University of Cantabria, Spain
Software Engineering and Real-Time Group

Calculating Latencies in an Engine Management System Using Response Time Analysis with MAST

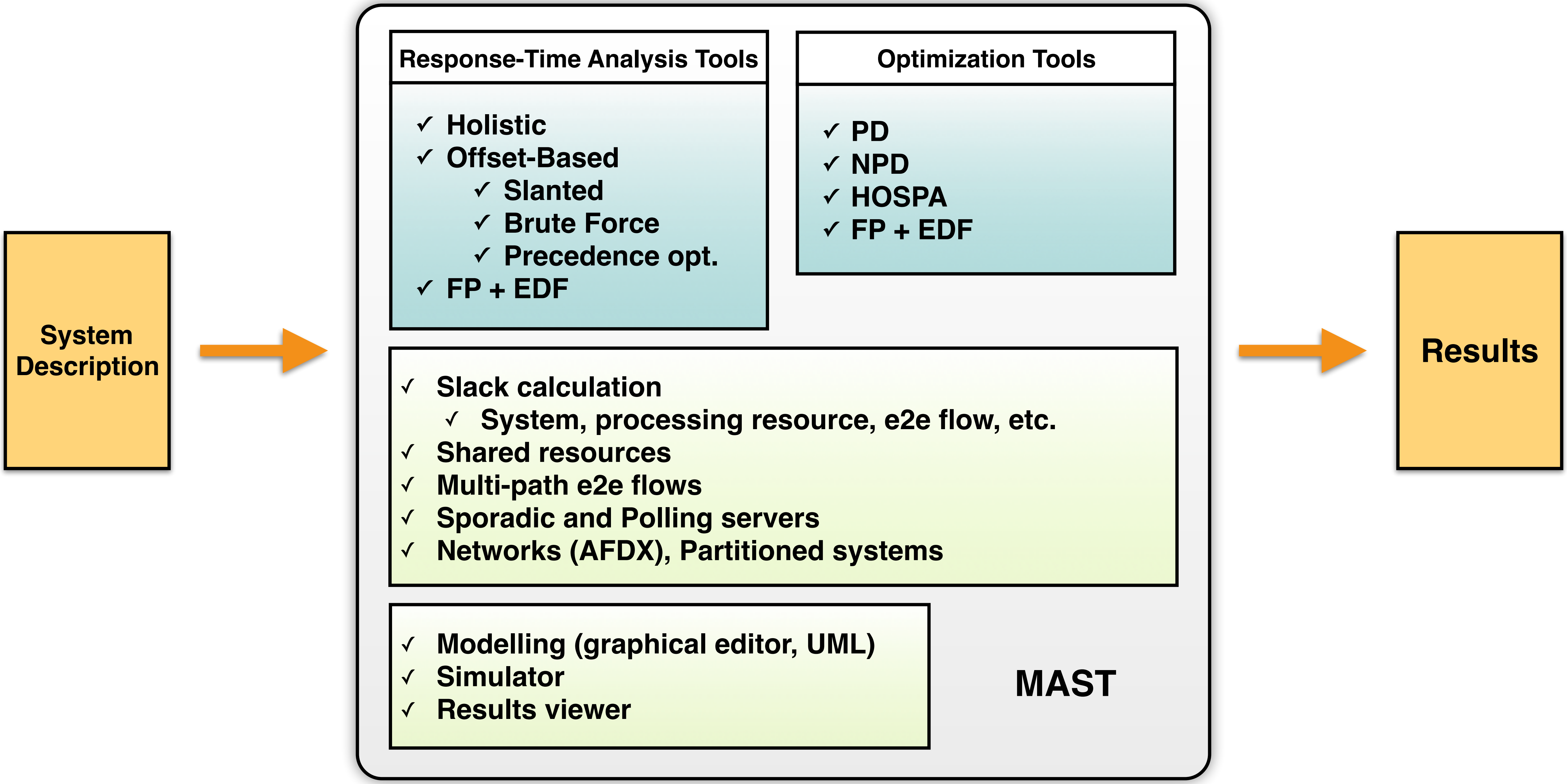
Juan M. Rivas
J. Javier Gutiérrez
J. Medina
Michael González Harbour

FMTV 2016

Toulouse, July 2016

MAST

- To model, analyze and optimize hard real-time systems
- Open Source (Ada). Available at www.mast.unican.es



System Description

Response-Time Analysis Tools

- ✓ Holistic
- ✓ Offset-Based
  - ✓ Slanted
  - ✓ Brute Force
  - ✓ Precedence opt.
- ✓ FP + EDF

Optimization Tools

- ✓ PD
- ✓ NPD
- ✓ HOSPA
- ✓ FP + EDF

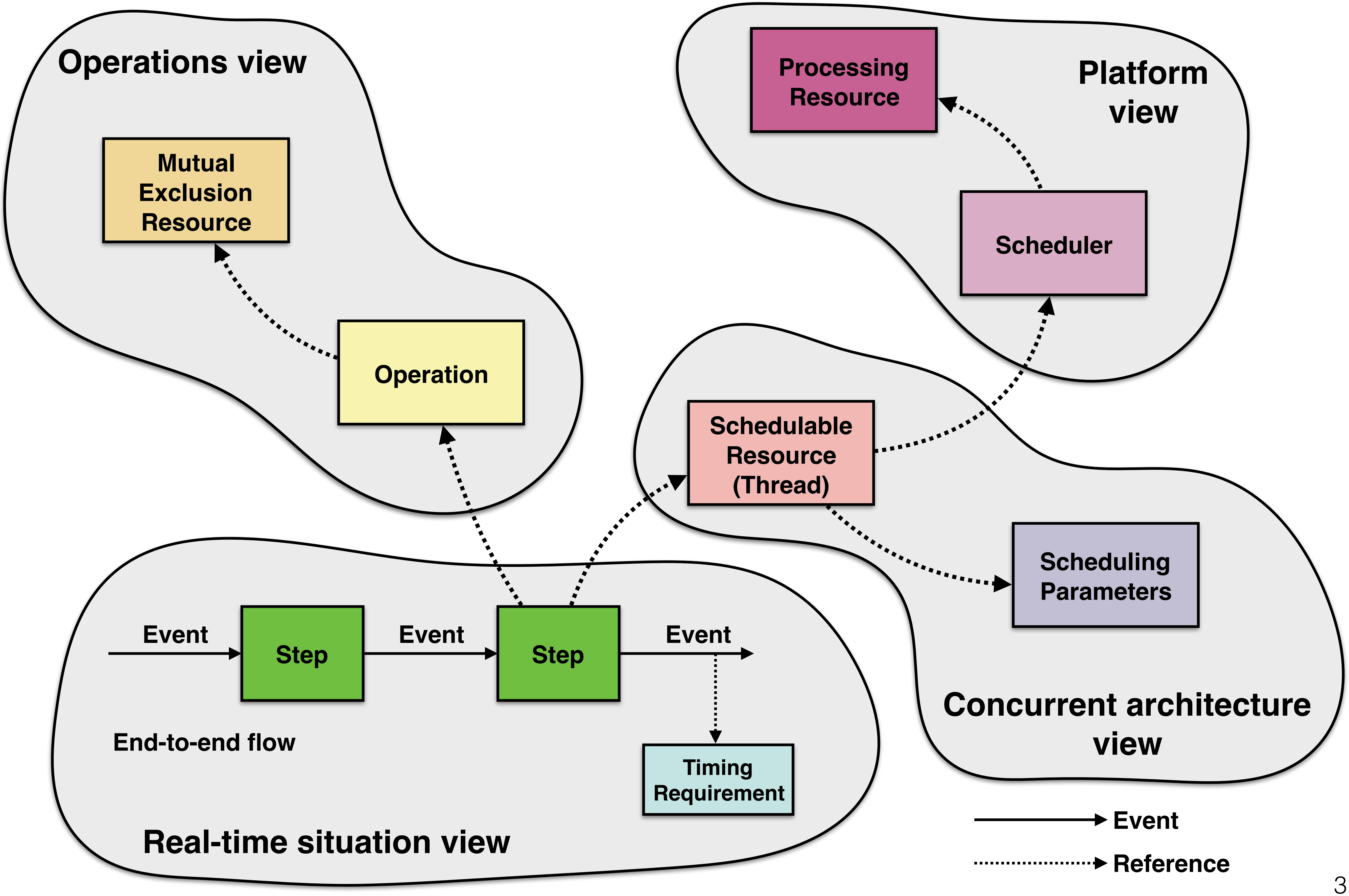
- ✓ Slack calculation
  - ✓ System, processing resource, e2e flow, etc.
- ✓ Shared resources
- ✓ Multi-path e2e flows
- ✓ Sporadic and Polling servers
- ✓ Networks (AFDX), Partitioned systems

- ✓ Modelling (graphical editor, UML)
- ✓ Simulator
- ✓ Results viewer

MAST

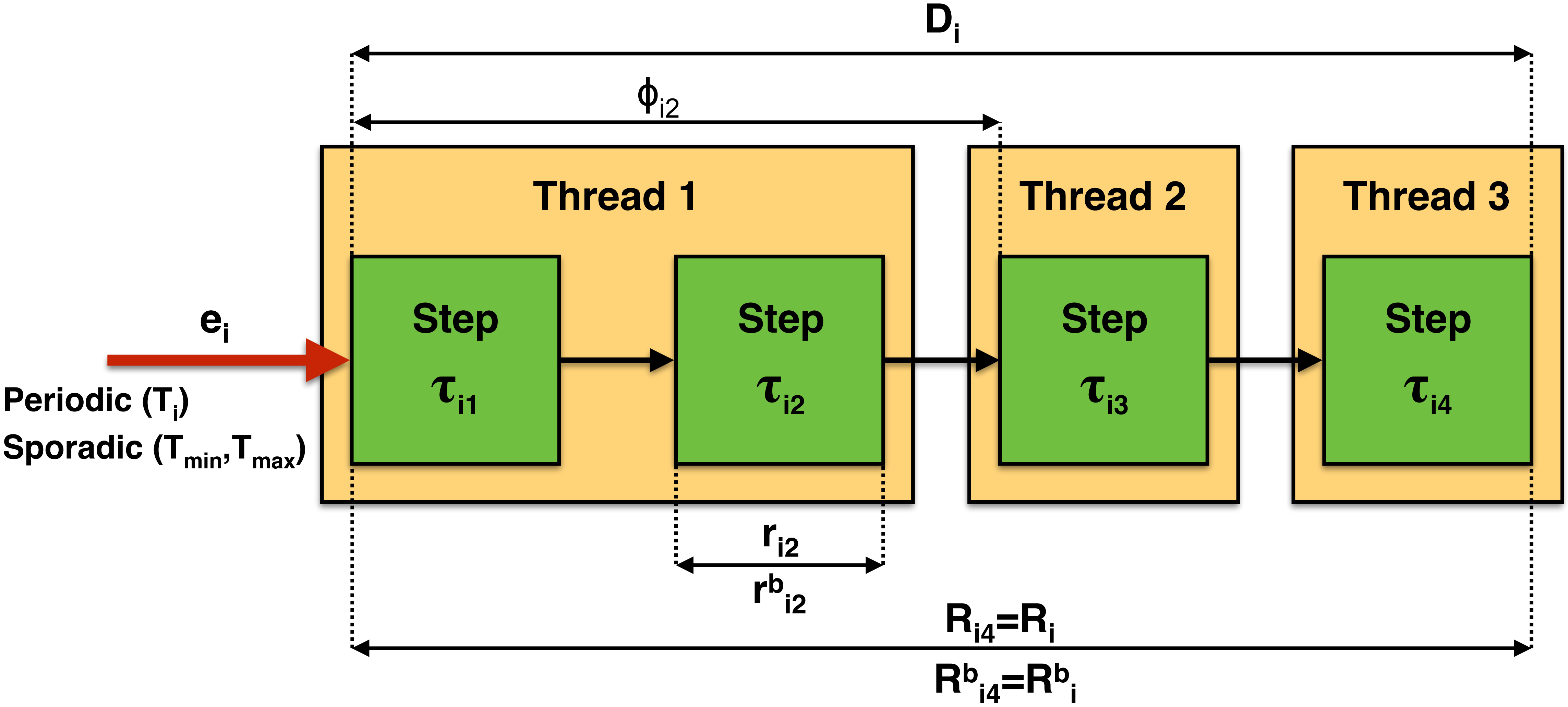
Results

MAST Model


Operations view
Mutual Exclusion Resource
Operation
Platform view
Processing Resource
Scheduler
Schedulable Resource (Thread)
Scheduling Parameters
Concurrent architecture view
Event
Step
Event
Step
Event
Timing Requirement
End-to-end flow
Real-time situation view
Event
Reference

MAST Model for analysis

- End-to-end flows, aligned with OMG MARTE



- Steps: Worst-case execution time (Cij), Best-case execution time (Cbij)
- Threads: Priority (Prioij), Processor (Procij), Preemptive/Non-preemptive
- Results from response time analysis:
  - Global response time: worst-case (R_i), best-case (R^b_i)
  - Local response time: worst-case (r_{ij}), best-case (r^b_{ij})

Amalthea to MAST transformation

Amalthea Tasks

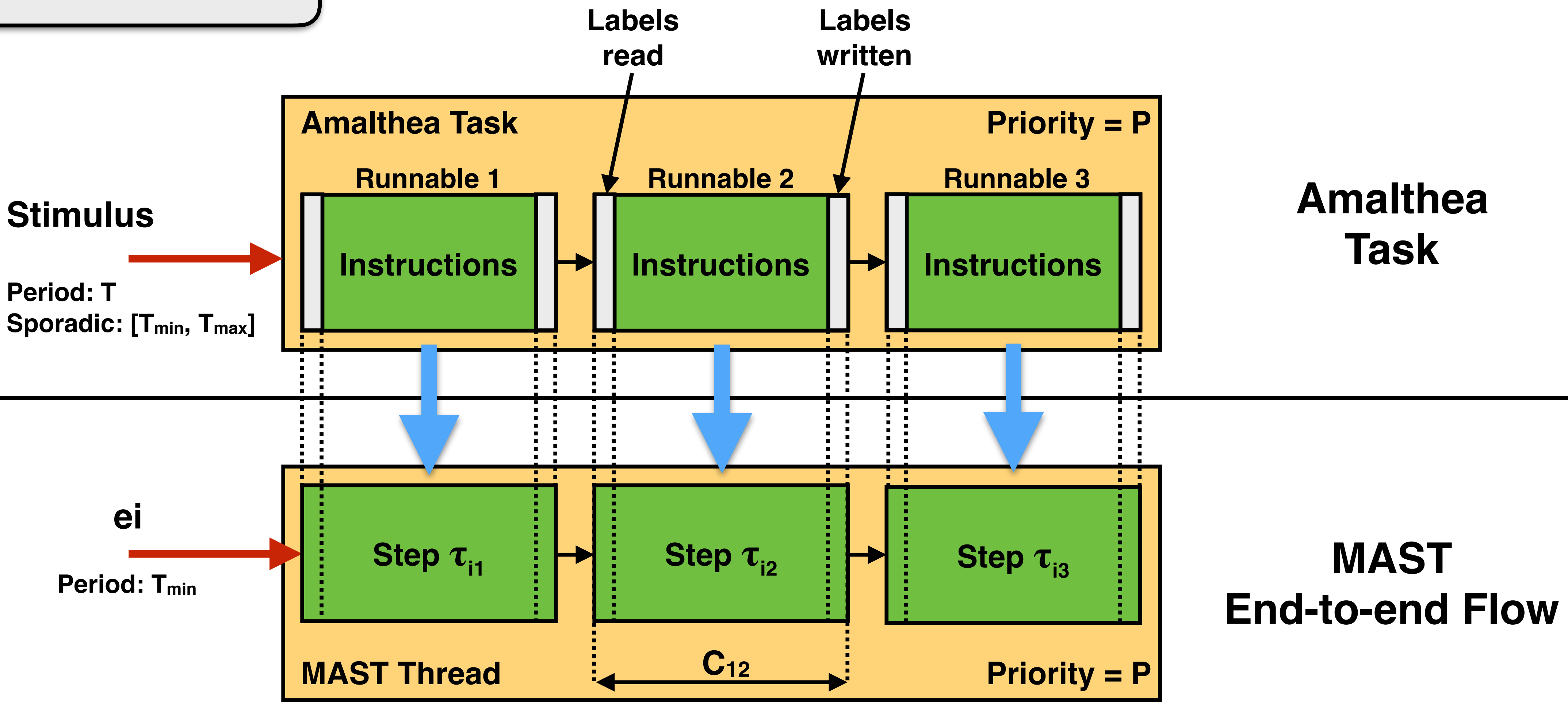
- 21 Tasks
- Statically assigned to a core
- Fixed Priority: preemptive/ cooperative
- Released by stimuli: periodic/ sporadic (arbitrary phasing)
- D=T
- Series of Runnables

Amalthea Runnables

- 1250 Runnables
- Read labels (memory)
- Instructions: constant/deviation
- Write labels (memory)

Labels

- 10000 Labels
- Mapped to GRAM/LRAM
- Local RAM = 1 cycle
- Non-Local RAM = 9 cycles

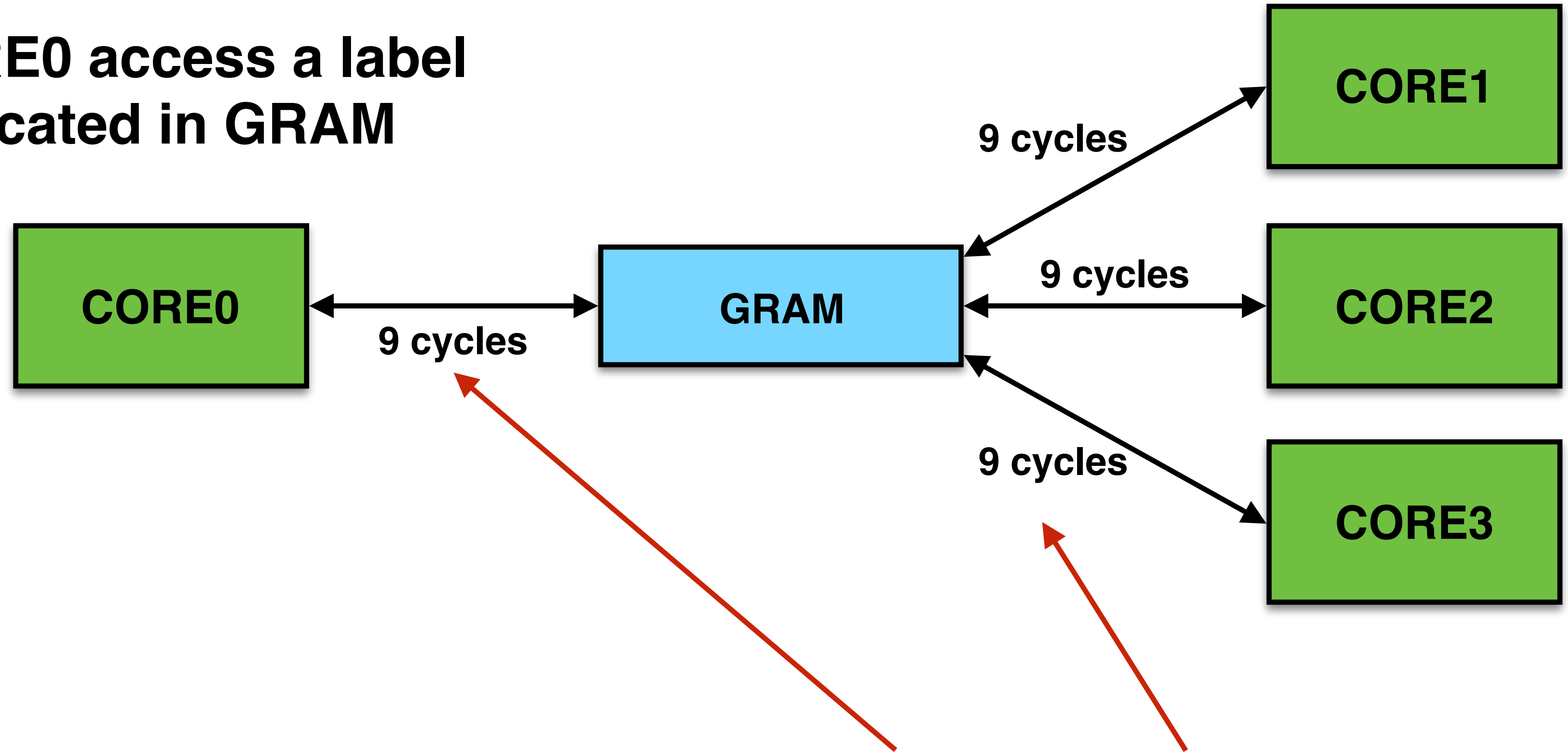


WCET of steps = instructions + worst-case memory accesses

Memory Accesses

- Modeled as execution time added to the steps
- Worst-case cost of accessing the labels
  - Assumes all cores accessing the same memory at the same time
- Best-case cost of accessing the labels
  - Assumes no other core is accessing the same memory
- Example with all labels in GRAM (Challenge 2):

CORE0 access a label located in GRAM



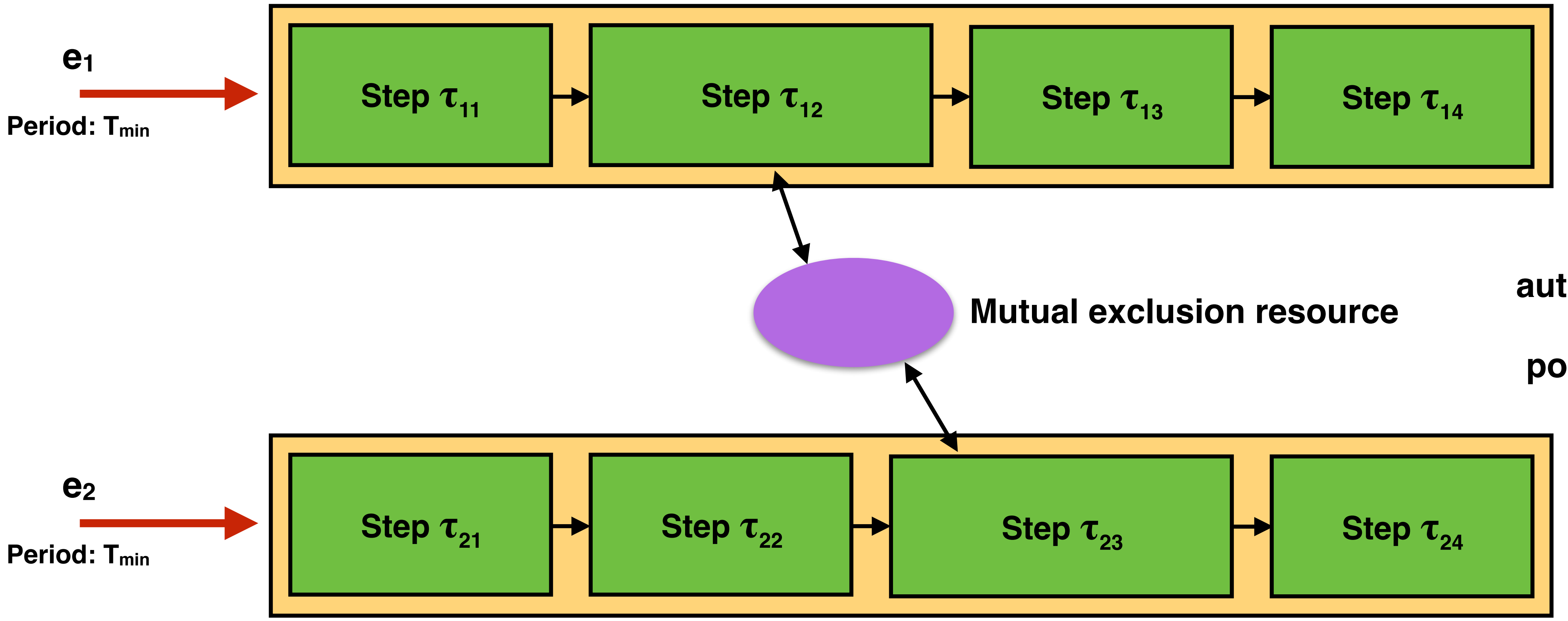
Worst-case cost of accessing a label = 9 cycles + 3*9 cycles = 36 cycles

Best-case cost of accessing a label = 9 cycles

Cooperative Scheduling

- Cooperative tasks can be preempted by higher priority…
  - Preemptive tasks at any moment
  - Cooperative tasks at runnable borders
- Cooperative tasks suffer a blocking equal to the longest runnable of lower priority
- We can model the blocking in MAST with a dummy shared resource
  - Accessed by the longest cooperative runnables

From a cooperative Amalthea Task



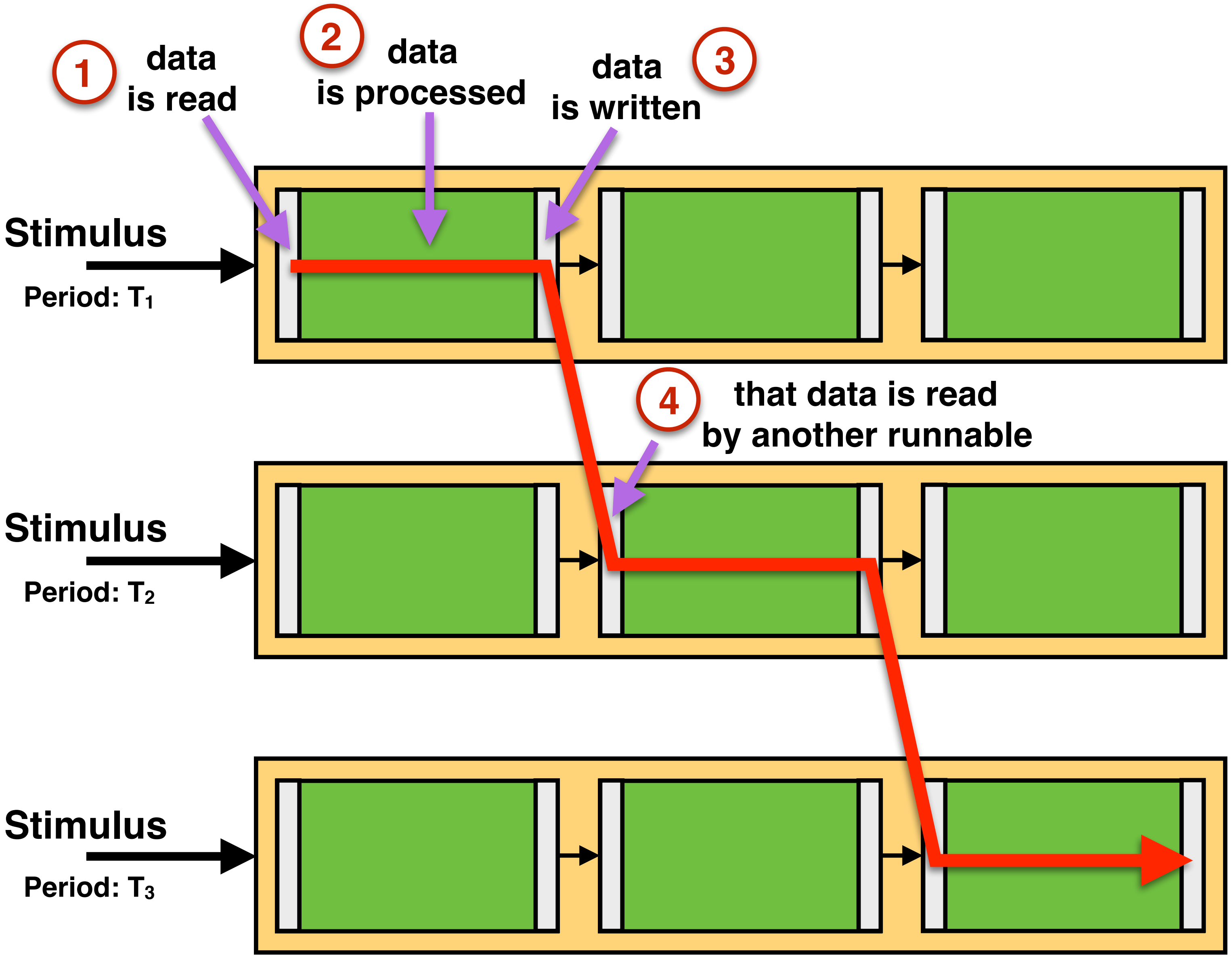
MAST automatically finds the longest possible blocking

From a cooperative Amalthea Task

Event-chain analysis (1/3)

- **Latency model of data traversing non-consecutive runnables**
  1. **Runnables from different Amalthea tasks: EffectChain_2 and EffectChain_3**
  2. **Runnables from the same Amalthea task: EffectChain_1**

- **Runnables from different Amalthea tasks:**

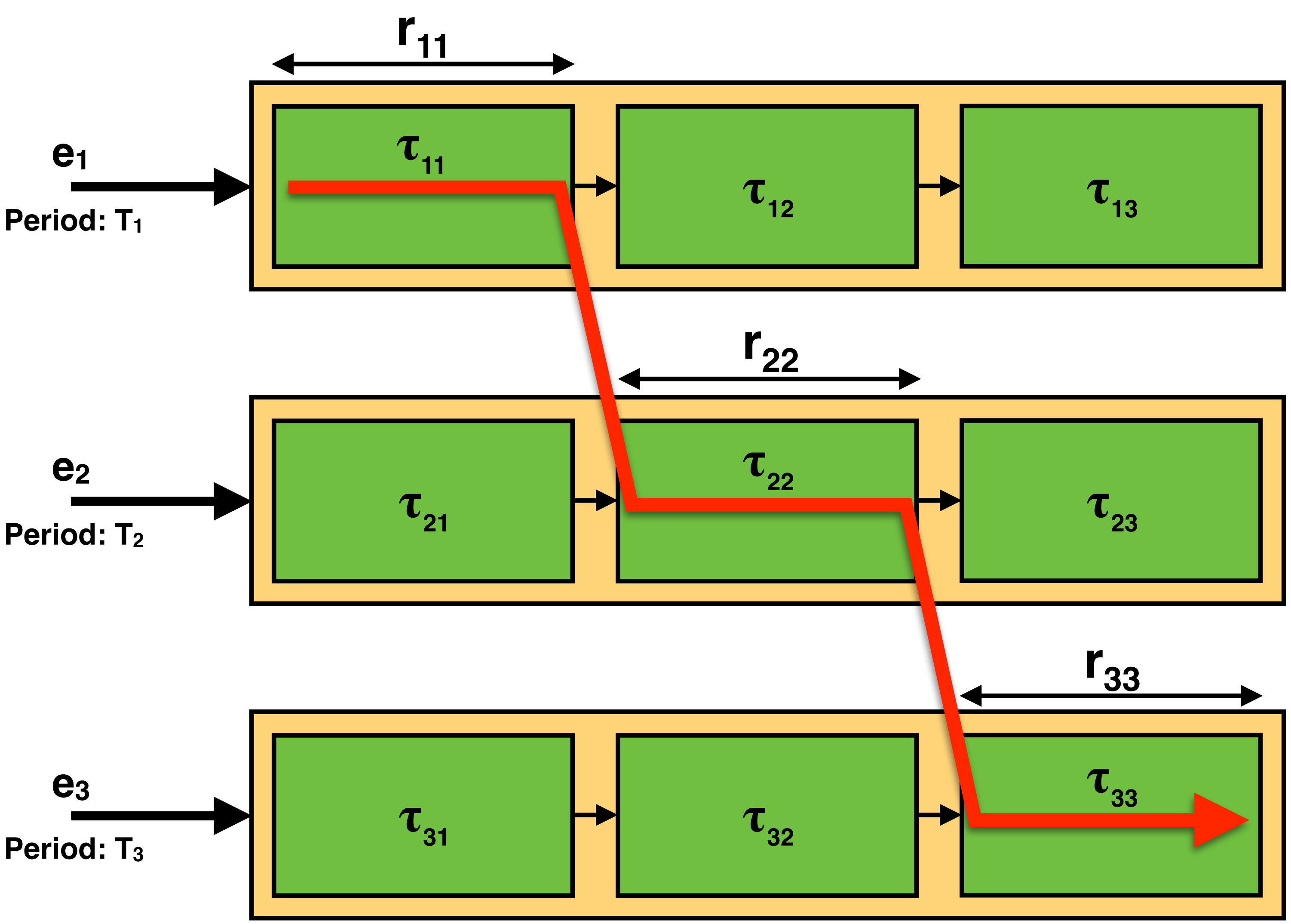


Event-chain analysis (2/3)

- **Latency model of data traversing non-consecutive runnables**
  1. **Runnables from different Amalthea tasks: EffectChain_2 and EffectChain_3**
  2. **Runnables from the same Amalthea task: EffectChain_1**

- **Runnables from different Amalthea tasks:**

MAST equivalent model



Worst-case latency assumes labels are written just after they are going to be read

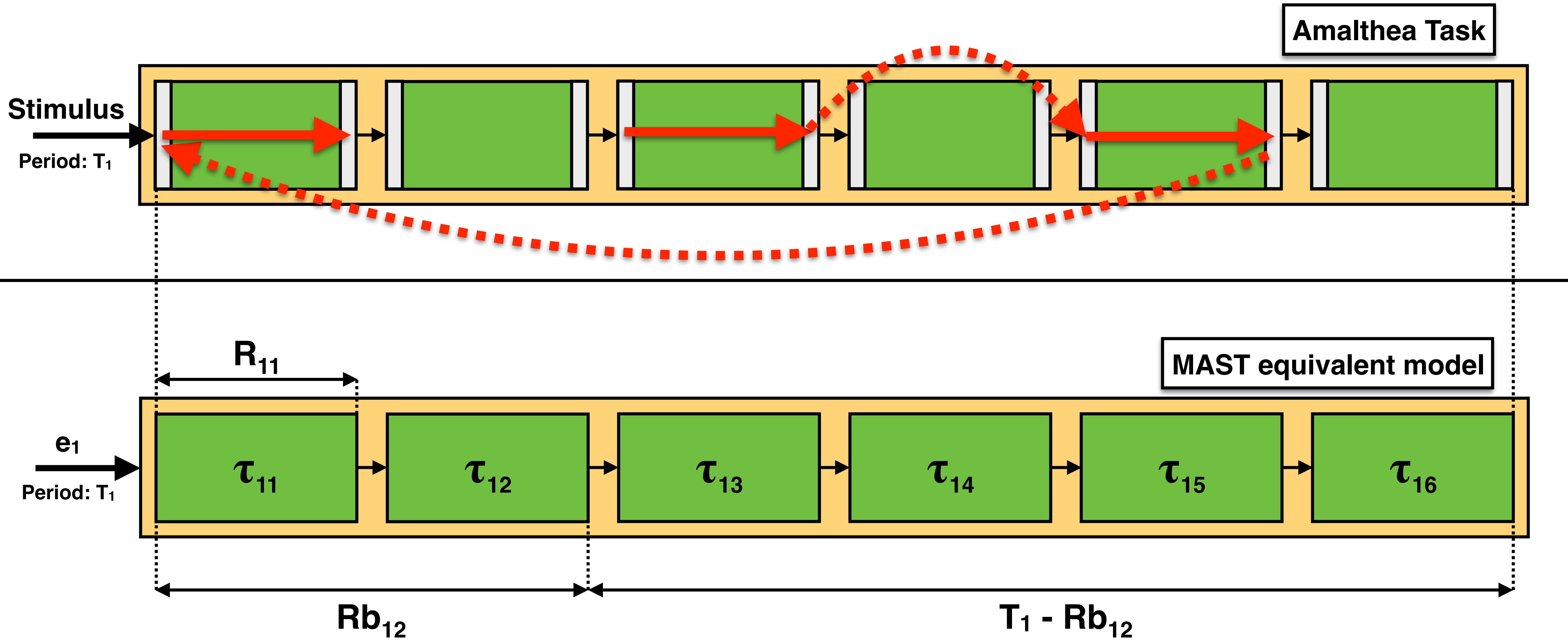
$$L=r_{11}+T_2+r_{22}+T_3+r_{33}$$

Best-case latency assumes labels are written just before they are going to be read

$$L^b=r^b_{11}+r^b_{22}+r^b_{33}$$

Event-chain analysis (3/3)

- **Runnables from the same Amalthea task:**



Worst-case latency

$$L=(T_1-R^b_{12})+R_{11}$$

Best-case latency

$$L^b=(T_1-R_{12})+R^b_{11}$$

Challenges

- **Challenge 1: ignoring memory accesses**
  - **Execution time of MAST steps comprised of only Runnable instructions**

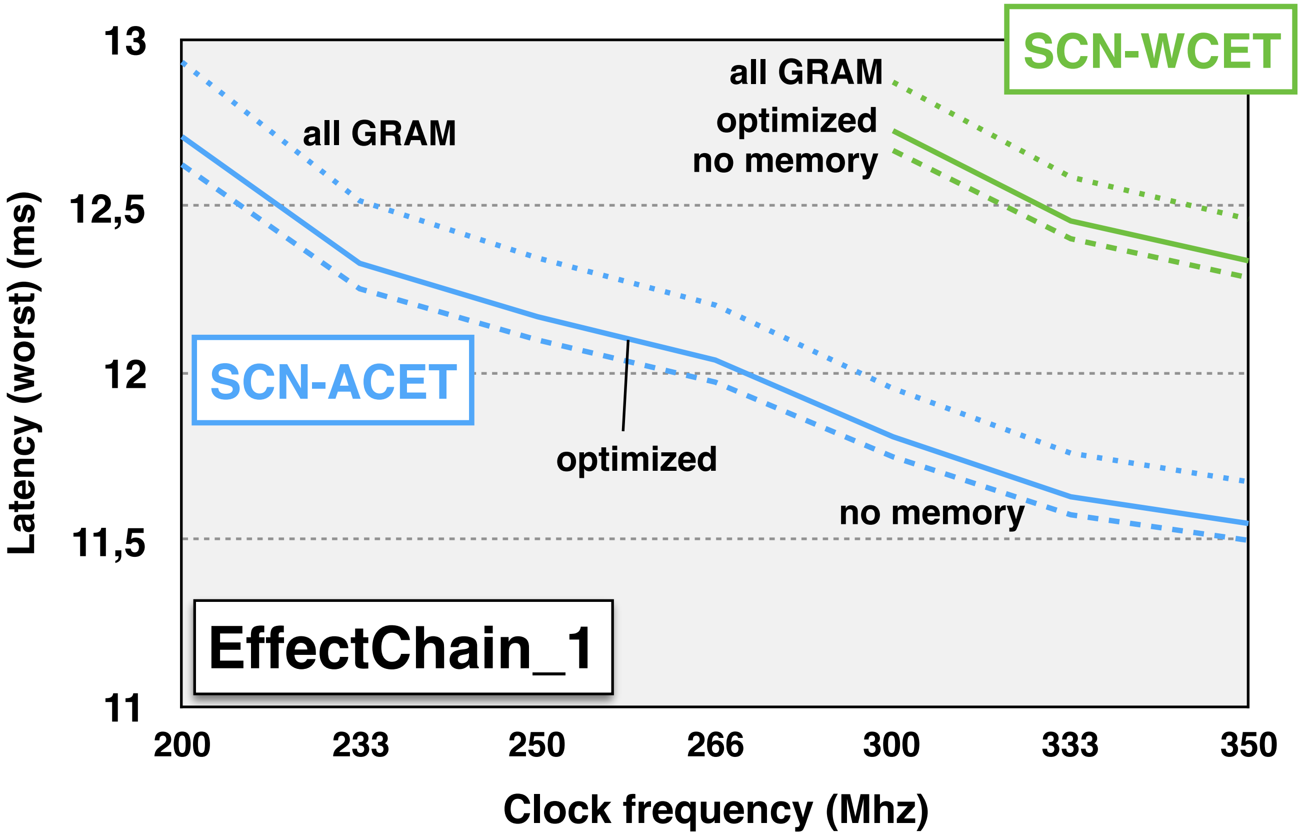
- **Challenge 2: all labels to GRAM**
  - **Execution time of MAST steps comprised of Runnable instructions + worst-case memory access costs**

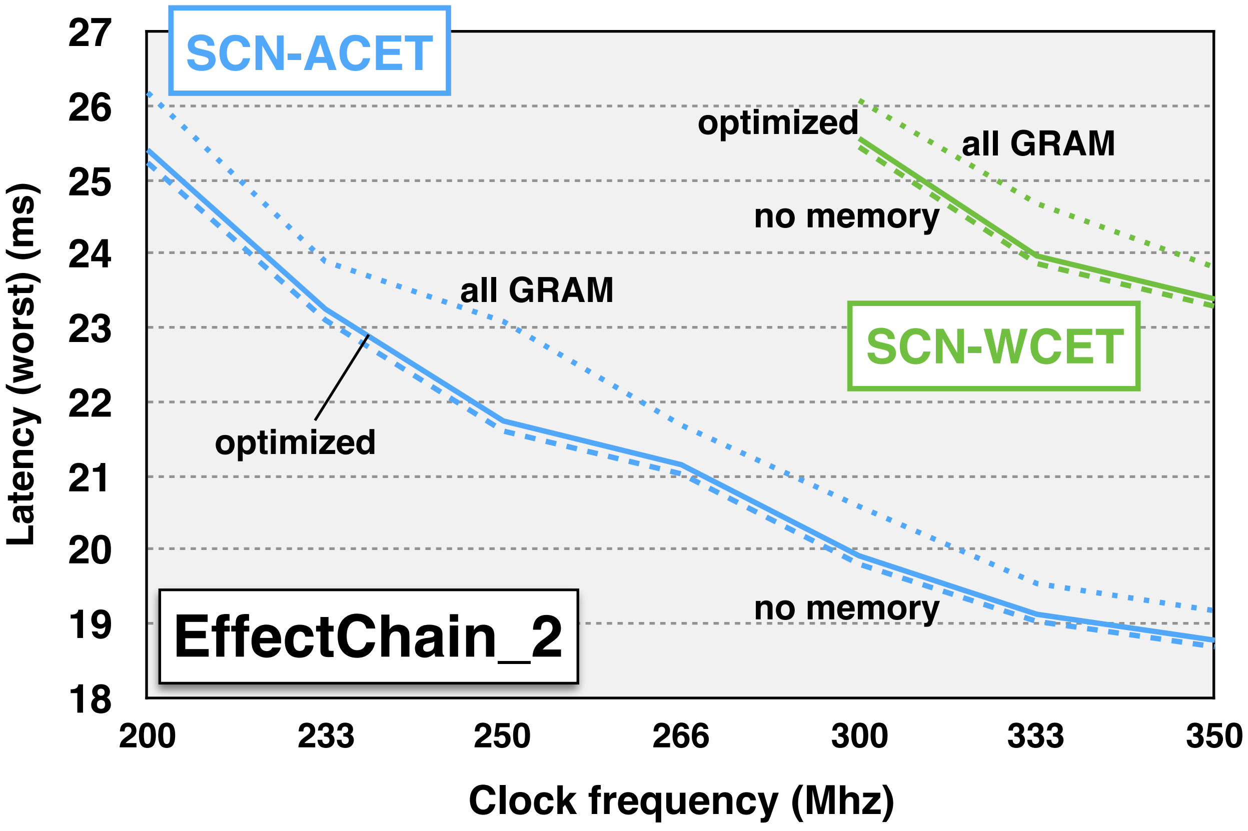
- **Challenge 3: find optimized allocation of labels to GRAM and LRAM**
  - **Label optimization not supported in MAST, but….**
  - **83% of labels are accessed by only one core**
  - **Proposal: Shared labels to GRAM, non-shared labels to their core LRAM**
    - LRAM is accessed without contention (1 cycle access)
    - GRAM is accessed as before (4*9=36 cycles access)

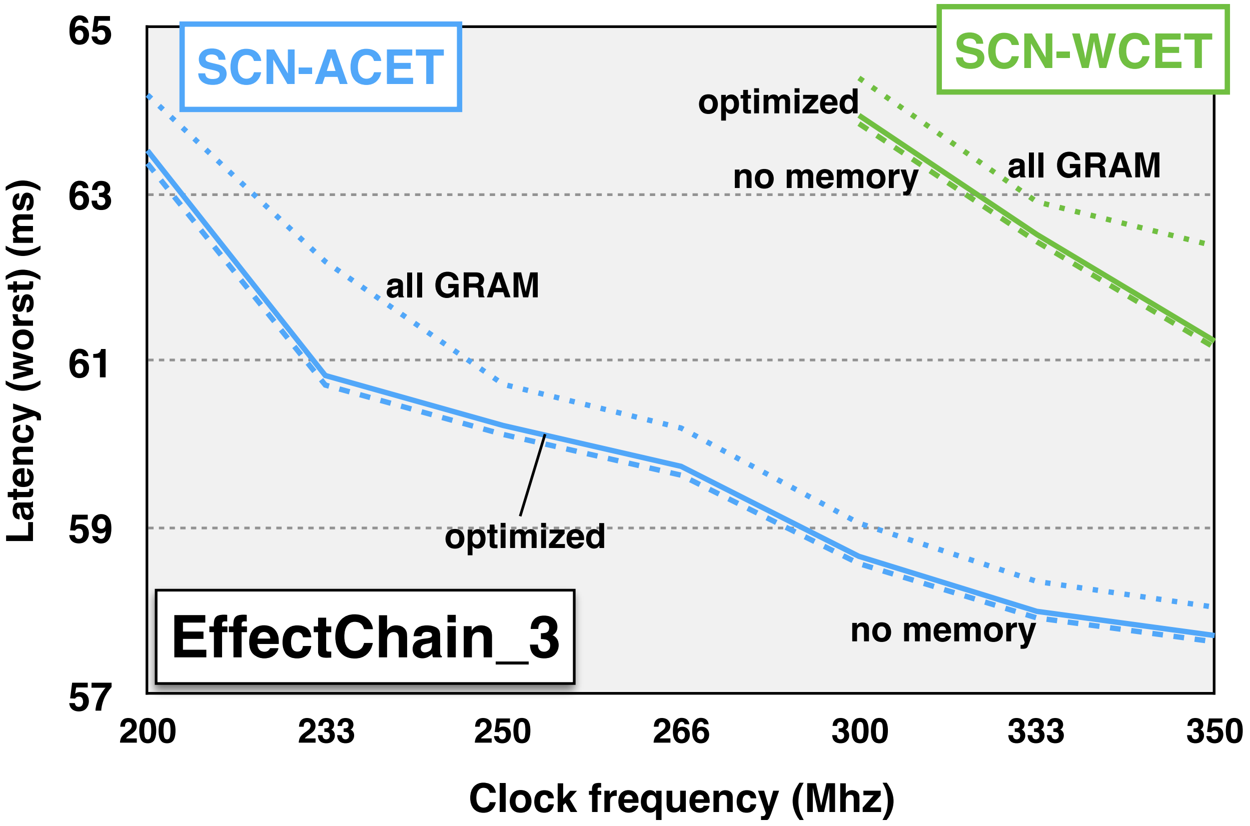
Evaluation

- Amalthea to MAST transformation (M2T)
  - 10 minutes approx.
- Response-time analysis technique applied
  - Offset-based Analysis with Precedence Relationships Optimizations
  - Better suited for end-to-end flows that stay in the same processor
  - 1-5 minutes to analyze each system
- Amalthea model has utilizations above 100%
  - SCN-ACET: Mean value of instructions used as steps WCET
  - SCN-WCET: Maximum value of instructions used as steps WCET
  - Different clock speeds tested: [200, 233, 266, 300, 333, 350] Mhz

Results (Event-chains)





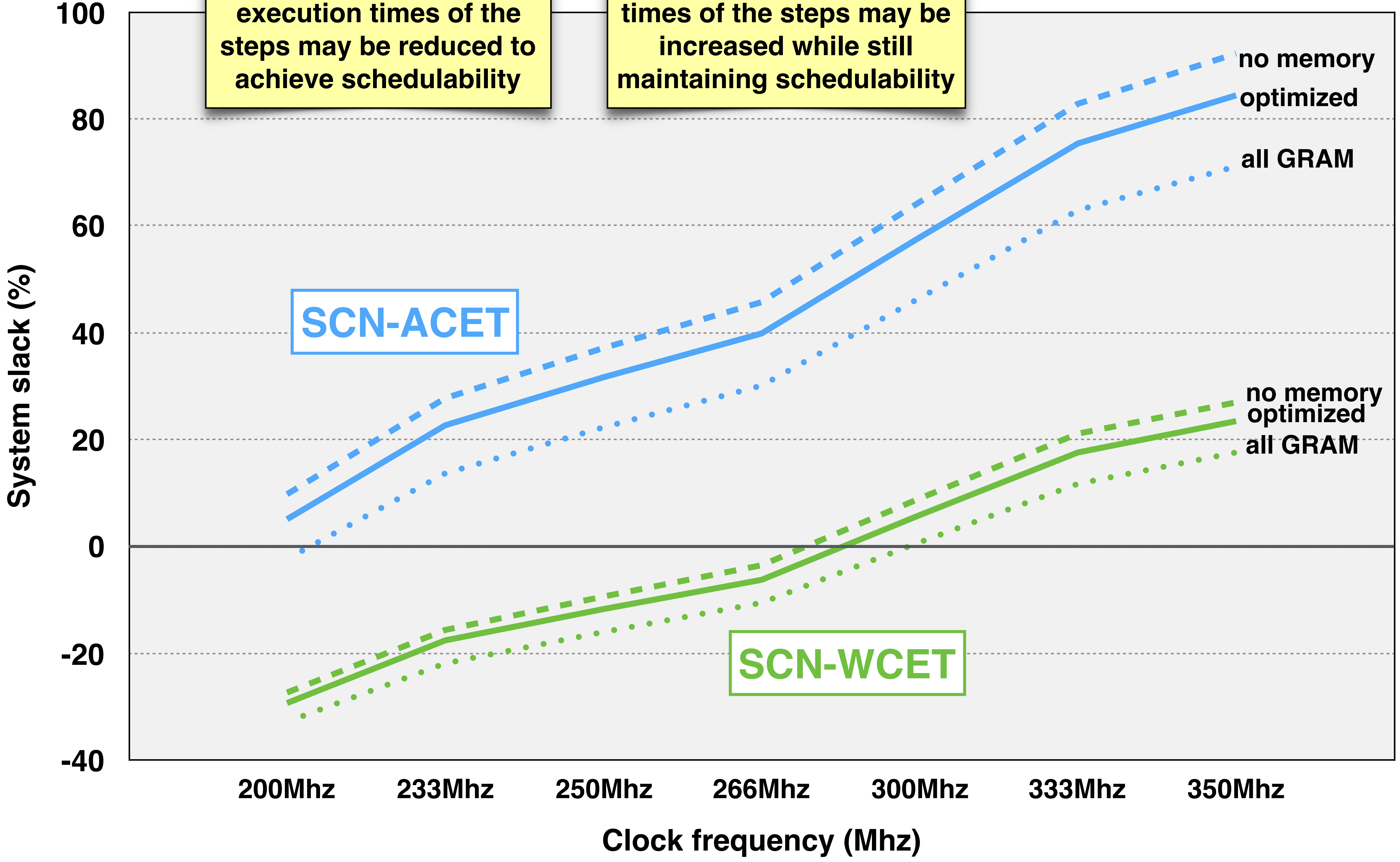


- **Latencies for the optimized label allocation are closer to the case ignoring memory accesses**
- **Latencies for SCN-WCET can only be obtained for 300Mhz and above**

Results (System Slacks)

Negative slack: Percentage by which the execution times of the steps may be reduced to achieve schedulability

Positive slack: Percentage by which the execution times of the steps may be increased while still maintaining schedulability



Conclusions

- **Demonstration of how MAST can be applied to this kind of systems**

- **System is analyzed as a whole**

- **Results for the three challenges**

- **Workspace and results are available:**
  - www.istr.unican.es/members/rivasjm/workspace_fmtv16_public.zip

- **Drawbacks**
  - **Pessimistic modeling of memory accesses**
  - **Pessimistic event-chain analysis**
  - **Cannot calculate latencies when overloaded**
    - But sensitivity analysis can be performed

Thank you for your attention!
Any questions?