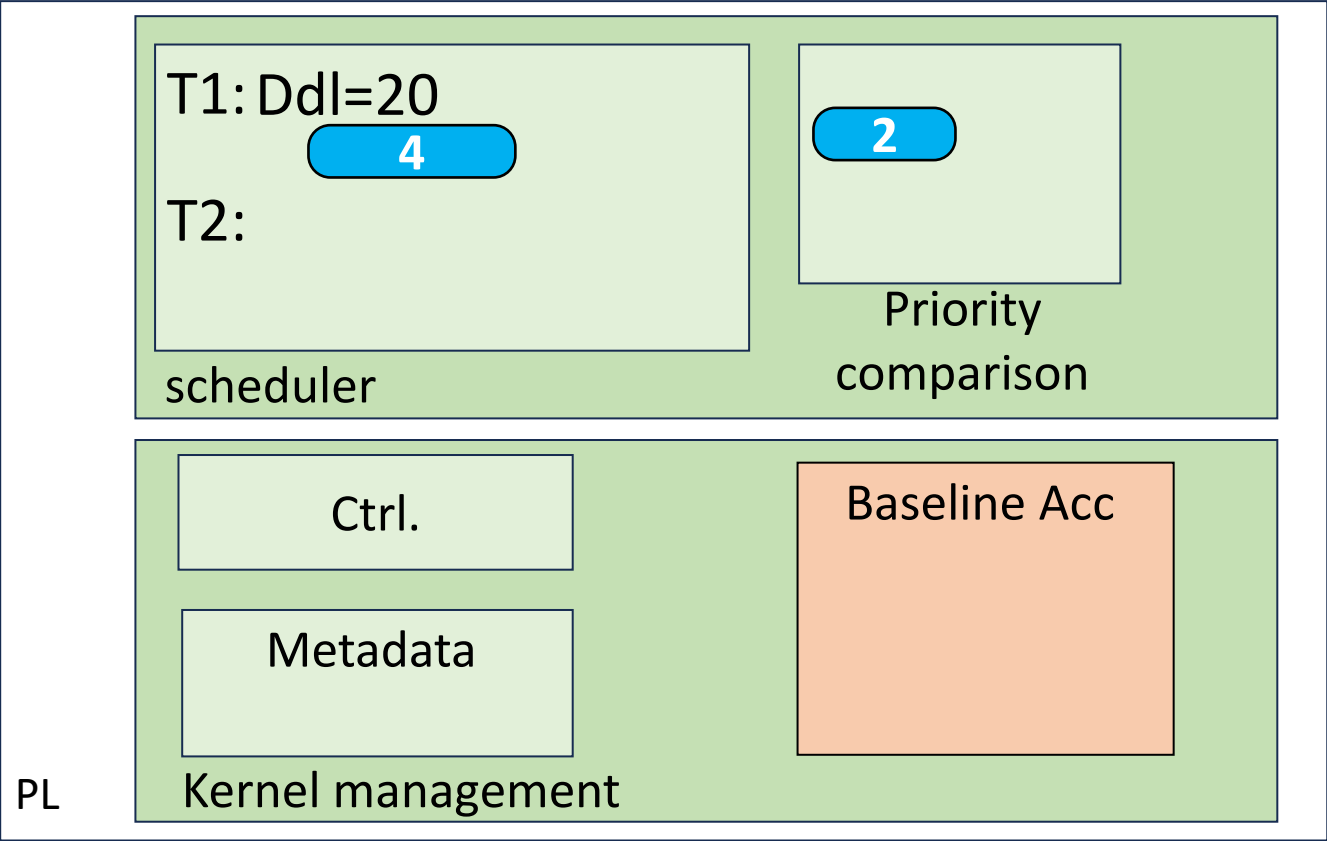
Task 1: 2 segments:



Scheduling & Preemption Support: Running Example

T=0:
Task 1 release a job 1, ddl = 20

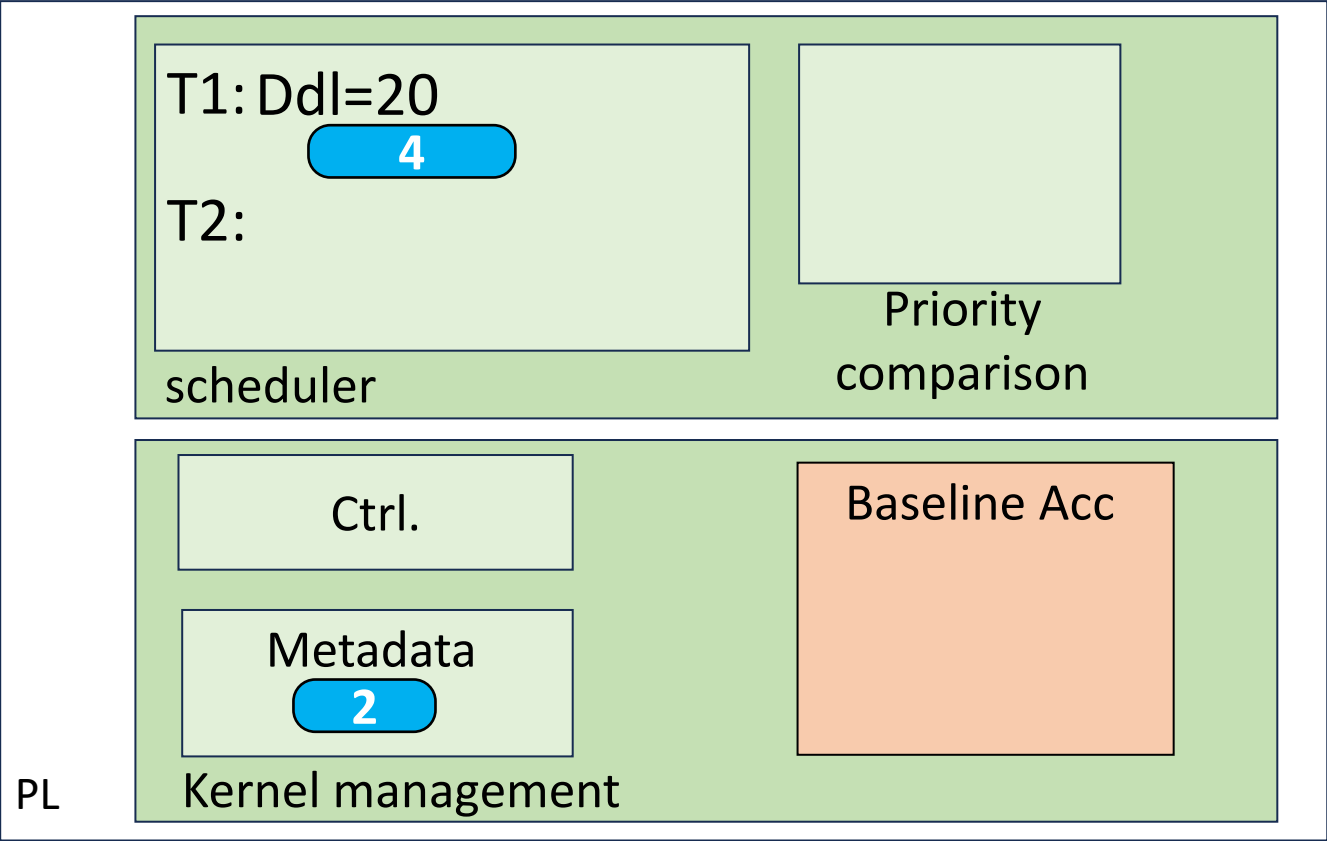
Task 1: 2 segments:



Scheduling & Preemption Support: Running Example

T=0:
Task 1 release a job 1, ddl = 20

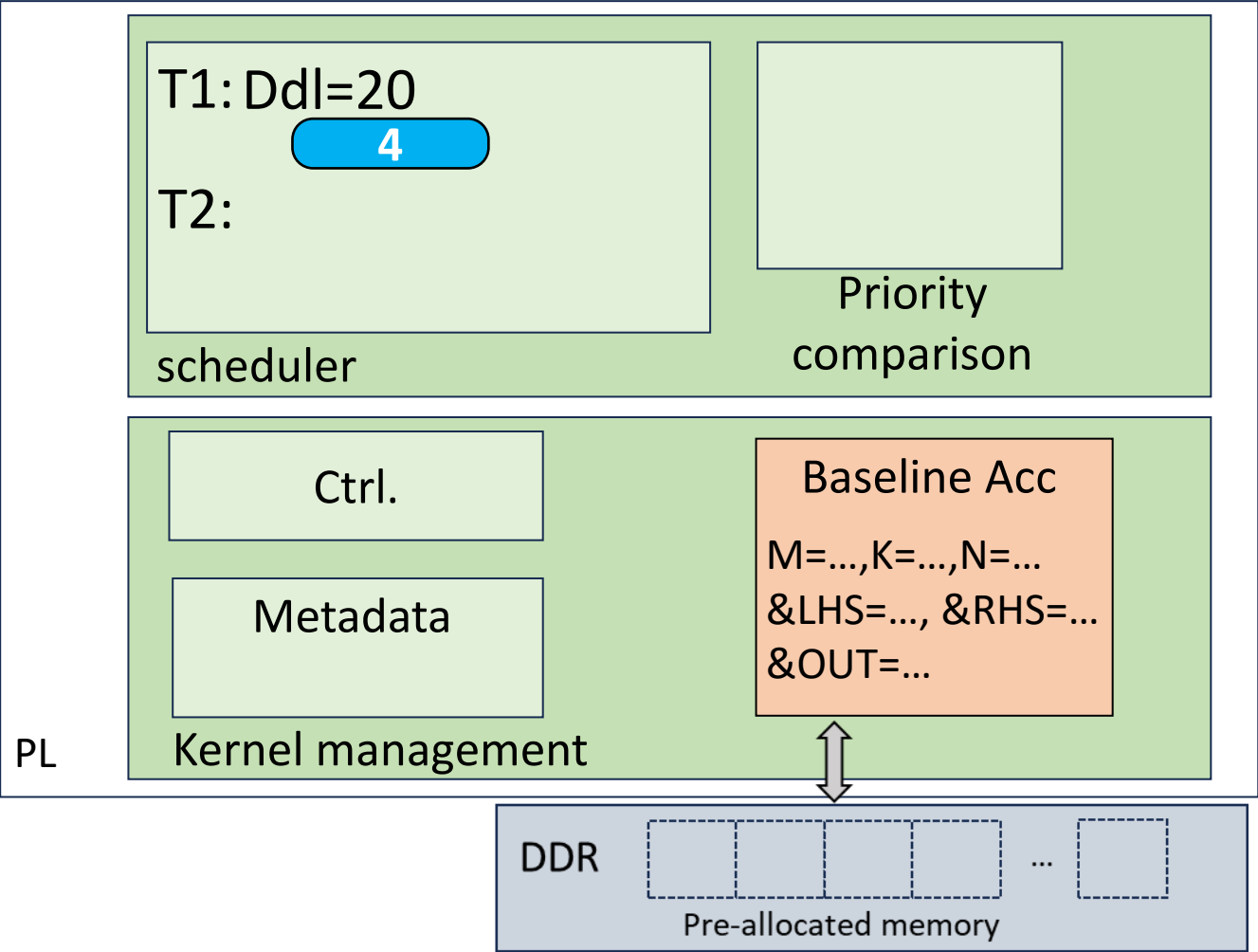
Task 1: 2 segments:




Scheduling & Preemption Support: Running Example

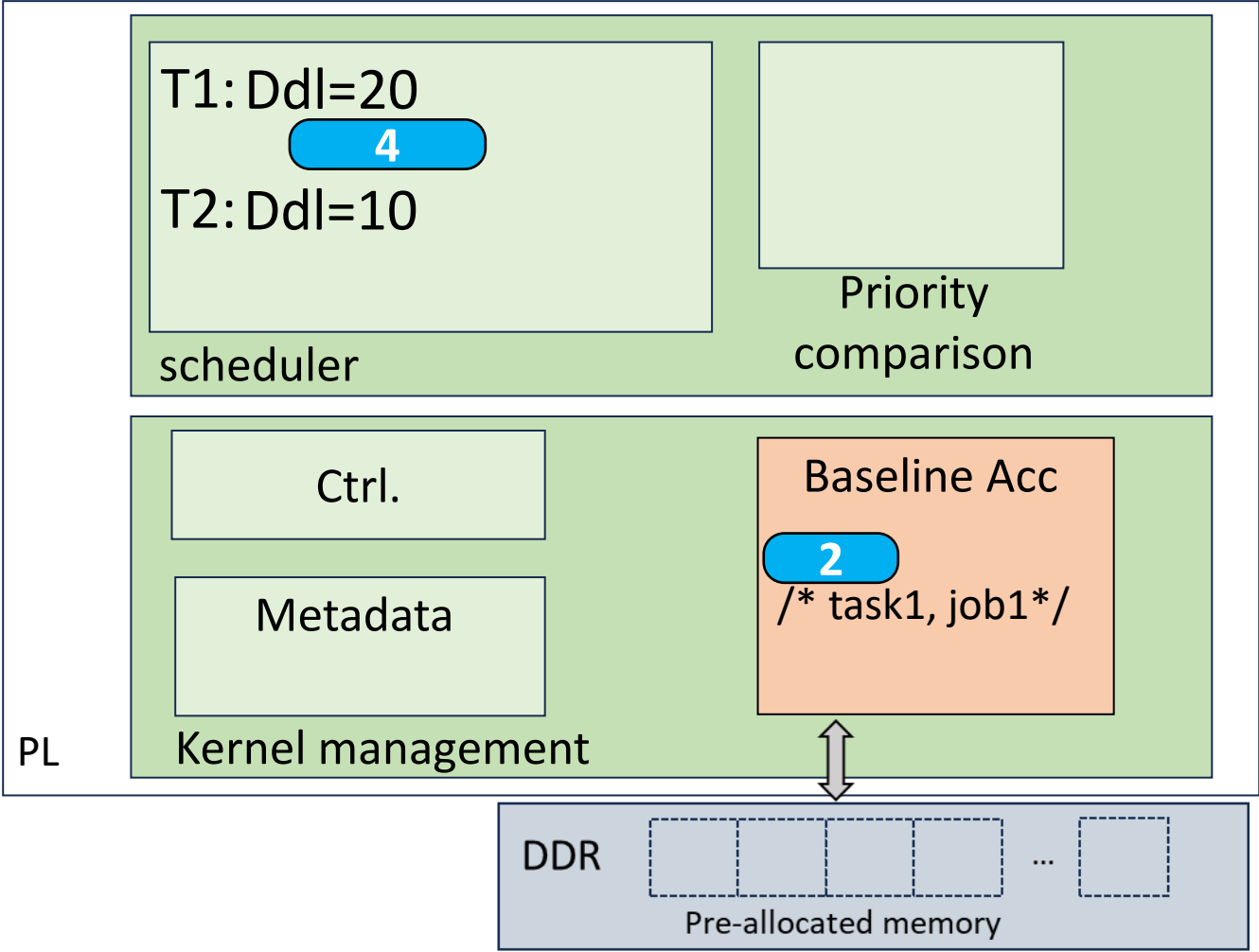
T=0:
Task 1 release a job 1, ddl = 20

Task 1: 2 segments:



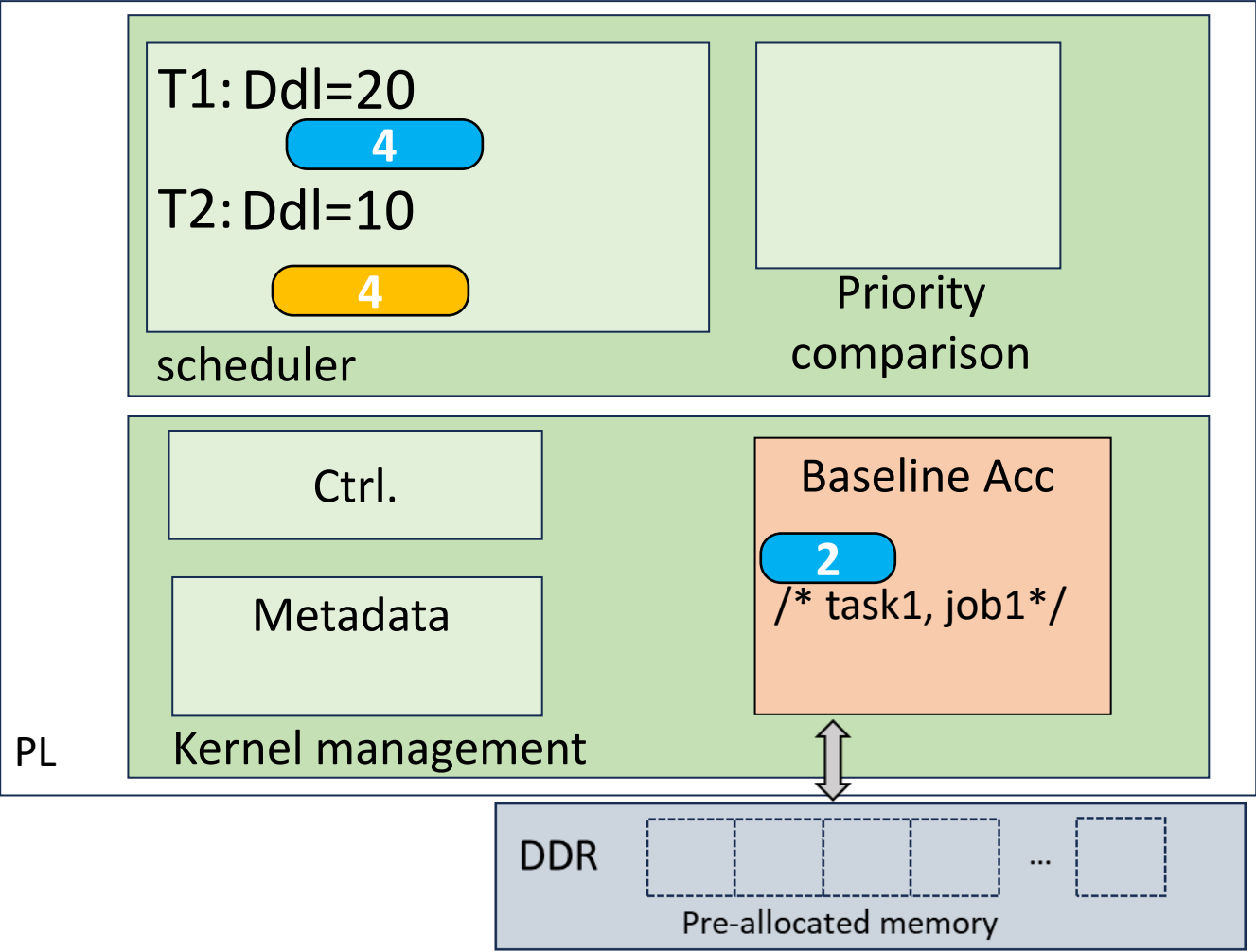
Scheduling & Preemption Support: Running Example

T=1:
Task2 release a job1, ddl=10
Task 2: 1 segment:




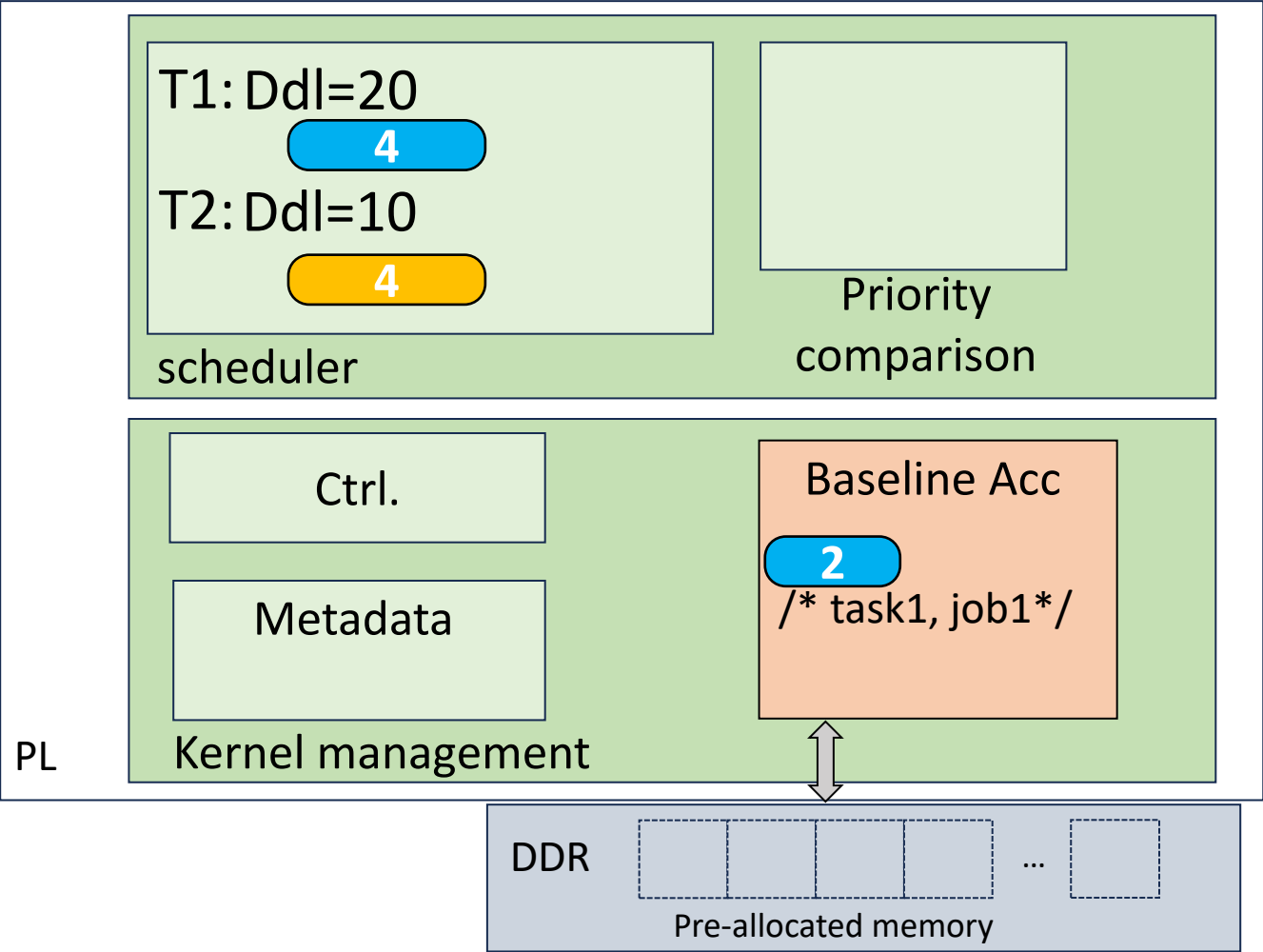
Scheduling & Preemption Support: Running Example

T=1:
Task2 release a job1, ddl=10
Task 2: 1 segment:



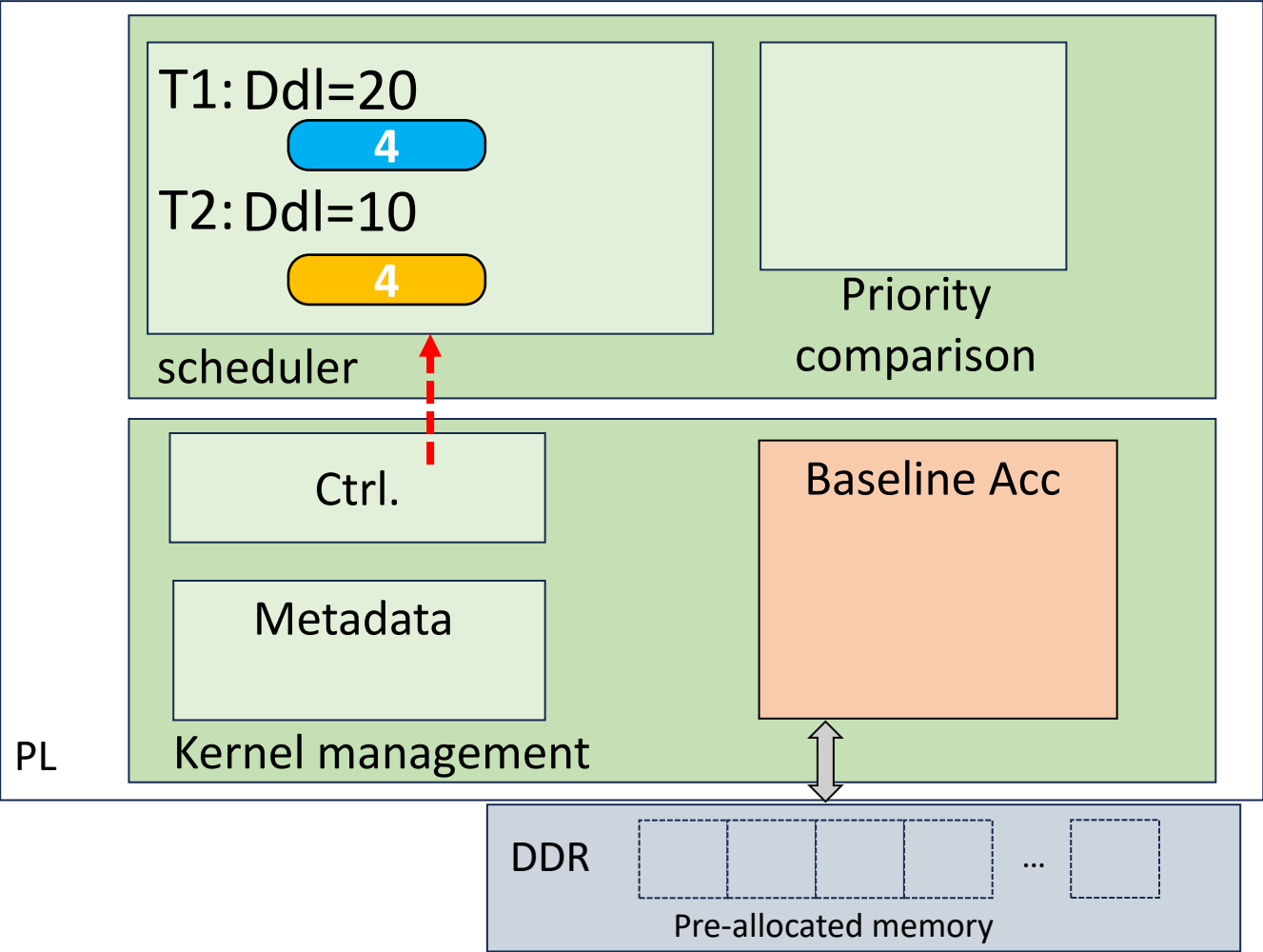
Scheduling & Preemption Support: Running Example

T=2:
Task 1 job 1 finishes its first
segment



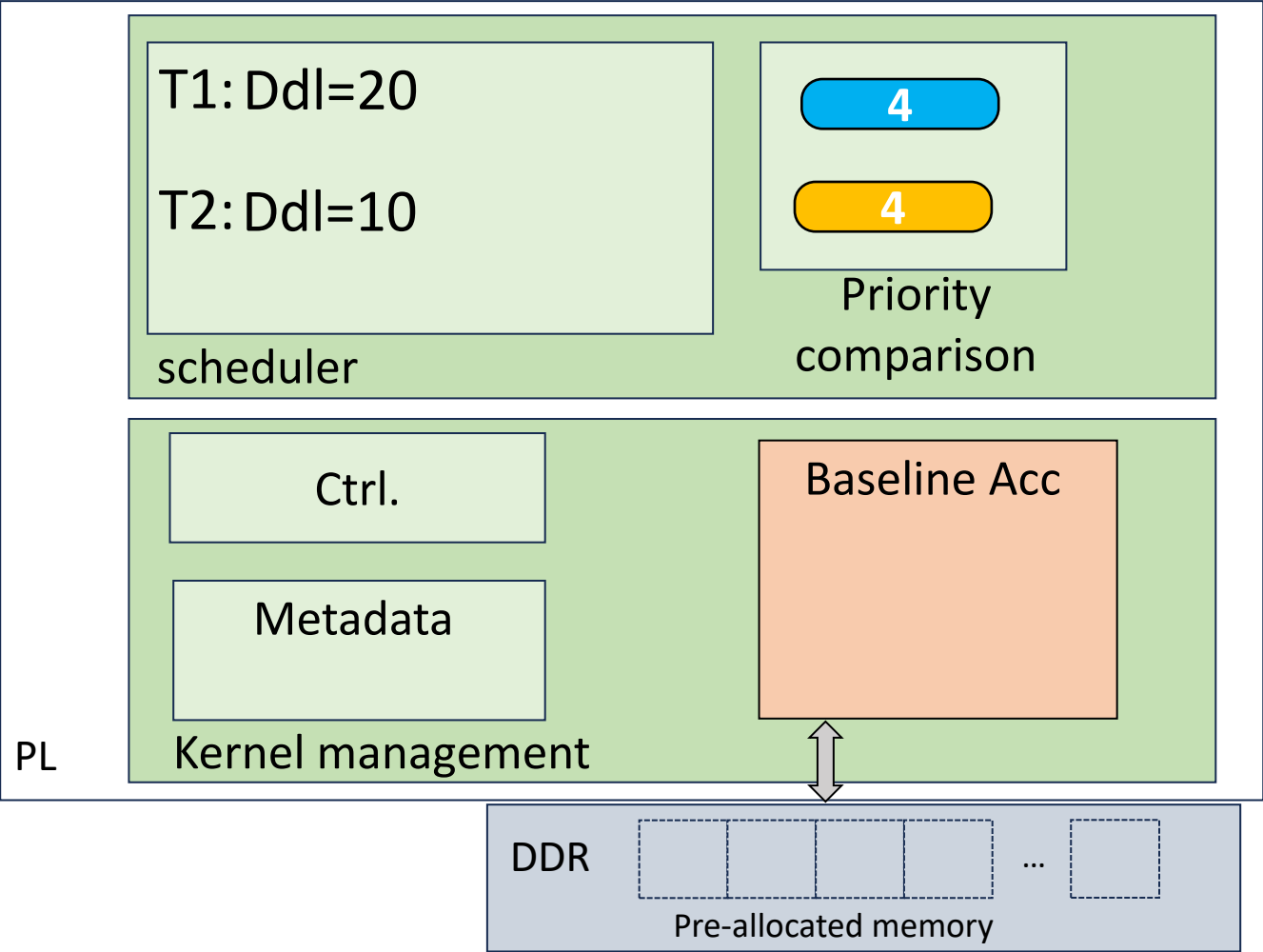
Scheduling & Preemption Support: Running Example

T=2:
Task 1 job 1 finishes its first segment



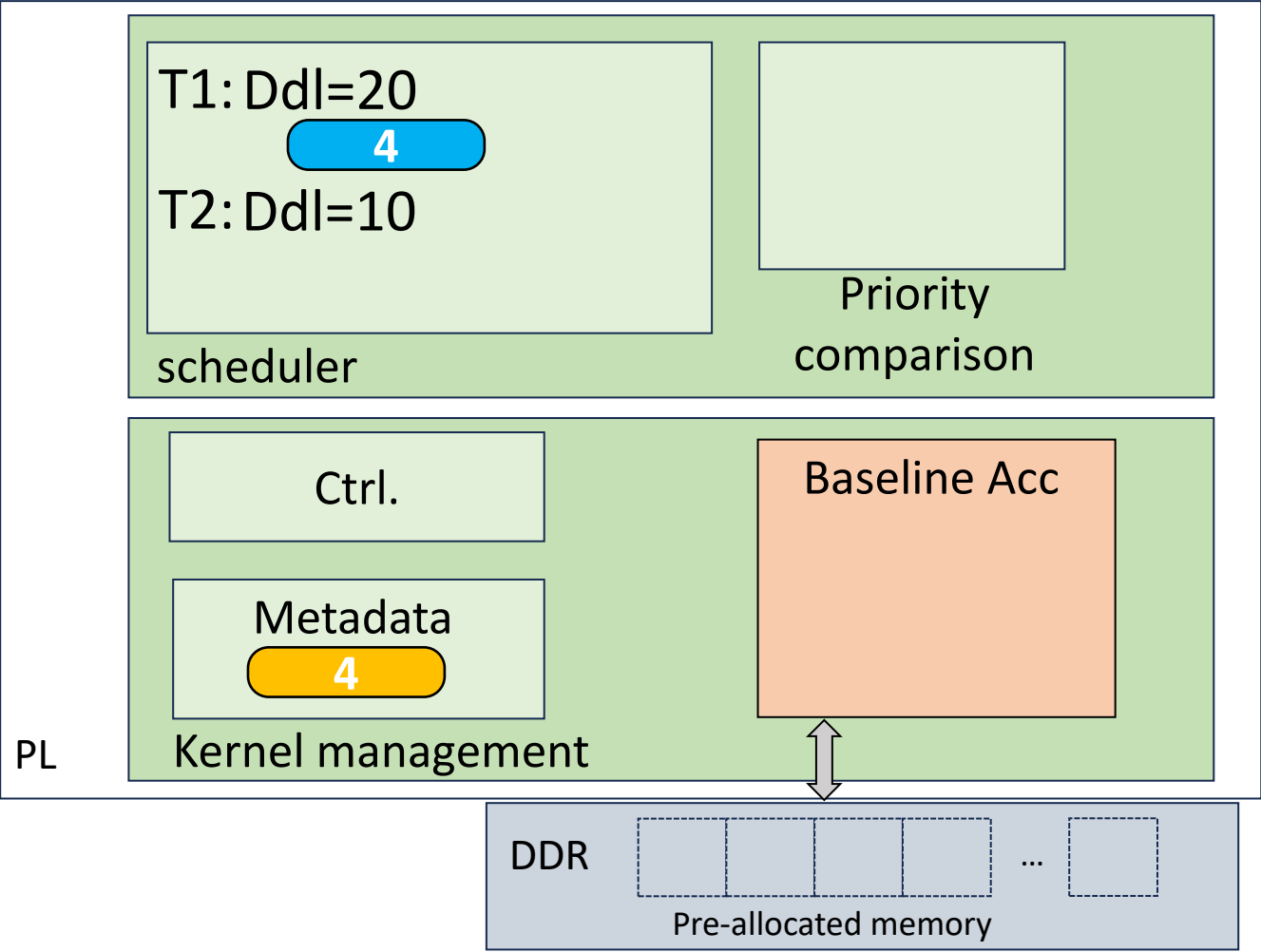
Scheduling & Preemption Support: Running Example

T=2:
Task 1 job 1 finishes its first
segment



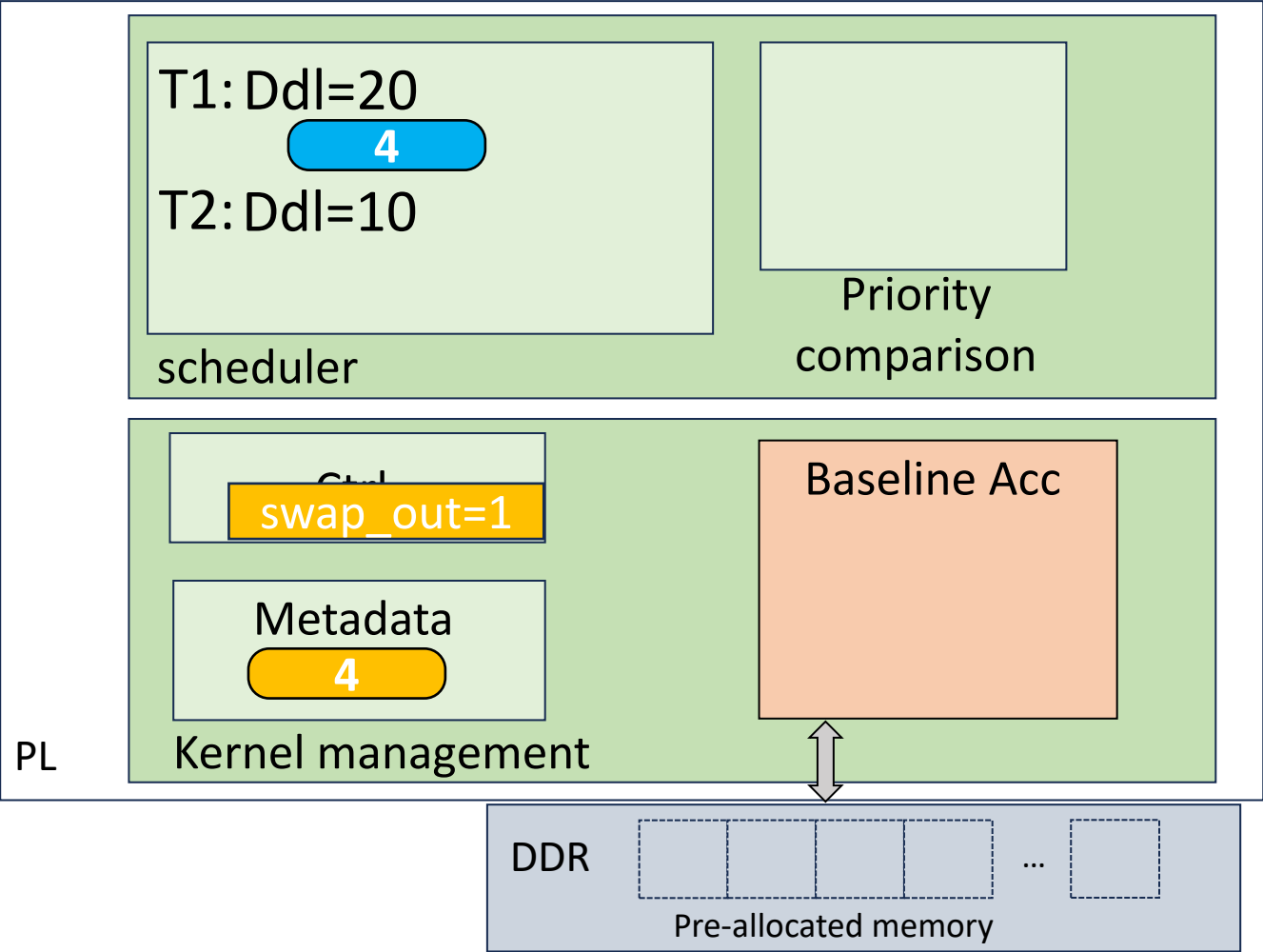
Scheduling & Preemption Support: Running Example

T=2:
Task 1 job 1 finishes its first segment



Scheduling & Preemption Support: Running Example

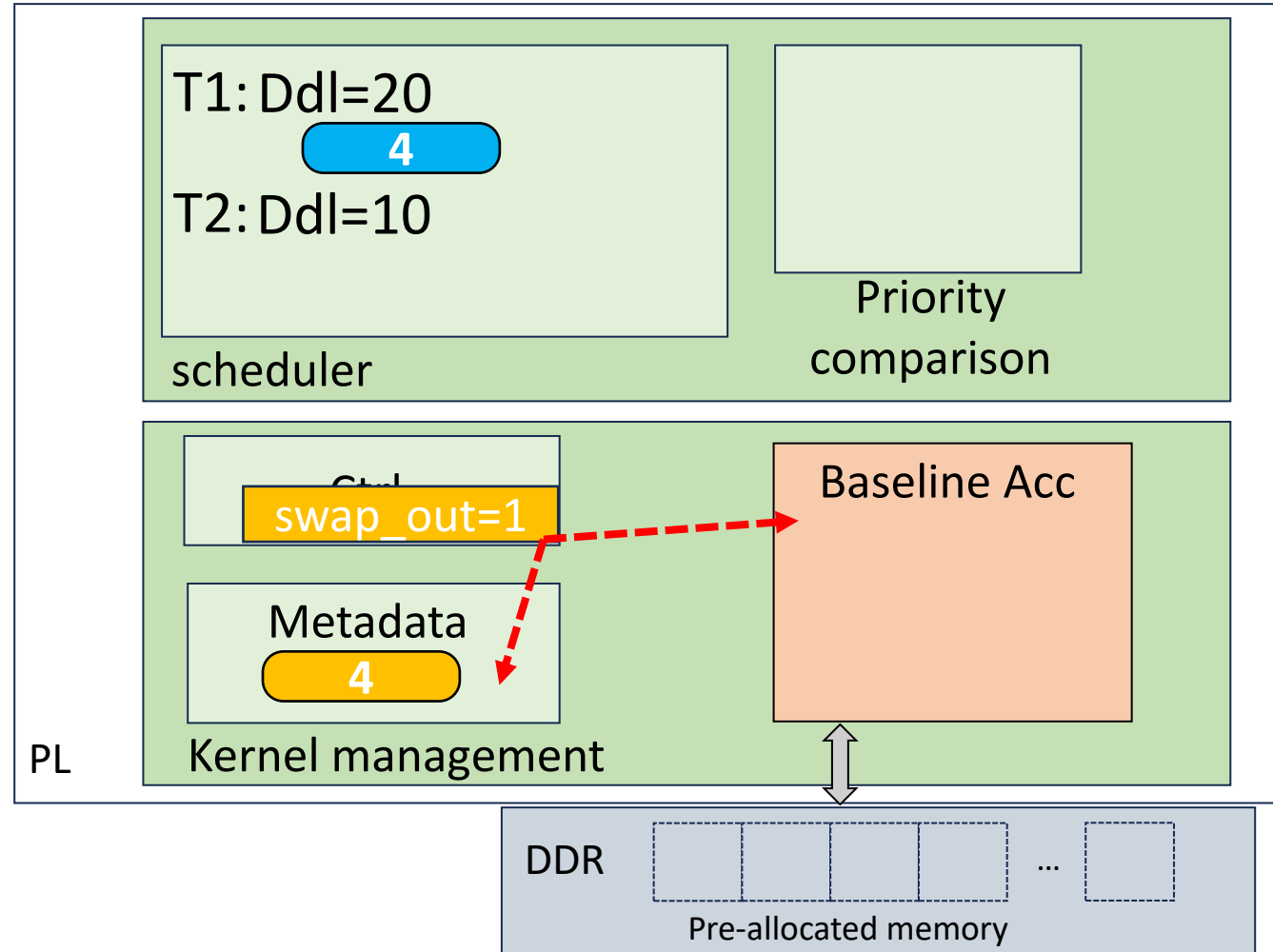
T=2:
Task 1 job 1 finishes its first segment



Scheduling & Preemption Support: Running Example

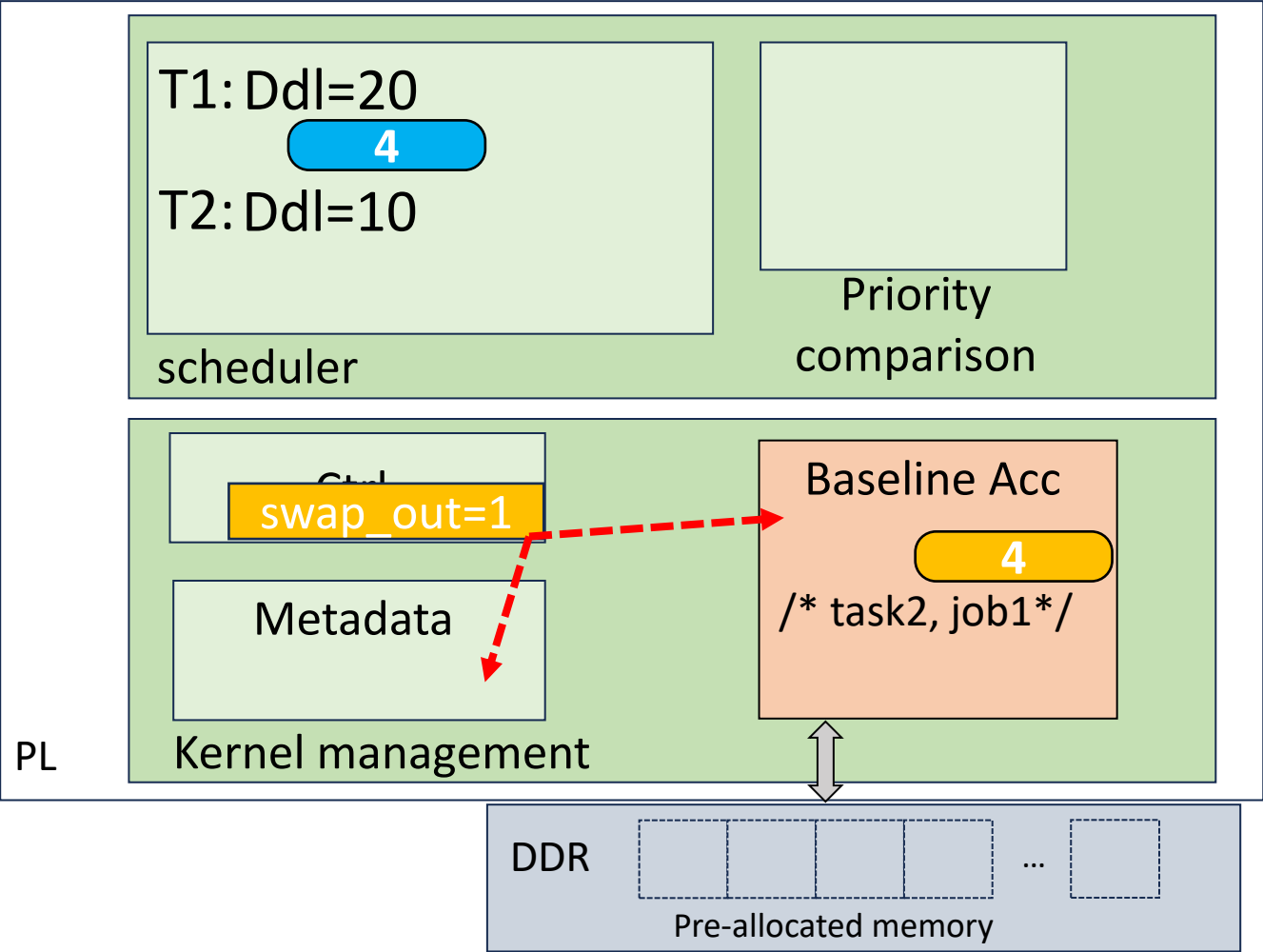
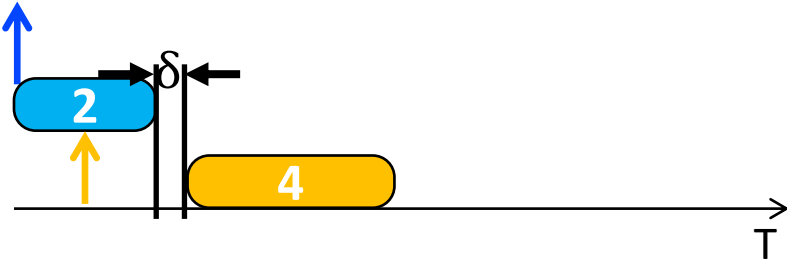
T=2:

Task 1 job 1 finishes its first segment



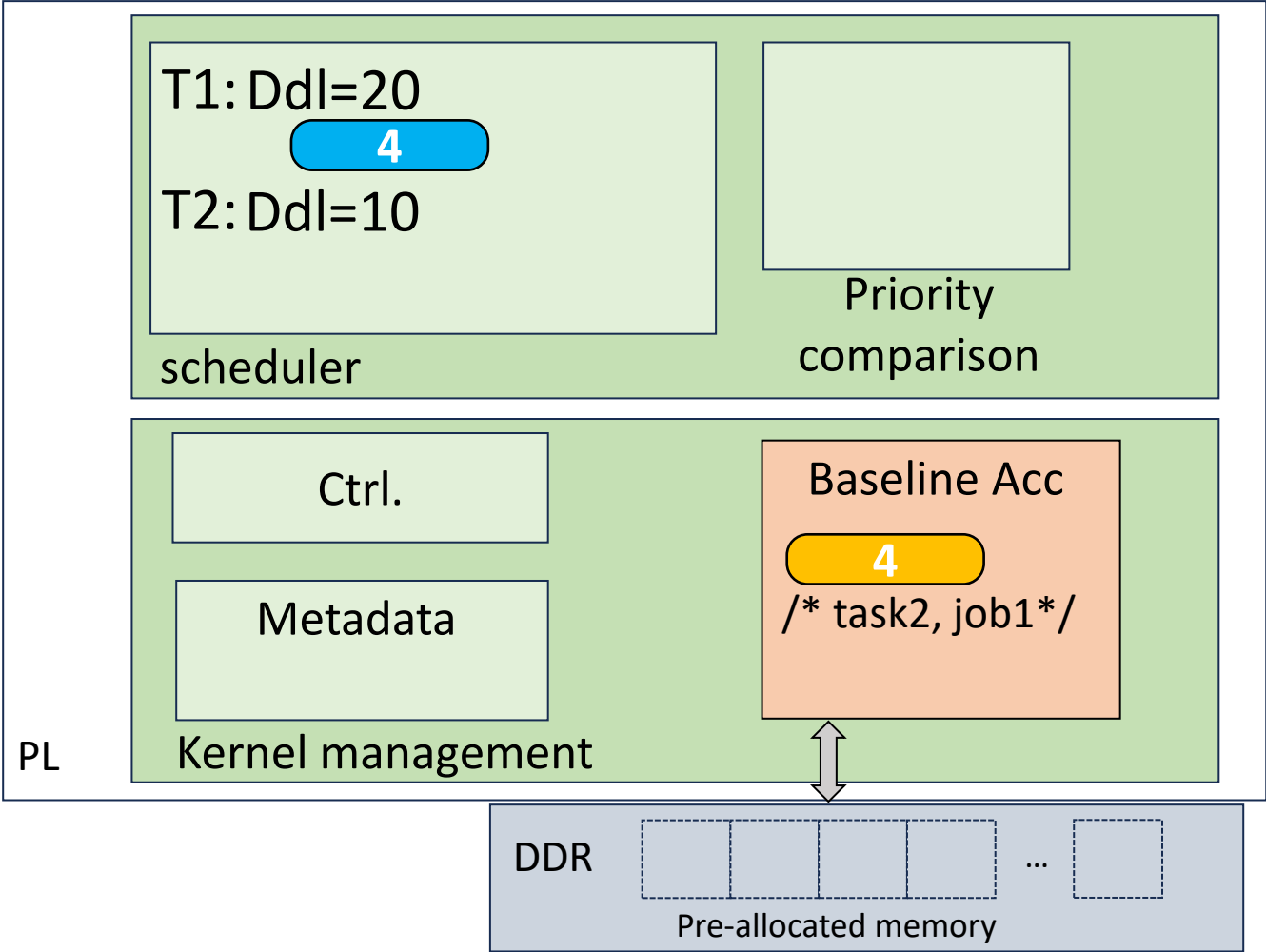
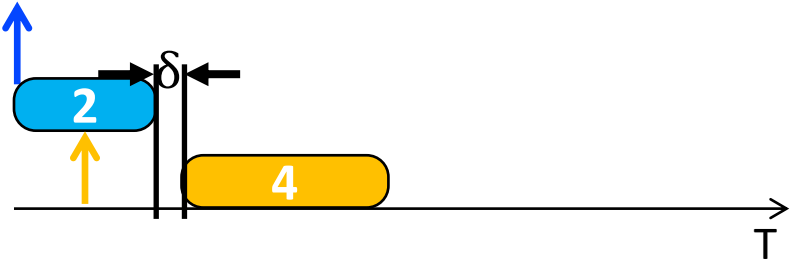
Scheduling & Preemption Support: Running Example

T=2:
Task 1 job 1 finishes its first
segment



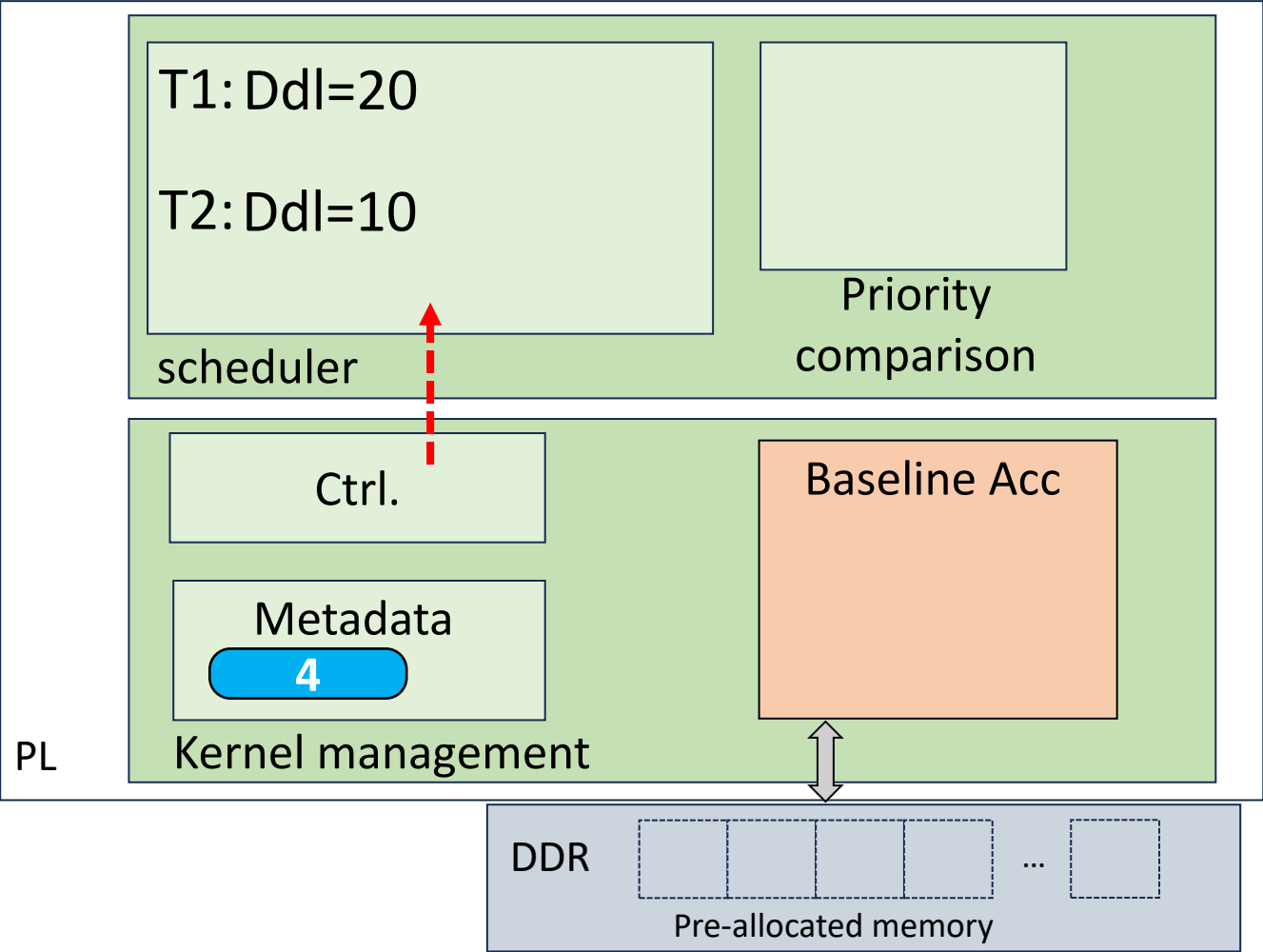
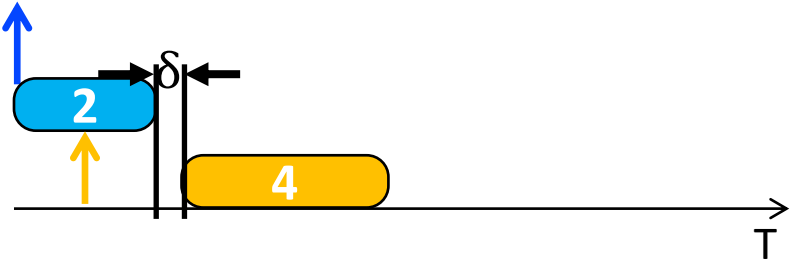
Scheduling & Preemption Support: Running Example

T=6:
Task 2 finishes



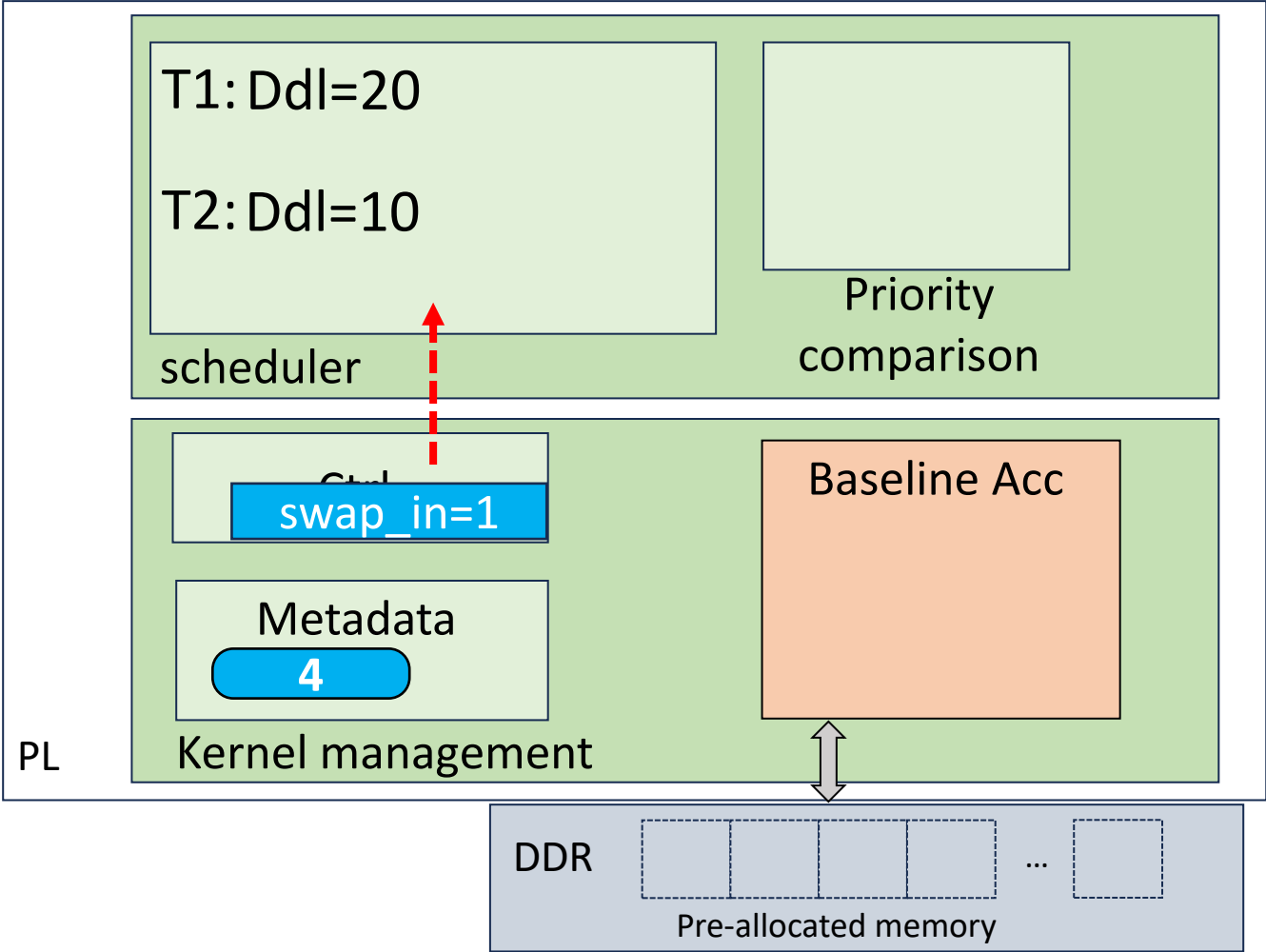
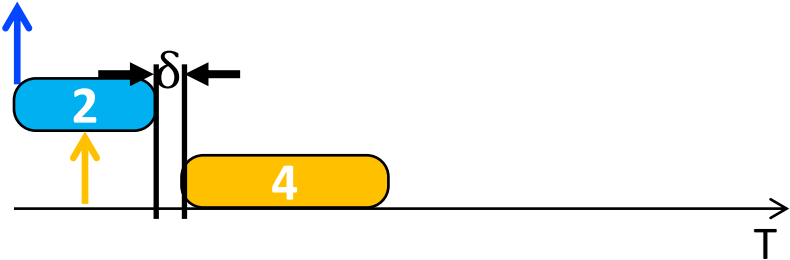
Scheduling & Preemption Support: Running Example

T=6:
Task 2 finishes



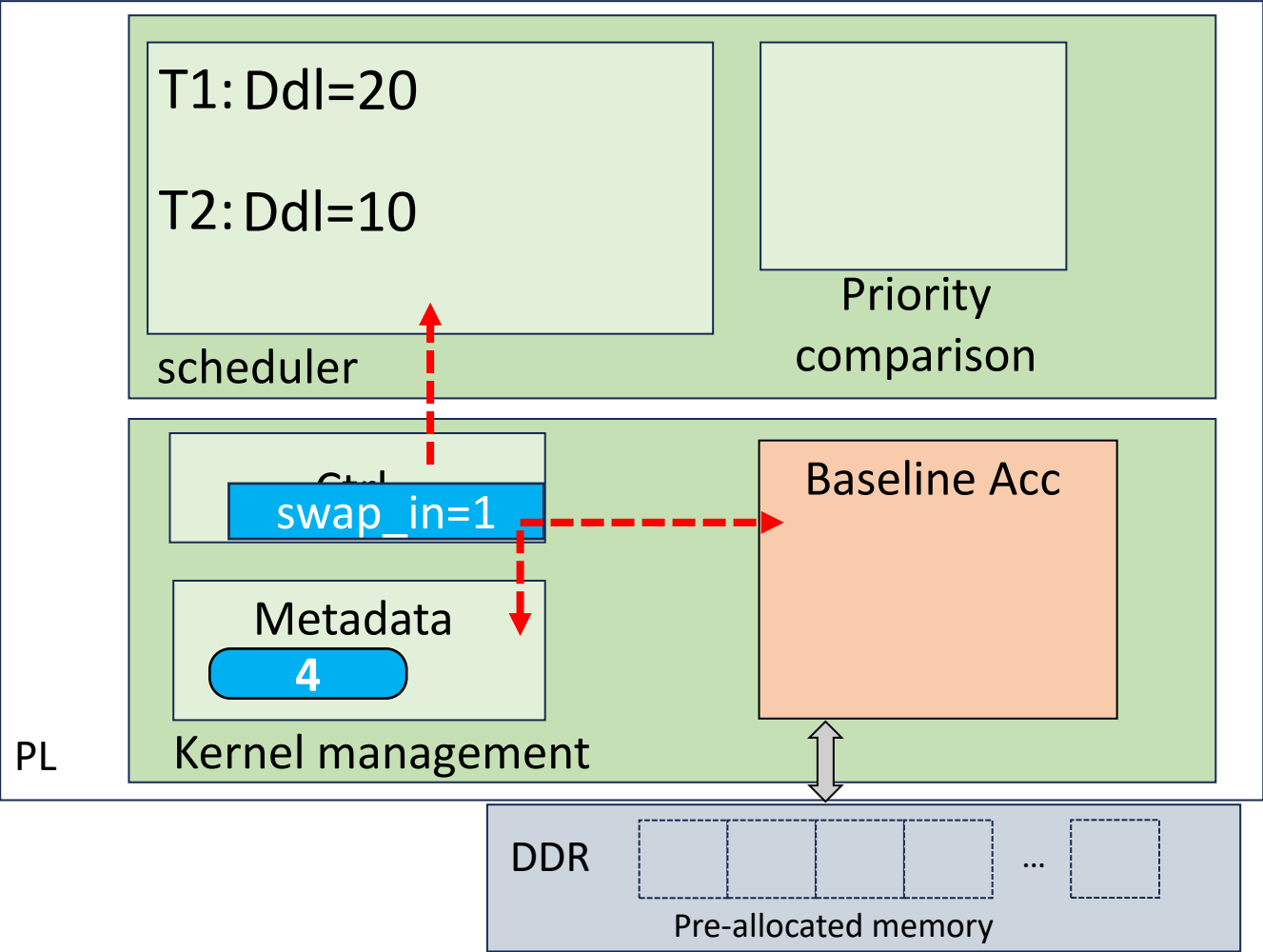
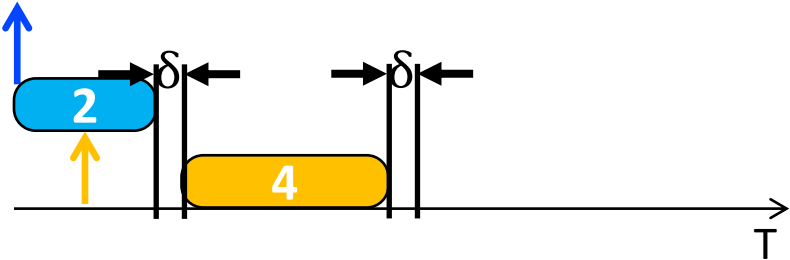
Scheduling & Preemption Support: Running Example

T=6:
Task 2 finishes



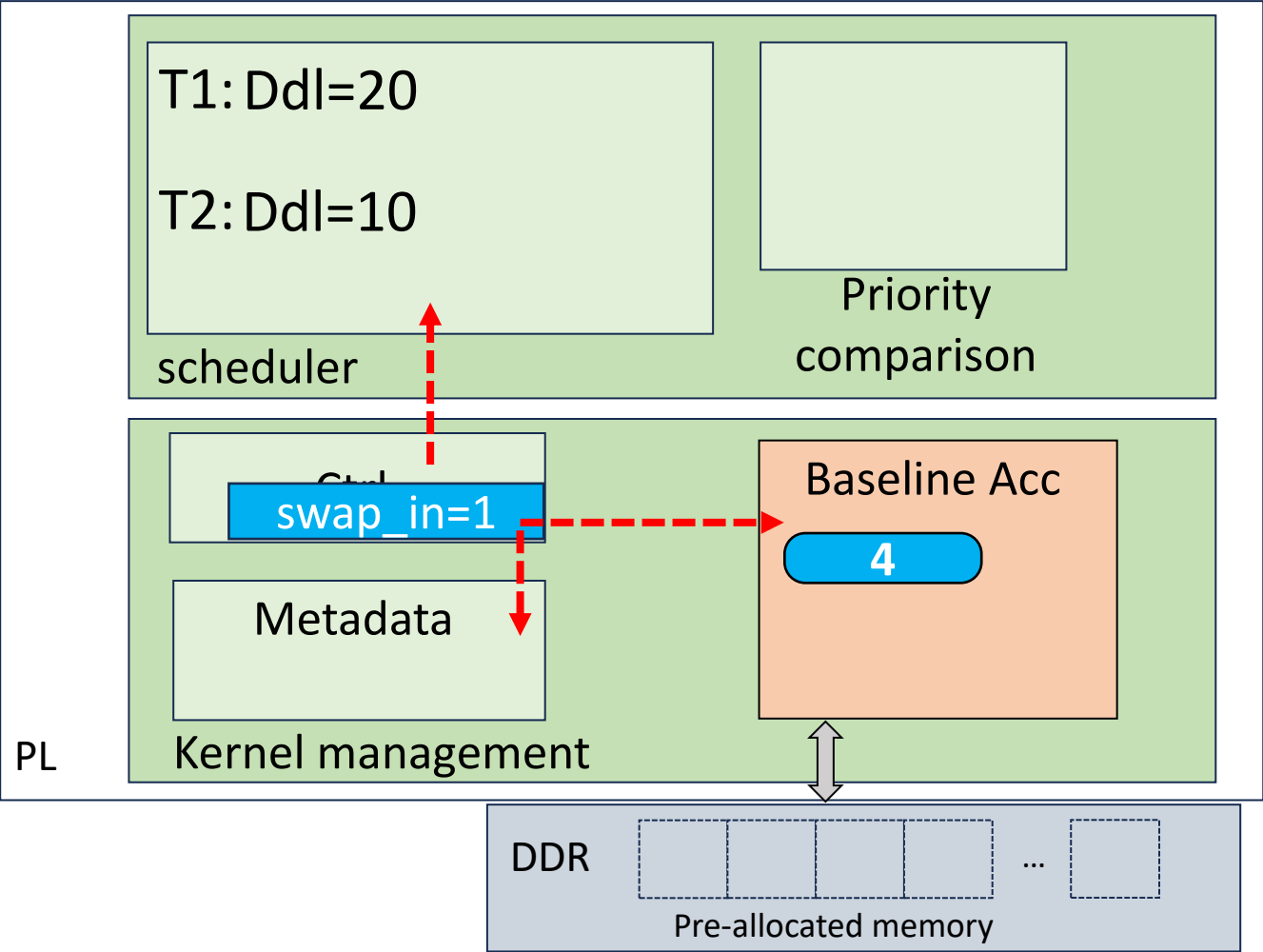
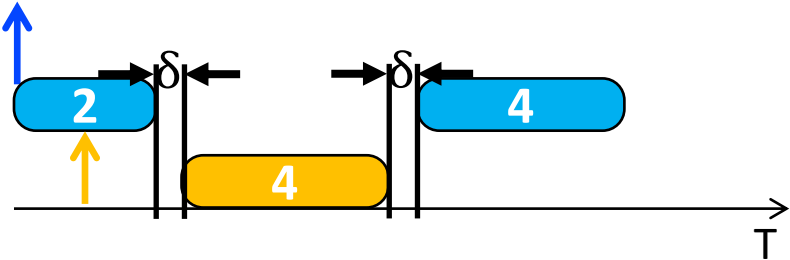
Scheduling & Preemption Support: Running Example

T=6:
Task 2 finishes



Scheduling & Preemption Support: Running Example

T=6:
Task 2 finishes

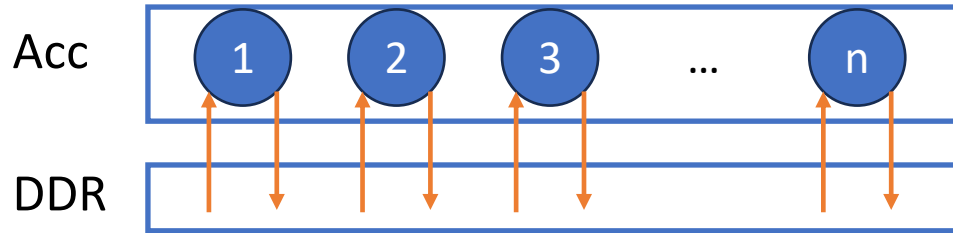


Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:

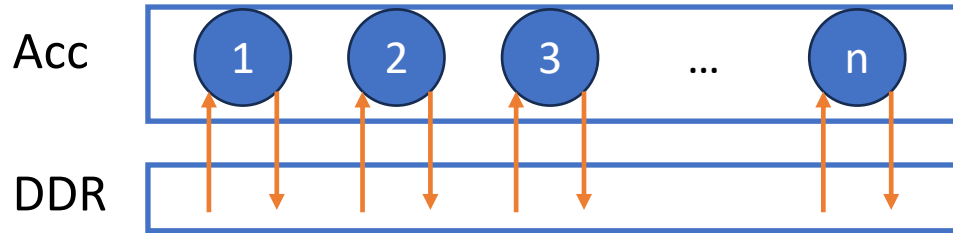
Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:



Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:

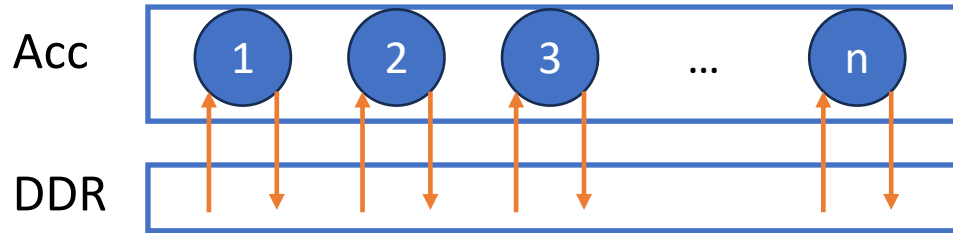


Sequentially launches DNN layers, layer
intermediate data is **on DDR**

--> No on-chip intermediate data between layers

Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:



Sequentially launches DNN layers, layer
intermediate data is **on DDR**

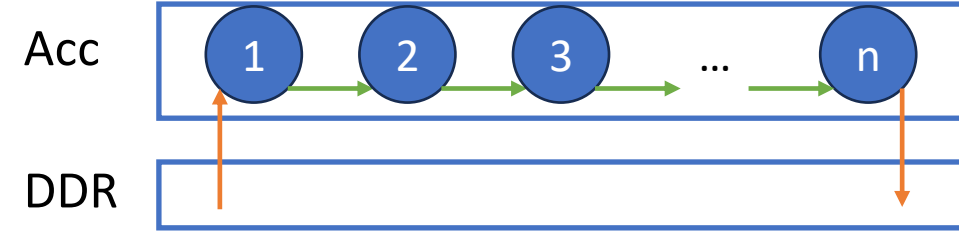
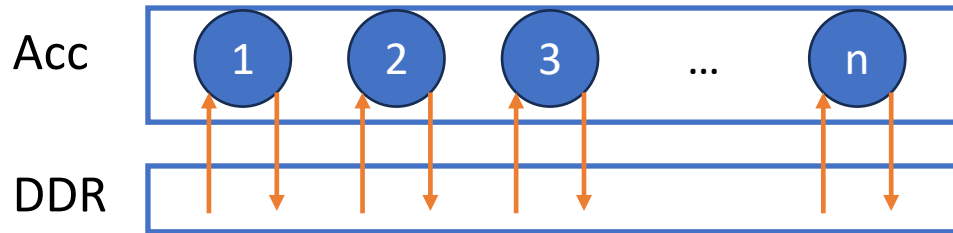
--> No on-chip intermediate data between layers

Preemption overhead:

- Scheduling operations
- Configuring the controller and accelerator

Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:



Sequentially launches DNN layers, layer
intermediate data is **on DDR**

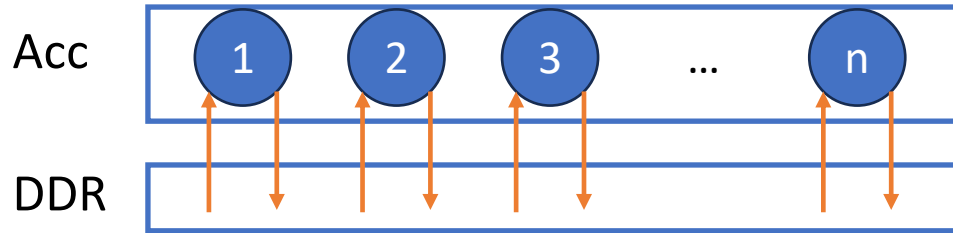
--> No on-chip intermediate data between layers

Preemption overhead:

- Scheduling operations
- Configuring the controller and accelerator

Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:

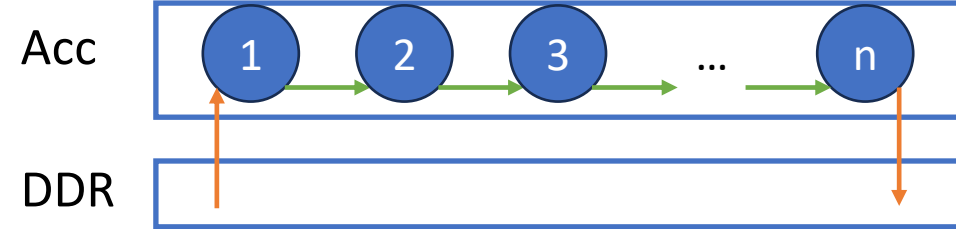


Sequentially launches DNN layers, layer intermediate data is **on DDR**

--> No on-chip intermediate data between layers

Preemption overhead:

- Scheduling operations
- Configuring the controller and accelerator

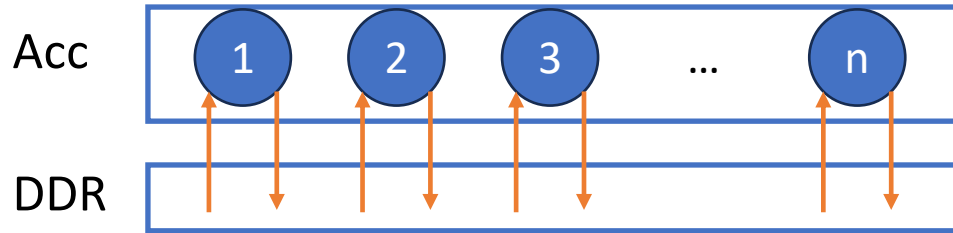


Sequentially launches DNN layers, layer intermediate data is **on chip**

--> on-chip forward data between layers

Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:

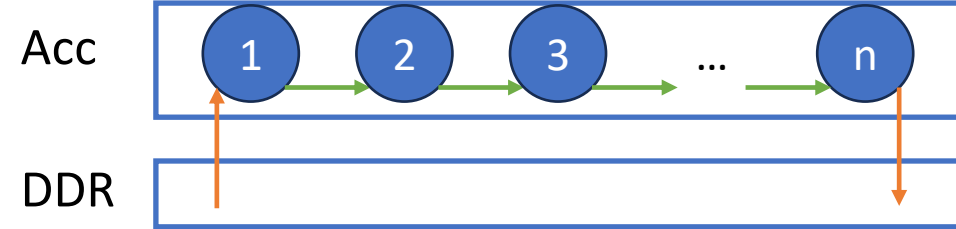


Sequentially launches DNN layers, layer intermediate data is **on DDR**

--> No on-chip intermediate data between layers

Preemption overhead:

- Scheduling operations
- Configuring the controller and accelerator



Sequentially launches DNN layers, layer intermediate data is **on chip**

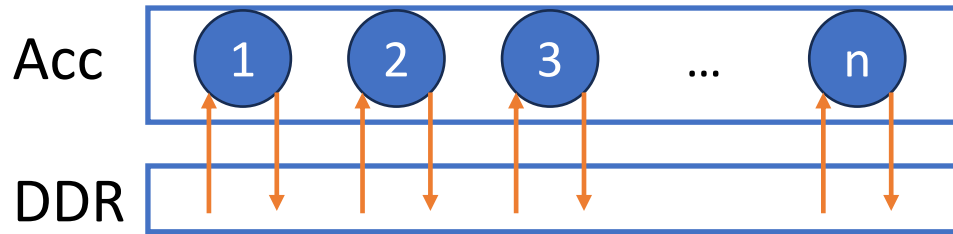
--> on-chip forward data between layers

Preemption overhead:

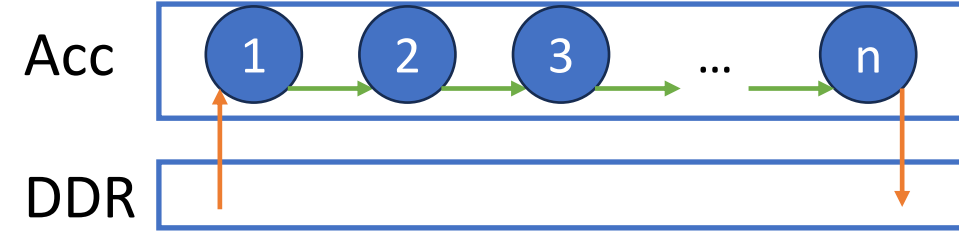
- Scheduling operations
- Configuring the controller and accelerator
- **Store/load the intermediate data**

Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:



layer intermediate data is **on DDR**

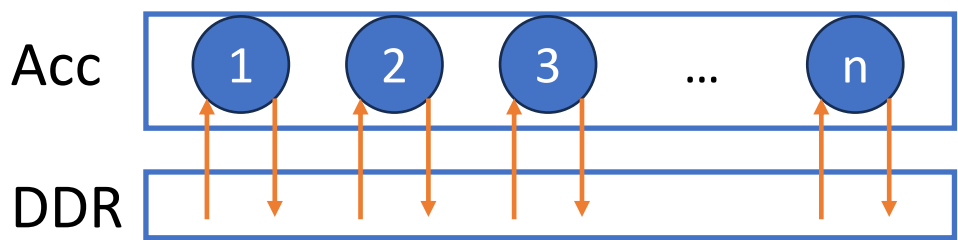


layer intermediate data is **on chip**

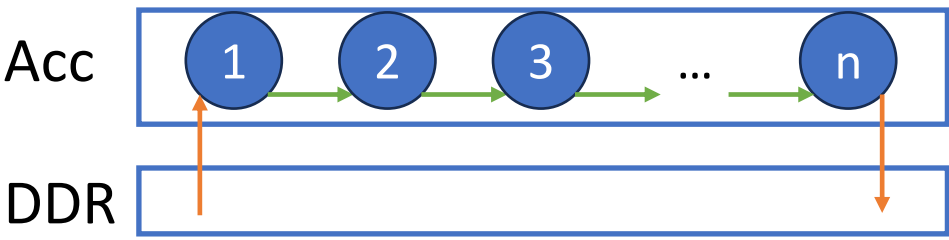
ART SW framework optimizes preemption overheads:

Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:

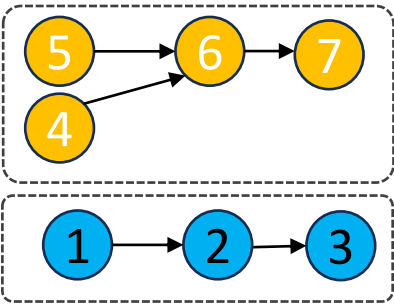


layer intermediate data is **on DDR**



layer intermediate data is **on chip**

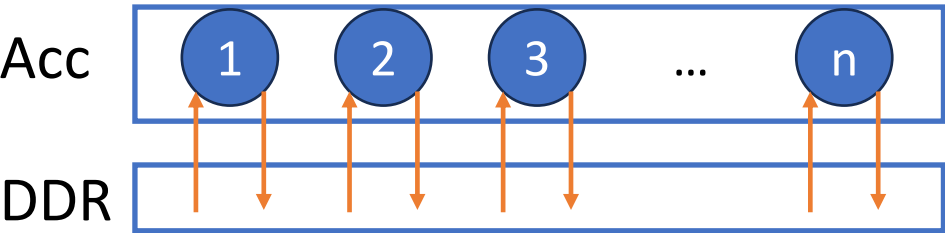
ART SW framework optimizes preemption overheads:



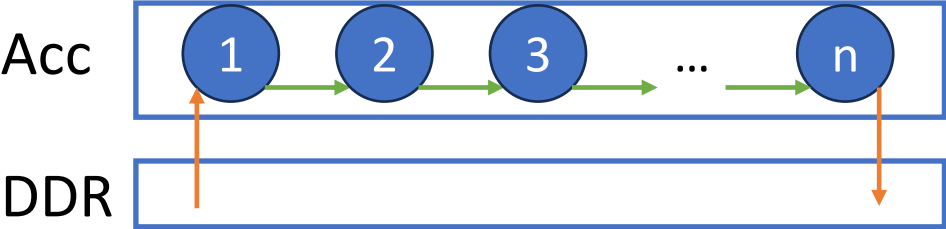
Task Configs

Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:

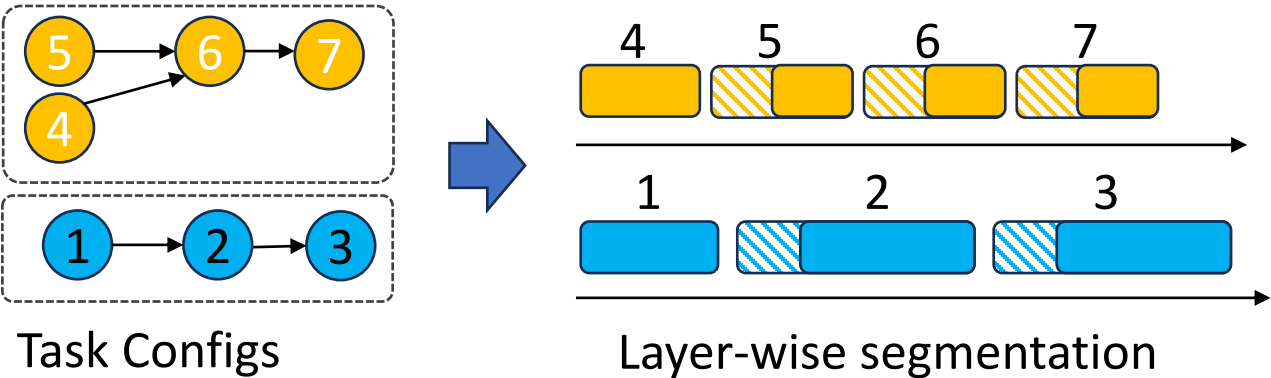


layer intermediate data is **on DDR**



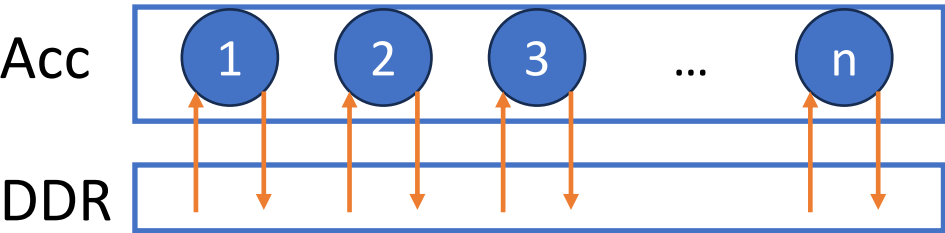
layer intermediate data is **on chip**

ART SW framework optimizes preemption overheads:

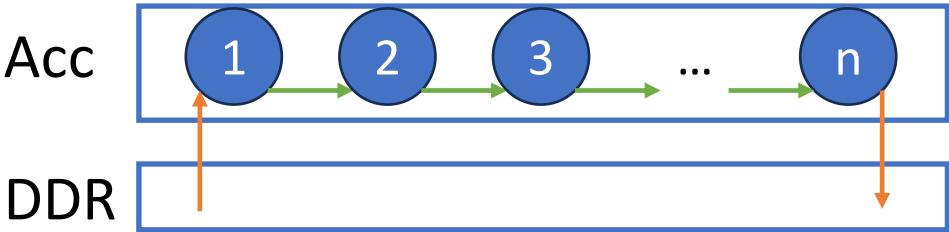


Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:

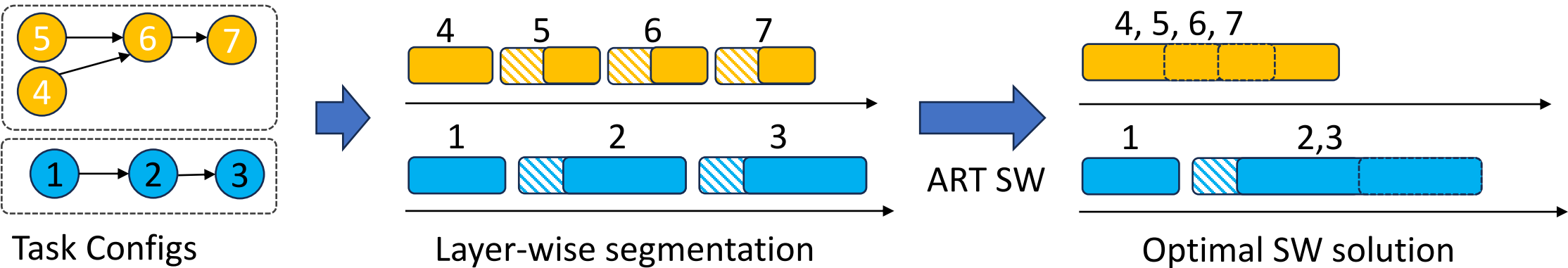


layer intermediate data is **on DDR**



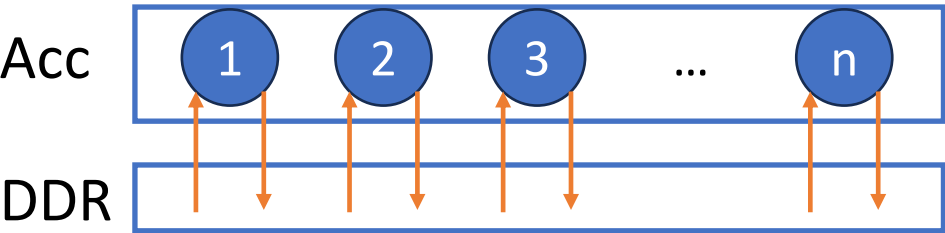
layer intermediate data is **on chip**

ART SW framework optimizes preemption overheads:

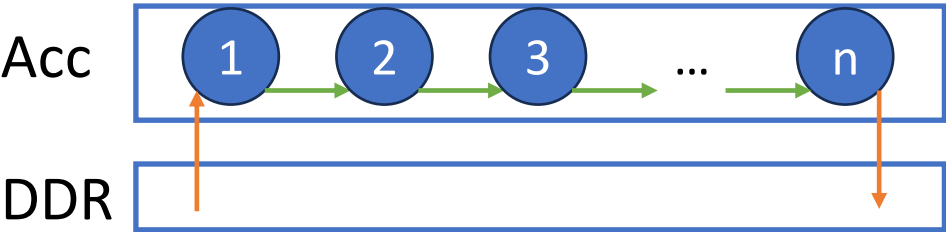


Preemption Overhead & ART SW Framework

Preemption overhead depends on Acc execution model:

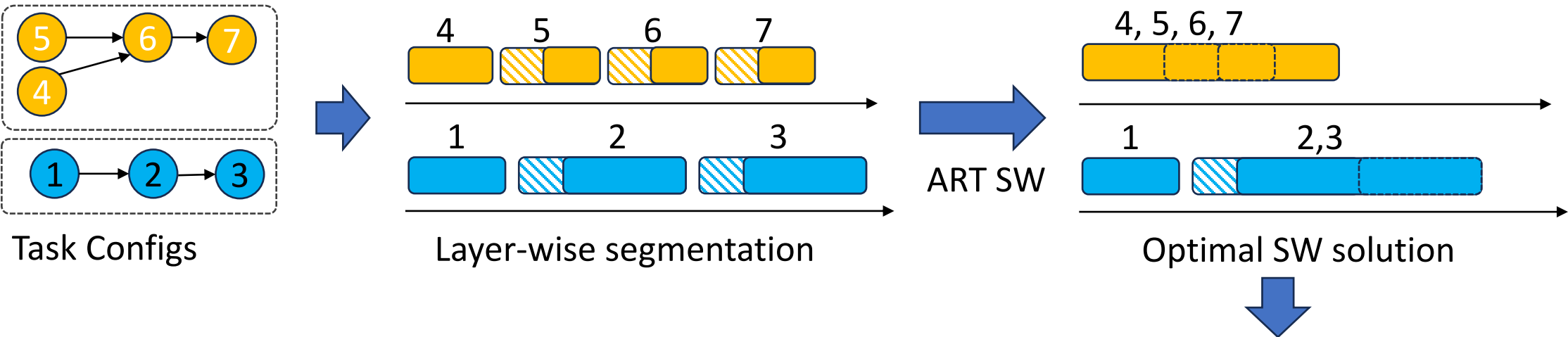


layer intermediate data is **on DDR**



layer intermediate data is **on chip**

ART SW framework optimizes preemption overheads:



Reduced worst-case execution time

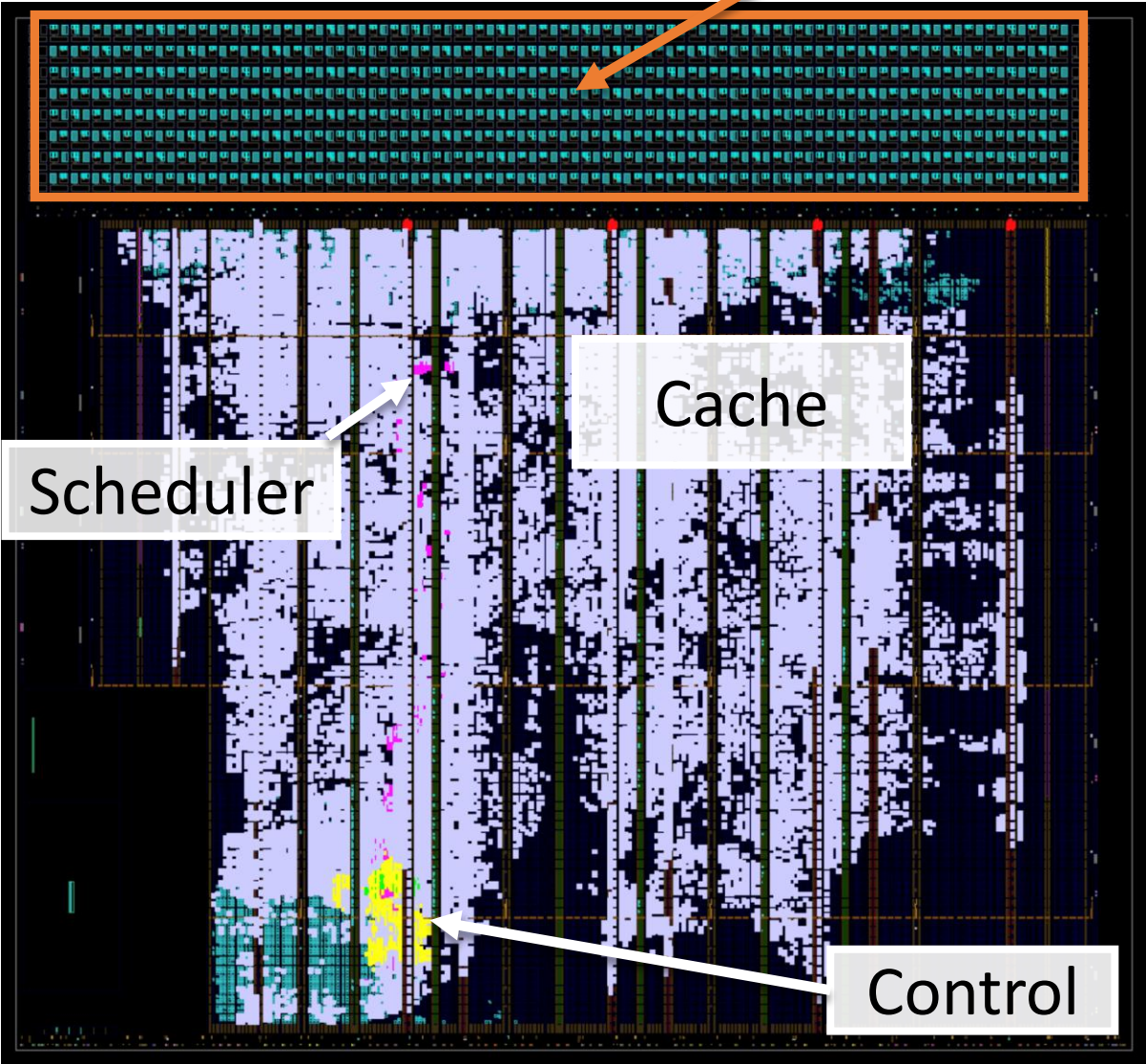
ART Evaluation

- We implement ART on the AMD VCK190 Evaluation board
- CHARM as the baseline accelerator
- ART is lightweight: <0.5 % resource utilization

Design	AIE	LUT	REG	BRAM	URAM	DSP
Baseline	384 96.00%	92549 10.29%	155526 8.64%	625 64.63%	384 82.94%	262 13.31%
ART	0 0%	2873 0.32%	2370 0.13%	4 0.41%	0 0%	0 0%
Total	384 96.00%	95422 10.60%	157896 8.77%	629 65.04%	384 82.94%	262 13.31%

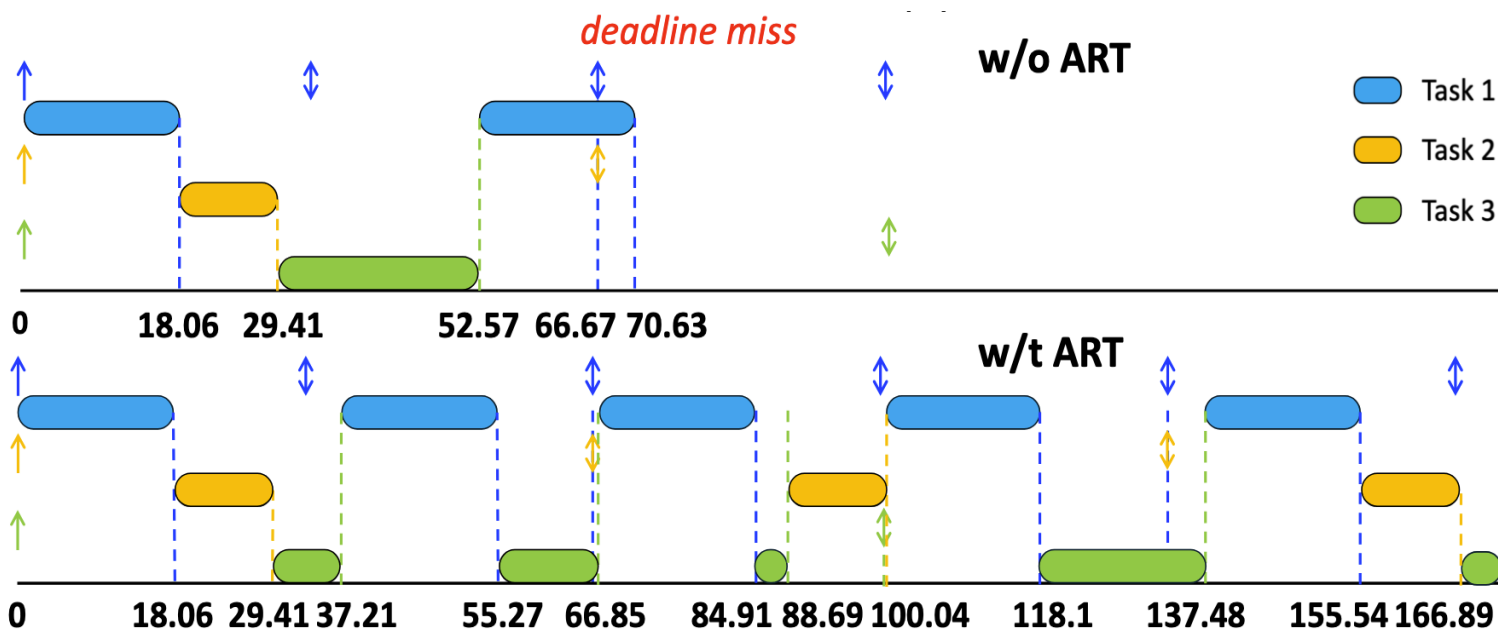


PE array(AI engines)



ART Evaluation

- ART can satisfy schedulability whereas the baseline accelerator cannot

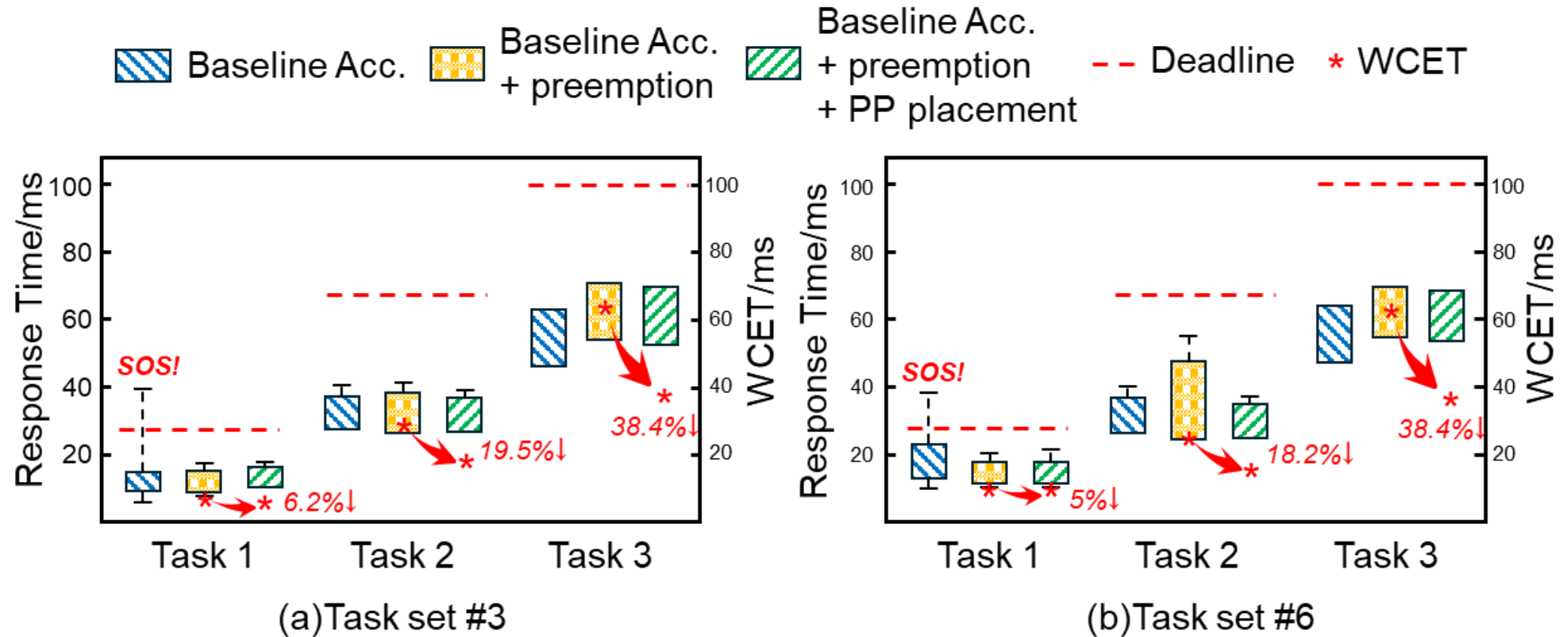


# of Scenario	w/o ART	w/t ART
1	×	✓
2	×	✓
3	×	✓
4	✓	✓
5	×	✓
6	×	✓

- Before : QoS 17% , After: QoS 100%
- Overhead: <0.1% degradation in throughput

ART Evaluation

- Preemption mechanism improves worst-case response time
- ART SW framework optimizes both the worst-case response time and execution time



Thank You

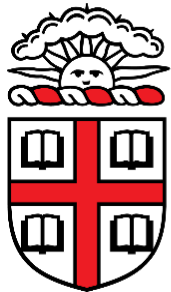
ART: Customizing Accelerators for DNN-Enabled Real-Time Safety-Critical Systems

Shixin Ji*, Xingzhen Chen*, Wei Zhang*, Zhuoping Yang*, Jinming Zhuang*, Sarah Schultz*,
Yukai Song ‡, Jingtong Hu ‡, Alex K. Jones§, Zheng Dong†, Peipei Zhou*

Brown University*; Wayne State University† ; University of Pittsburgh‡ ; Syracuse University§

shixin_ji@brown.edu
peipei_zhou@brown.edu

<https://peipeizhou-eecs.github.io/>



BROWN



WAYNE STATE
UNIVERSITY



National
Science
Foundation



University of
Pittsburgh®



Syracuse University

AMD
XILINX