Critical Applications
on Predictable High-Performance
Computing Architectures

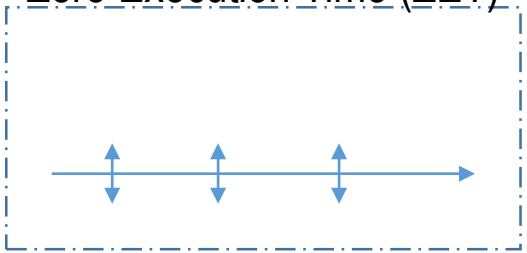
Accounting for interferences in the design of Time-Triggered Applications

Antoine FERLIN, Eric Jenn, Marc Kaufmann

ERTS 2020

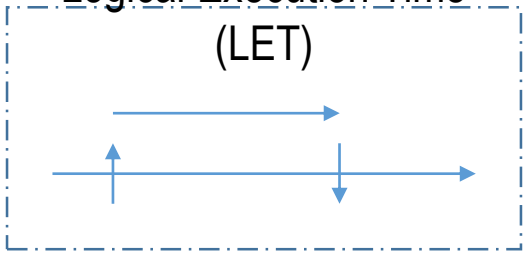
INTRODUCTION

Zero Execution Time (ZET)



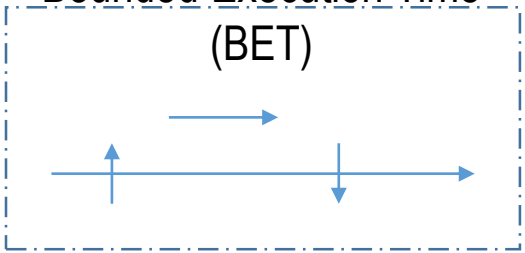
et = 0

Logical Execution Time (LET)



et = T

Bounded Execution Time (BET)



0< et <T

↑ Reading input

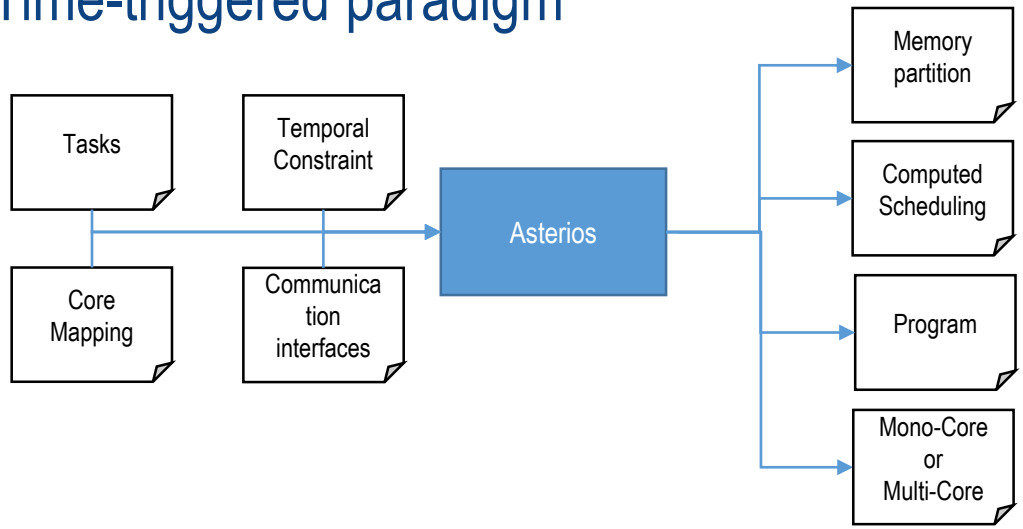
↓ Writing output

et Execution time

Time-Triggered = **LET** with event triggered by real-time

CONTEXT : ASTERIOS

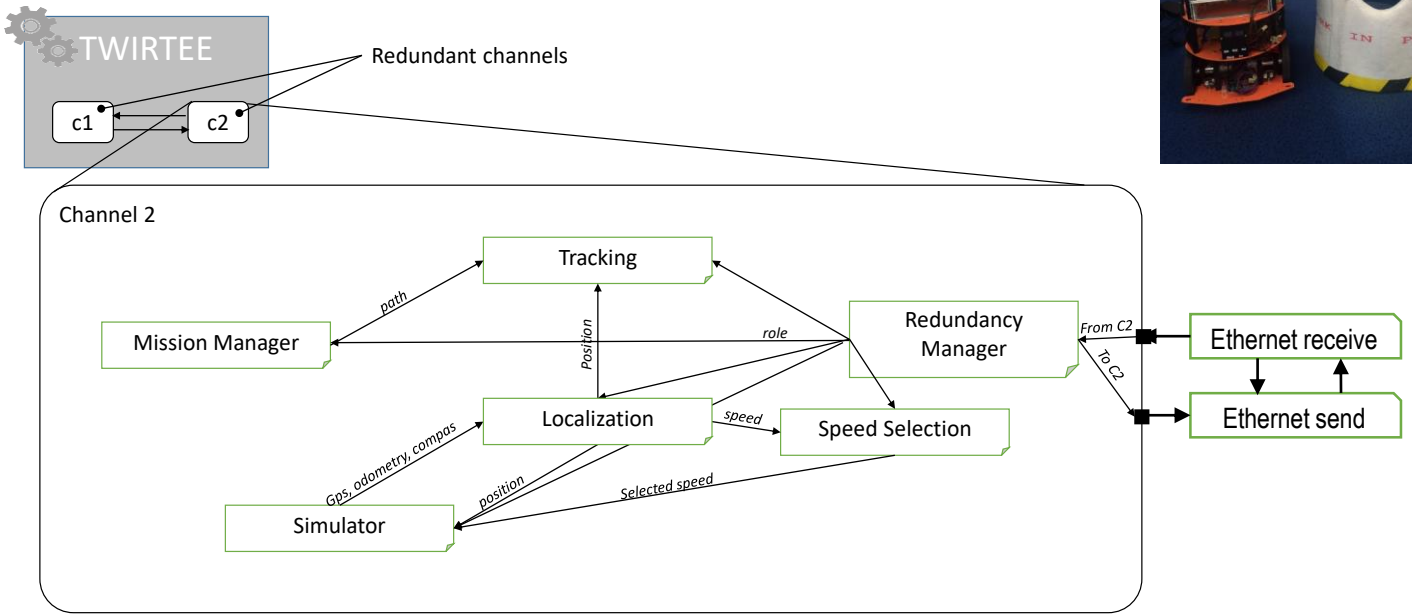
- Asterios : Time-triggered paradigm



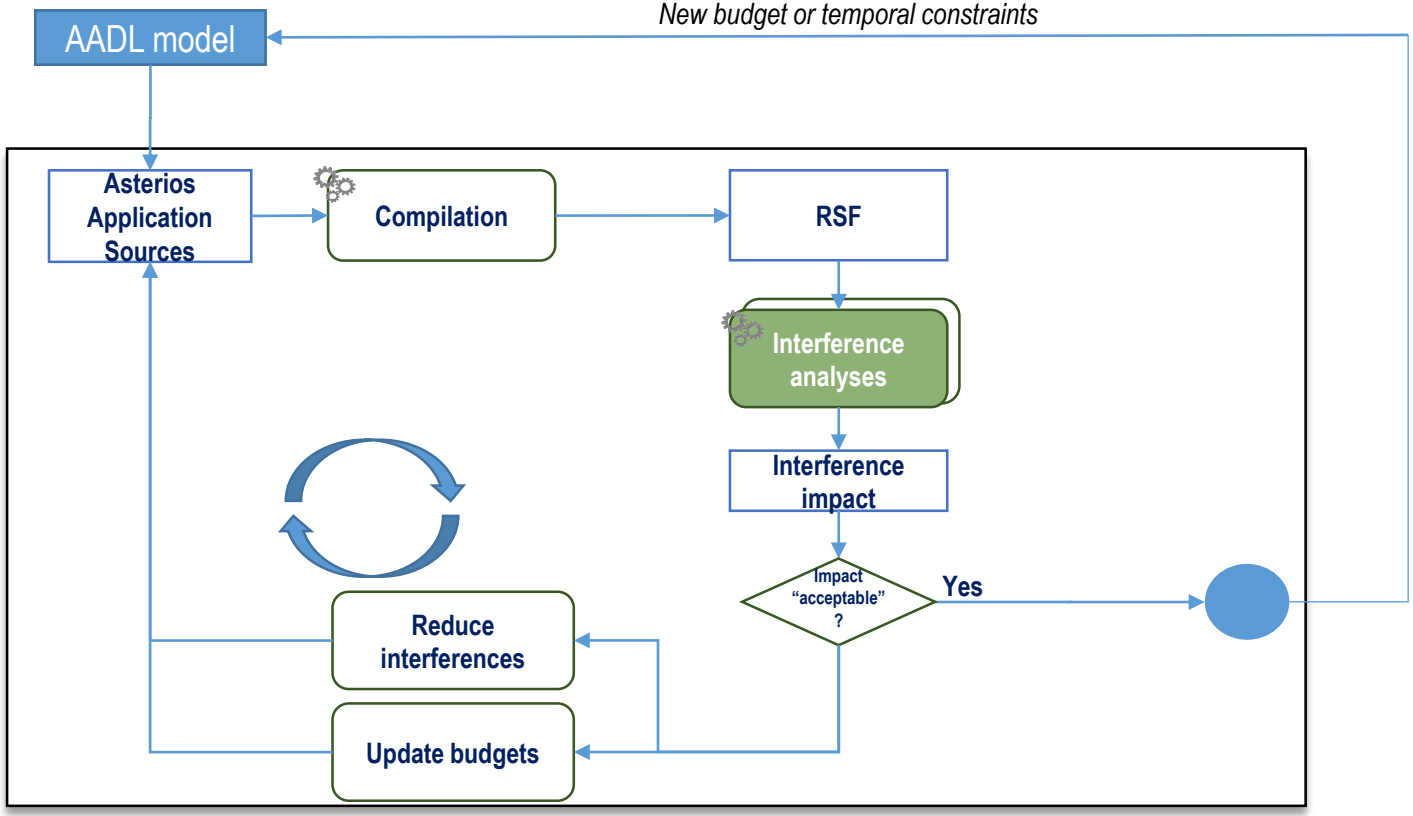
- Difficulty :
  - Time-budget evaluation of each task

CONTEXT : USE CASE

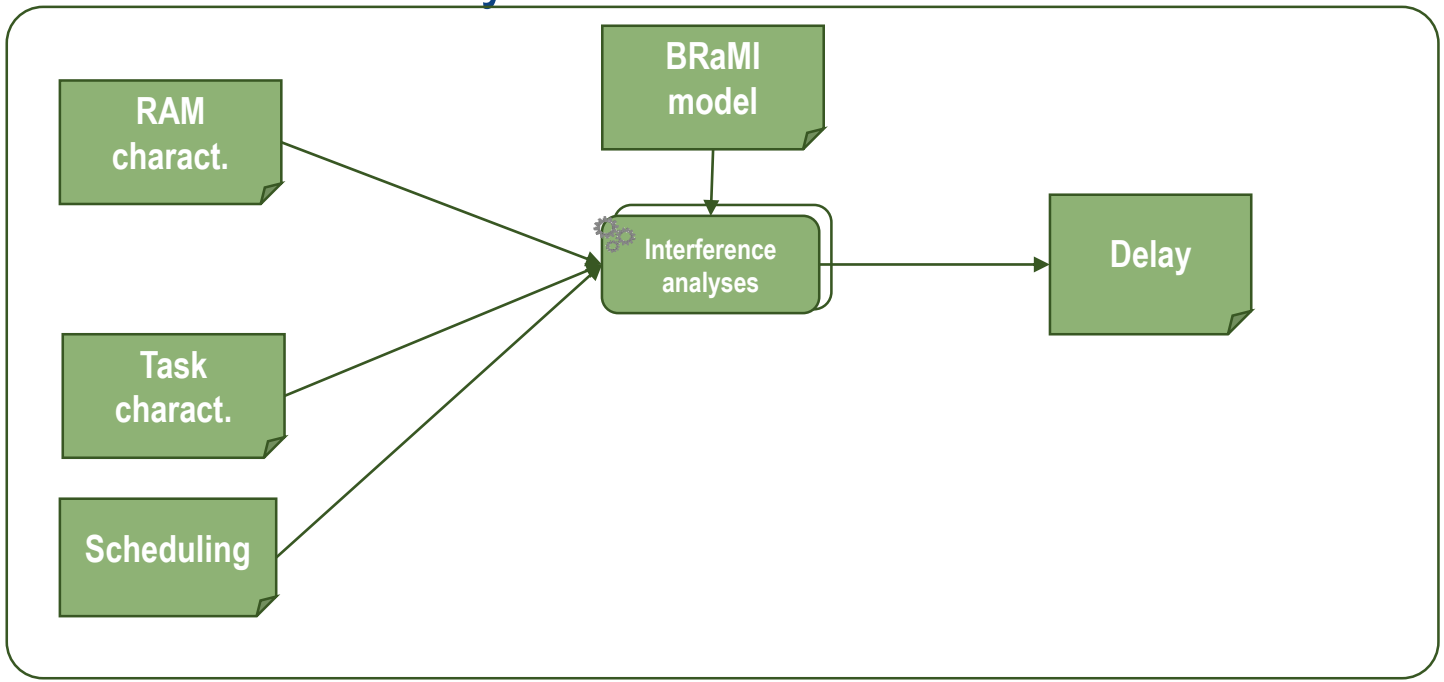
- Software : TwIRTee
- Hardware : T1040



OVERVIEW

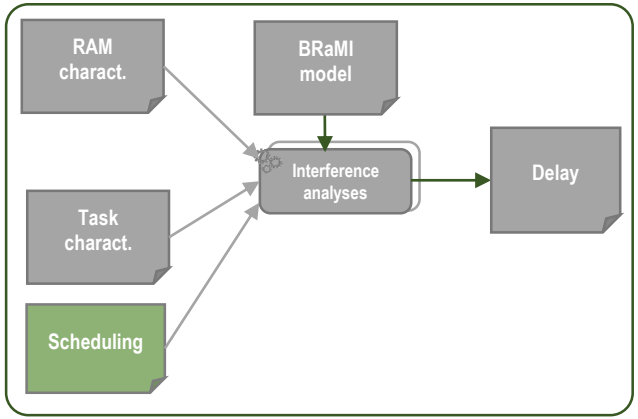


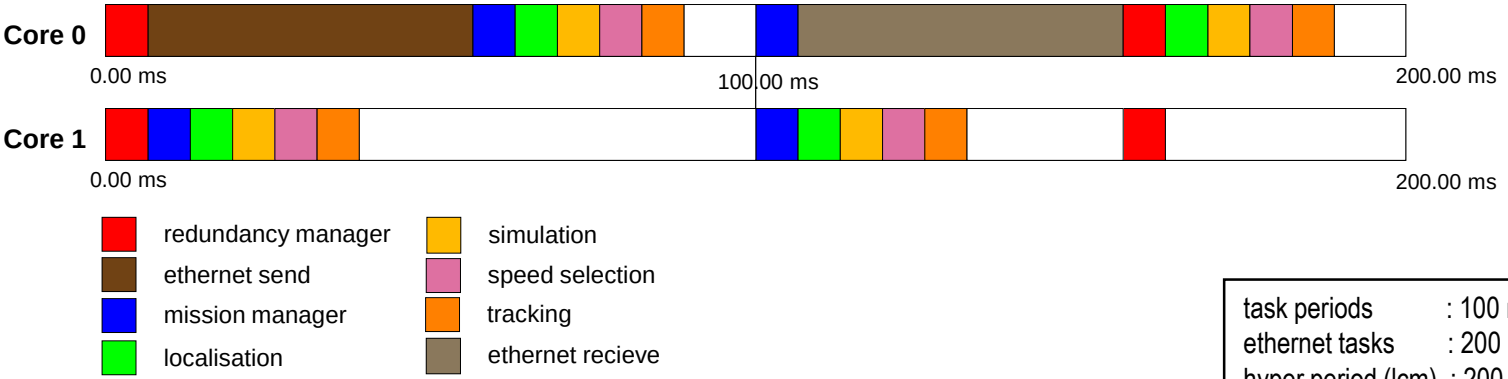
- Detection of memory interferences



BRAMI

- Scheduling extraction
  - RSF computed by Asterios
    - Focus on the repetitive part

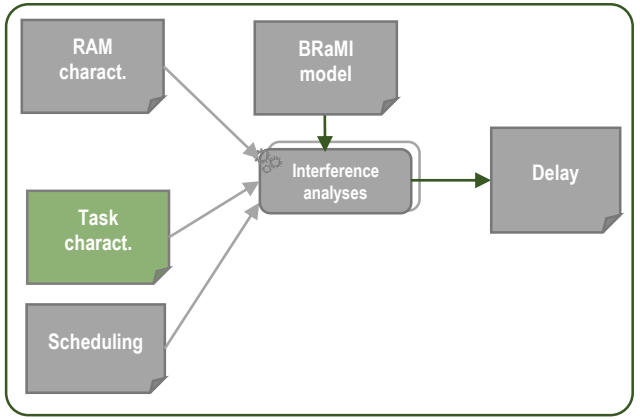




task periods : 100 ms
ethernet tasks : 200 ms
hyper period (lcm) : 200 ms

BRAMI

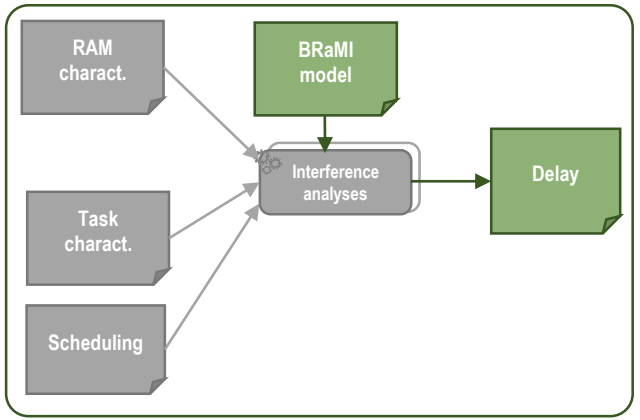
- Estimating the number of memory accesses
  - On the targeted hardware using a probe
  - **On a virtual platform using VLab**



- Worst case of memory access (WCMA)?
  - Could be deduced from measure using Extreme Value Theory (EVT)
    - MBPTA : Measurement Based Probabilistic Timing Analysis
    - Adaptable to obtain WCMA instead of WCET

BRAMI

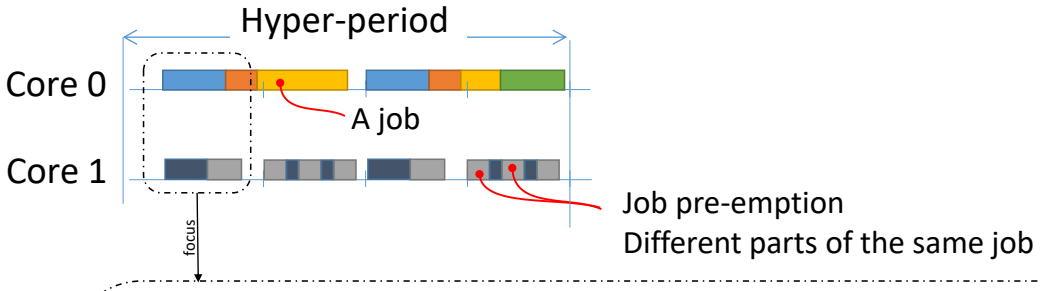
- Memory interference when
  - Two tasks are executed simultaneously
  - Tasks require simultaneously memory access
    - Data Request (Read, Write)
    - Instruction Request (Read)

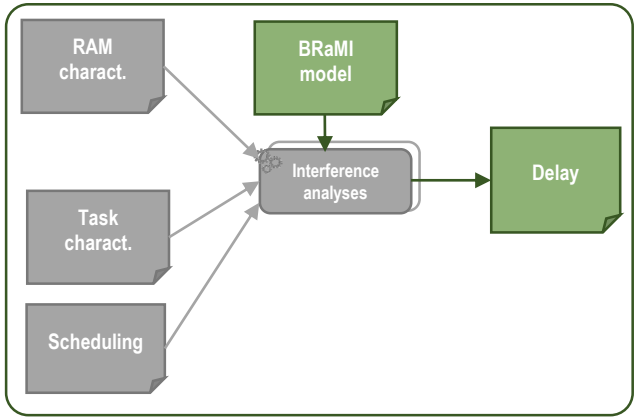


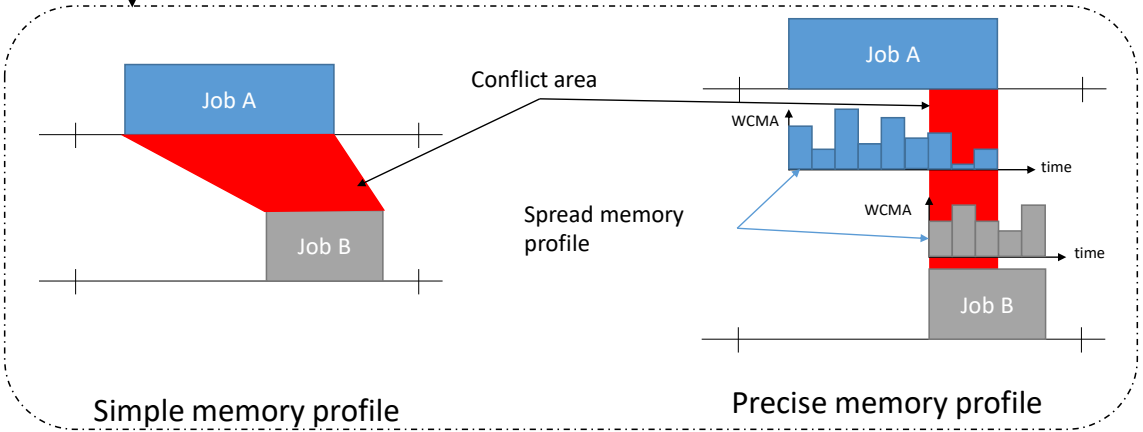
- Memory interference generates delays during HP:
  - $WCMAD(\tau) = \underbrace{\max_{j \in J(\tau)}(delay(j))}_{\text{Worst delay}} \times \underbrace{\frac{HP}{P(\tau)}}_{\text{Number of Job during HP} = ||J(\tau)||}$
  - $delay(j) = \sum_{j' \in \text{conflict}(j)} \underbrace{RD_{core(j')}}_{\text{Delay of one request [13]}} \times \underbrace{WCMA(j')}_{\text{Worst Number of requests}}$
  - CPU cache not taken into account

[13] H. Kim, D. de Niz, B. Anderson, M. Klein, O. Mutlu and R. Rajkumar, Bounding and reducing memory interference in COTS-based multi-core systems, Real-Time Syst. Vol. 52, no. 3, pp. 356-395, May 2016

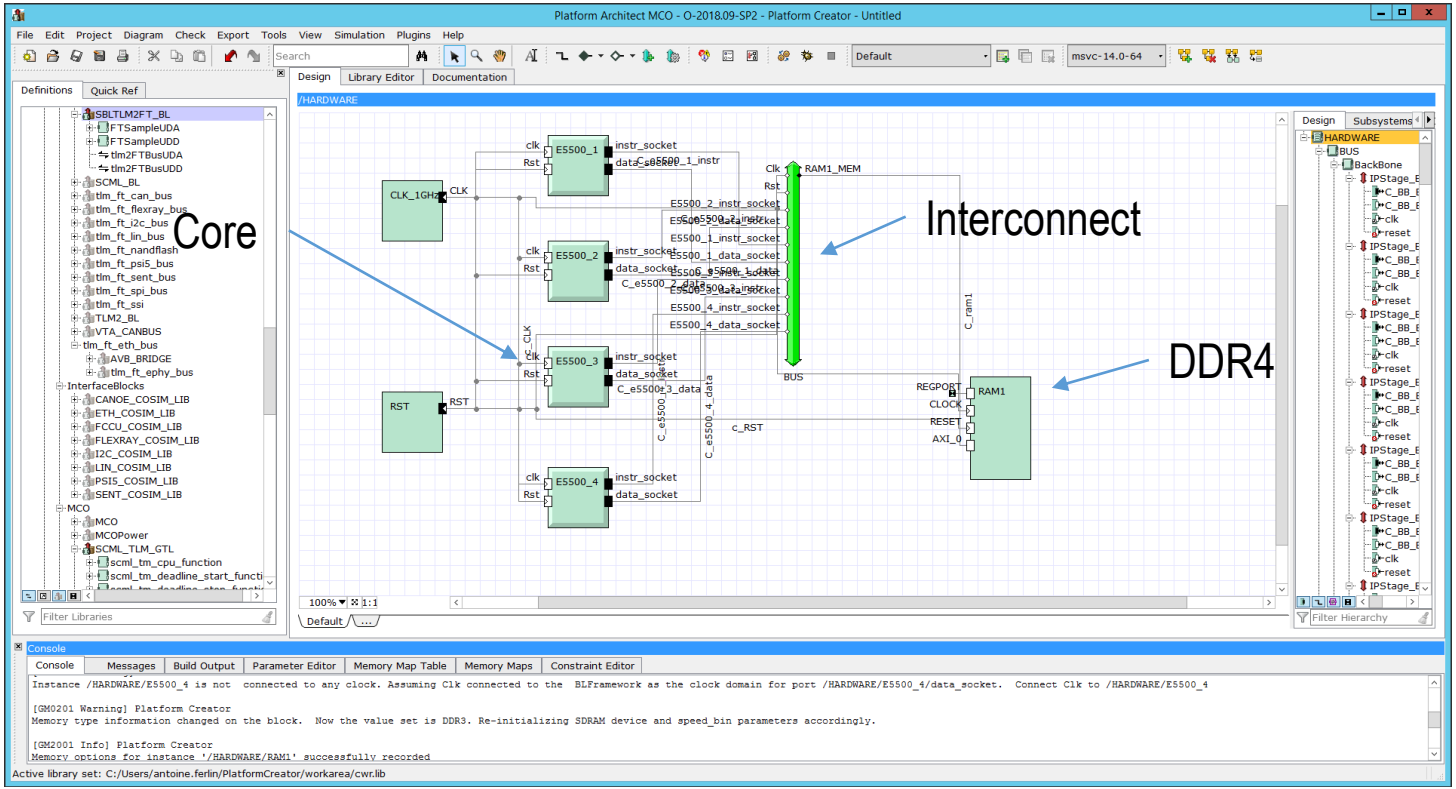
- Conflicts





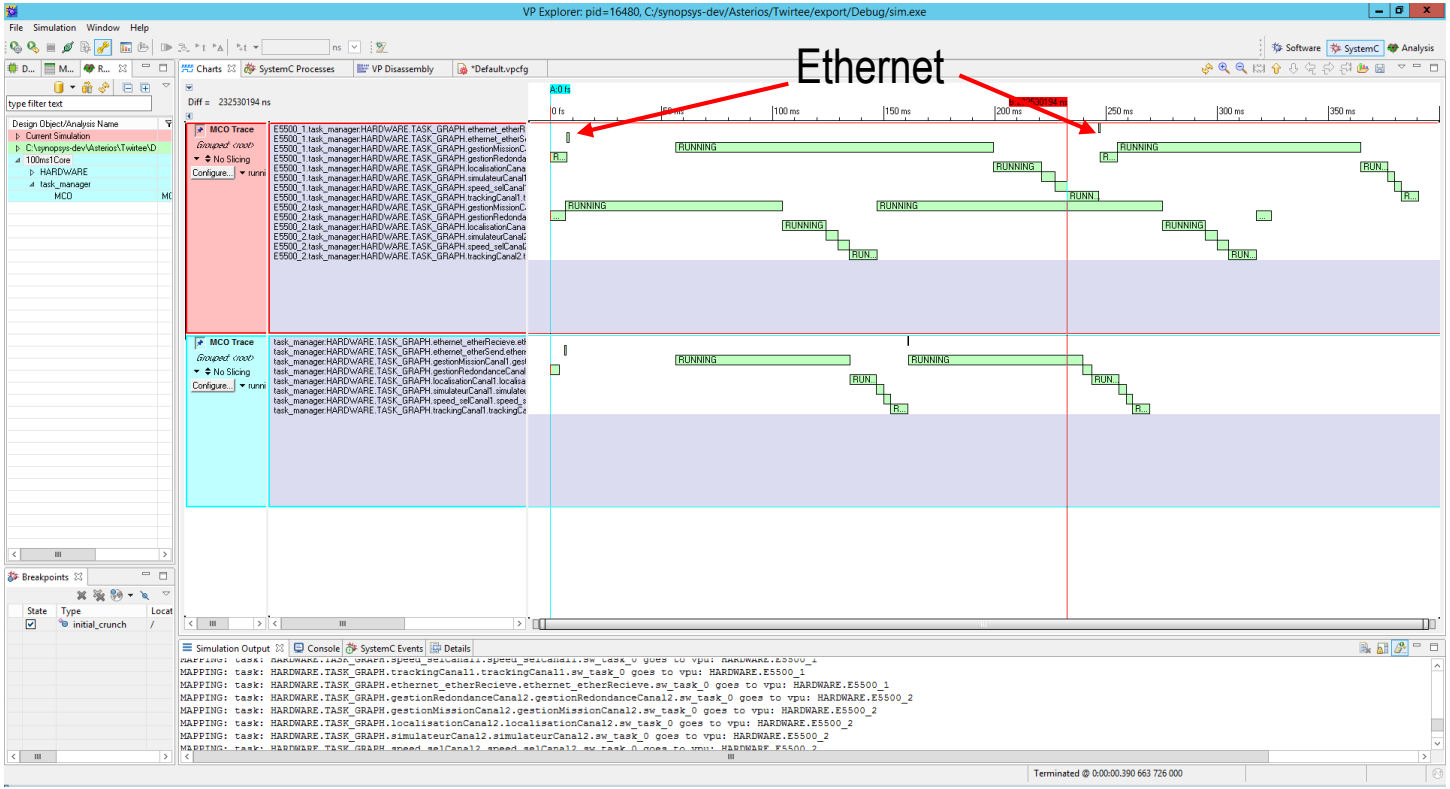


PLATFORM ARCHITECT



Hardware model + Software model

PLATFORM ARCHITECT

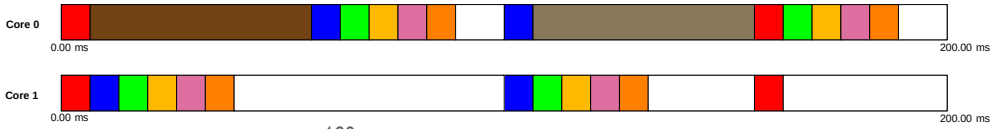


Two channels on two cores

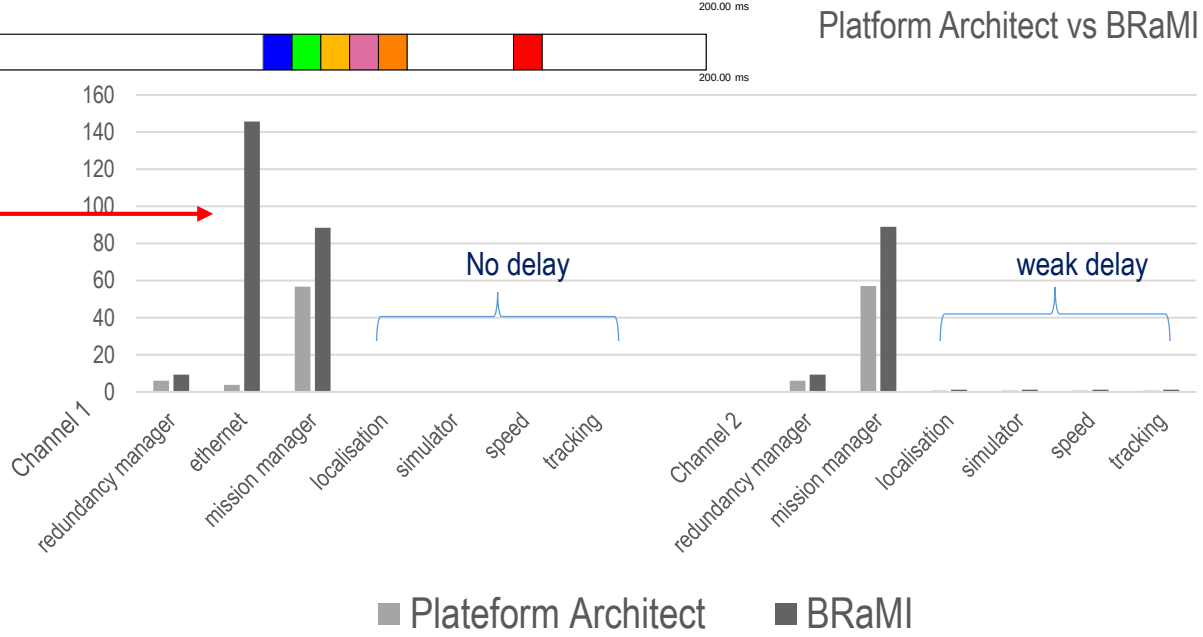
One channel on one core

BRAMI VS PLATFORM ARCHITECT

- **BRaMI vs Platform Architect**
  - BRaMI results close to Platform Architect results



- Singularity
  - Ethernet initial budget over-estimated
  - Model granularity to be enhanced



CONCLUSION

- Achievements :
  - Detection of interference and estimation of their impact
    - BRaMI model and computation of delay
    - Extraction of task characteristics using VLab
  - Using Platform Architect to compare results
  - First experiments using TwIRTee use case

- Remaining Work :
  - Validate our model against targeted hardware
  - Accounting for CPU caches
  - Improve memory access profile

Merci de votre attention