SAFE SCHEDULING ON MULTICORES

AN APPROACH LEVERAGING MIXED-CRITICALITY AND END-TO-END DEADLINES

Daniel Loche*†, Michaël Lauer†, Jean-Charles Fabre †

*Technocentre RENAULT
Guyancourt, France
daniel.loche@renault.com

† LAAS-CNRS
Toulouse, France
First.lastName@laas.fr

LET'S GET STARTED ..!

AGENDA

01

INTRODUCTION

- EMBEDDED SYSTEM EVOLUTION

- SAFETY CONCERN

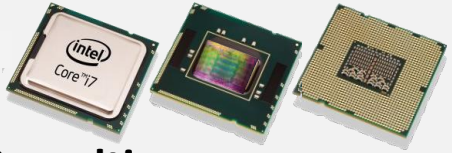
- TASK CHAIN BASED MODEL

EMBEDDED SYSTEM EVOLUTION

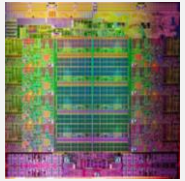
- Rise of resource demanding **software**
  ADAS, Autonomous & Connected car

- Monocore limitations & foundry tendency
  Parallel computing instead of frequency increases

- Industrial constraints : limited space, evolutivity
  Over-the-air updates, software services



→ **multicore processors**



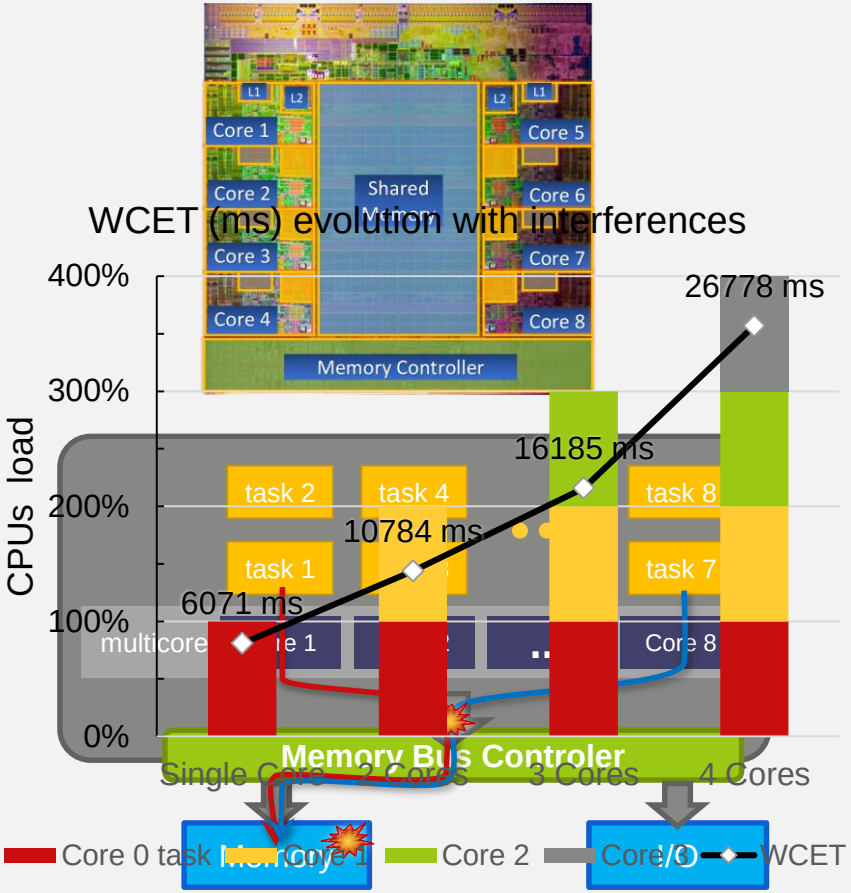
SAFETY CONCERNS

Multicore Complexity

Resource Sharing

uncontrolled inter-software Interferences

➢ **real-time constraint failures** ≺



SAFETY CONCERNS

- **Mixed-critical application**
  - Guarantee critical software execution
  - Allow non-critical quality of service

- **Existing Approaches**
  - *Scheduling policies (P-EDF, EDF-VD, RM…)*
    → Do not fit all the objectives

  - *time/space isolation (certified PikeOS…)*
    → Under-used resources

  - Watchdog mechanisms
    → Only failure detection

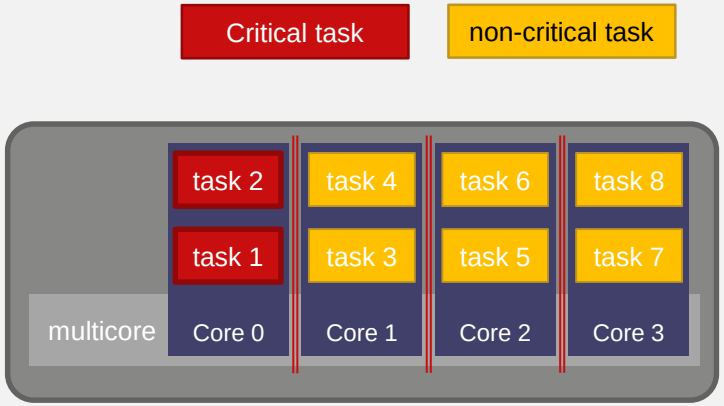
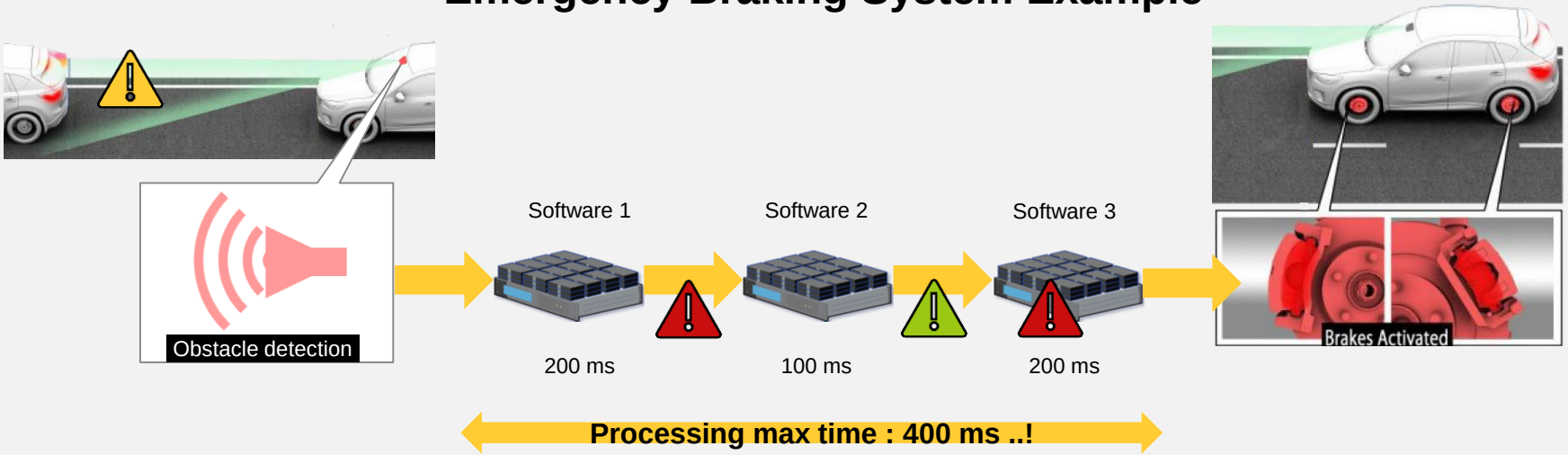


Figure 5 : Partitioned Scheduling

➢Anticipation mechanism ⮜

TASK CHAIN BASED MODEL

Emergency Braking System Example

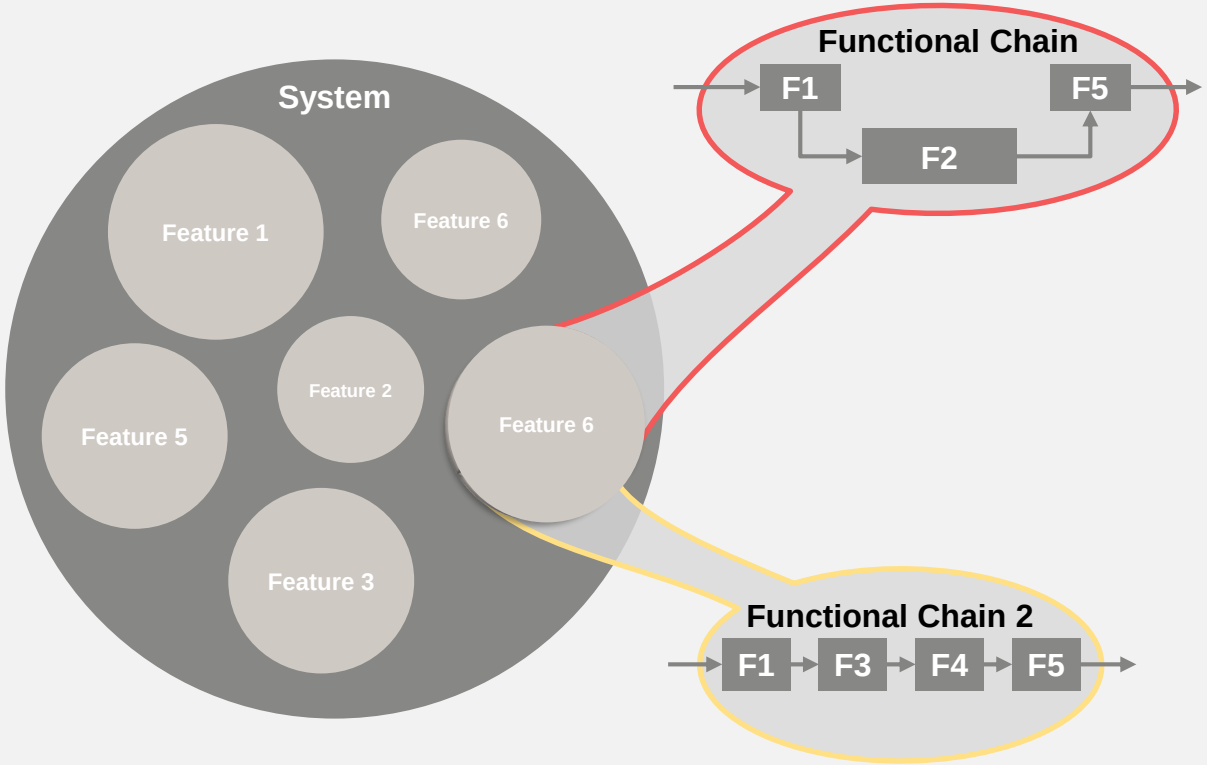


- Too late – ineffective.
- Too soon – over-anticipation. Resource waste. Possible instabilities.
- Optimal time – maximum resource use. System can self-regulate before.

➢ **Task chain observation to estimate best anticipation timing.** ⮜

TASK CHAIN BASED MODEL

Functional Chains



- System includes multiple **features**
- Each feature is defined by **Use Cases**
- A **Use Case** is realised by a **Functional Chain** made of functions and sub-functions.

TASK CHAIN BASED MODEL

Objectives

- Core functions + expendable functions

Vital components

possible exceptionnal drops

➢ **Anticipation** for vital component execute on guarantees

- Example - Lane Support System ADAS :
  - Emergency Lane Keeping => V
  - Lane Keeping Assist => NV
  - Lane Departure Warning => NV

Software correspondence

- Functions and sub-functions are realized by software components
  ➔ tasks
- Vital Functional Chains defines vital tasks

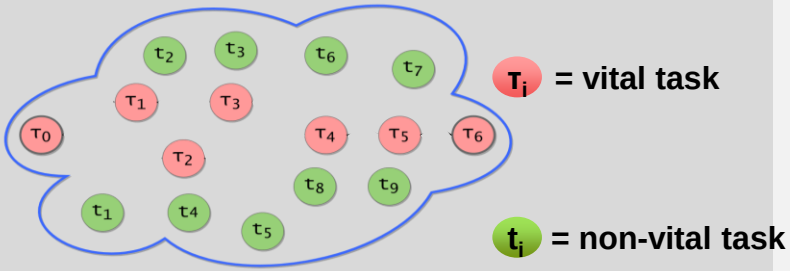


Figure 2 : Task set example

02

MONITORING AND CONTROL AGENT

- CONCEPT DESCRIPTION

- ARCHITECTURE

TASK CHAIN BASED MODEL

- **Software domain**
  - Vital tasks modeled as a task chain
  - Concurrent non-vital tasks

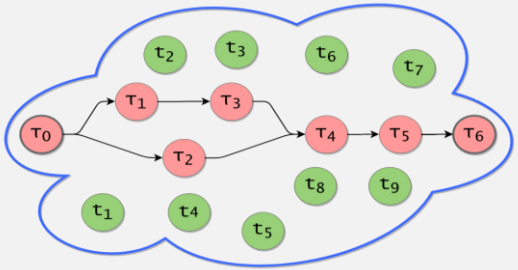


Figure 3 : Task set with a vital task chain

τ_i = vital task t_i = non-vital task

➢ End-to-end based constraints

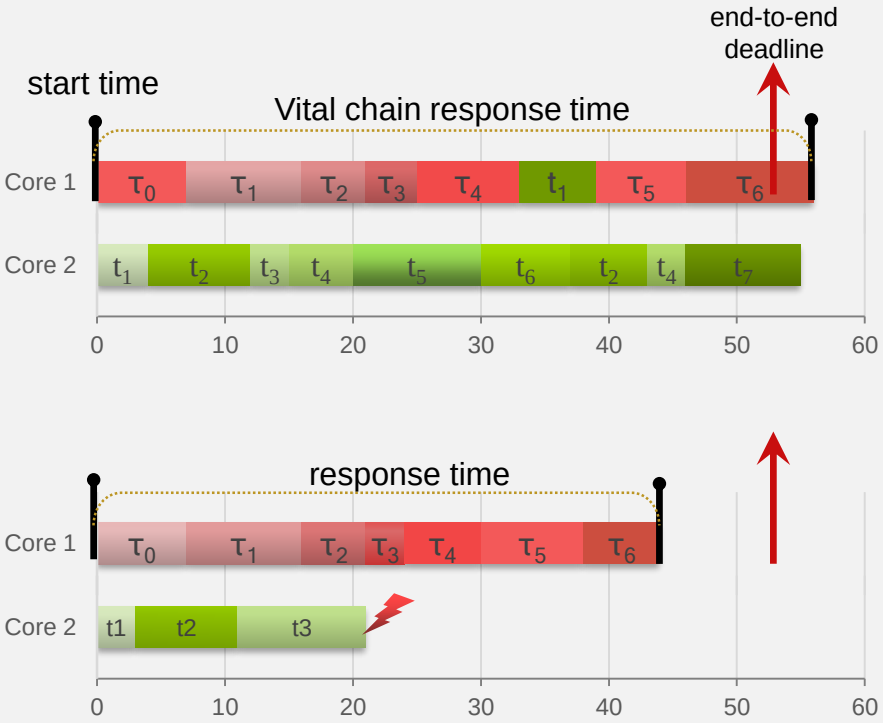
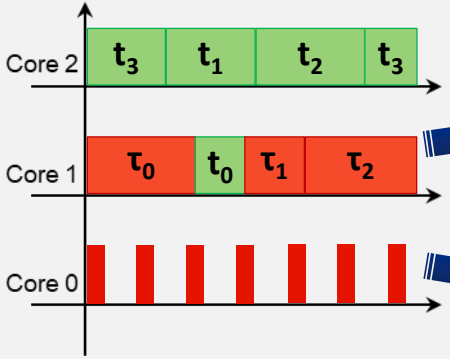


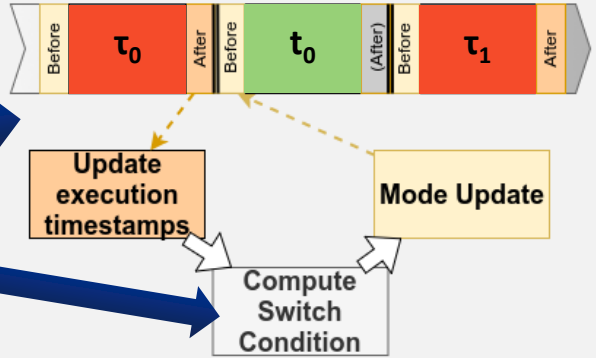
Figure 4 : Dual-core execution example

CONCEPT DESCRIPTION

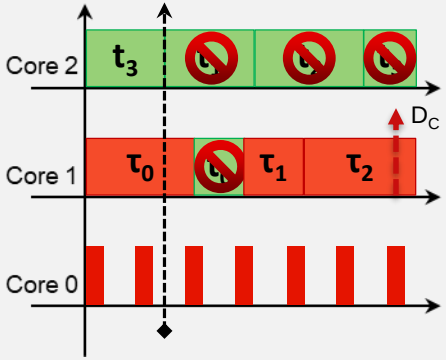
1. Monitor tasks execution timings



2. Pause non-vital tasks if needed



3. Garantee task chain deadline



t_0 Non vital tasks

τ_0 Vital tasks (Core 1)

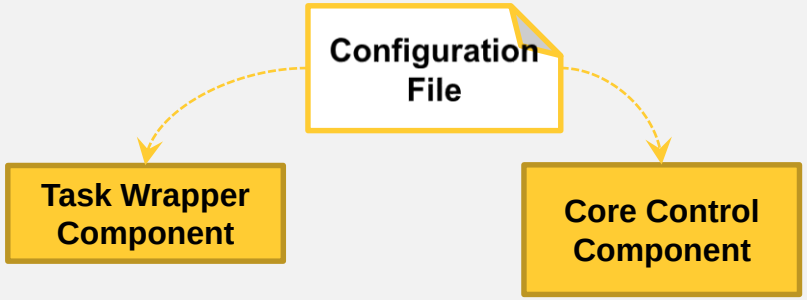
Monitoring (Core 0)

- Monitor vital tasks
  - ➢ *chain state*
- Compute switch condition

- Switch to **degraded mode**
  - ➢ Isolated chain executed
- Restore non-vital when task chain ends.

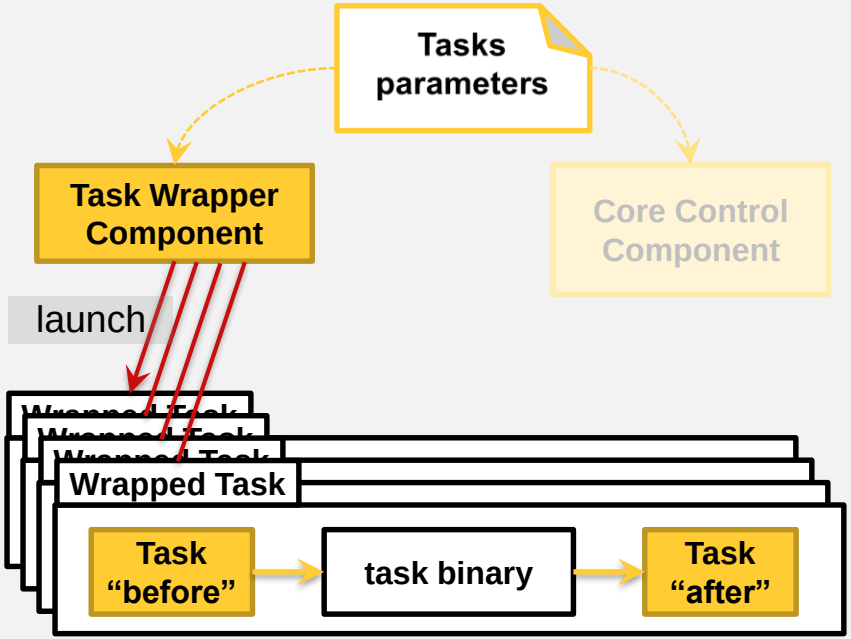
ARCHITECTURE

- **Application task set parameters**
  - for Monitoring & Control Agent database
- **Task Wrapper Component (TWC)**
- **Core Control Component (CCC)**



ARCHITECTURE

- **Application task set parameters**
- **Task Wrapper Component (TWC)**
  - Monitoring tools
  - Before block
    - non-vital “firewall” for backup mode
    - monitor vital tasks start
  - After block
    - monitor vital tasks ending
- **Core Control Component (CCC)**



ARCHITECTURE

- **Application task set parameters**
- **Task Wrapper Component (TWC)**
- **Core Control Component (CCC)**
  - task chain timing state update
  - Back-up mode triggering

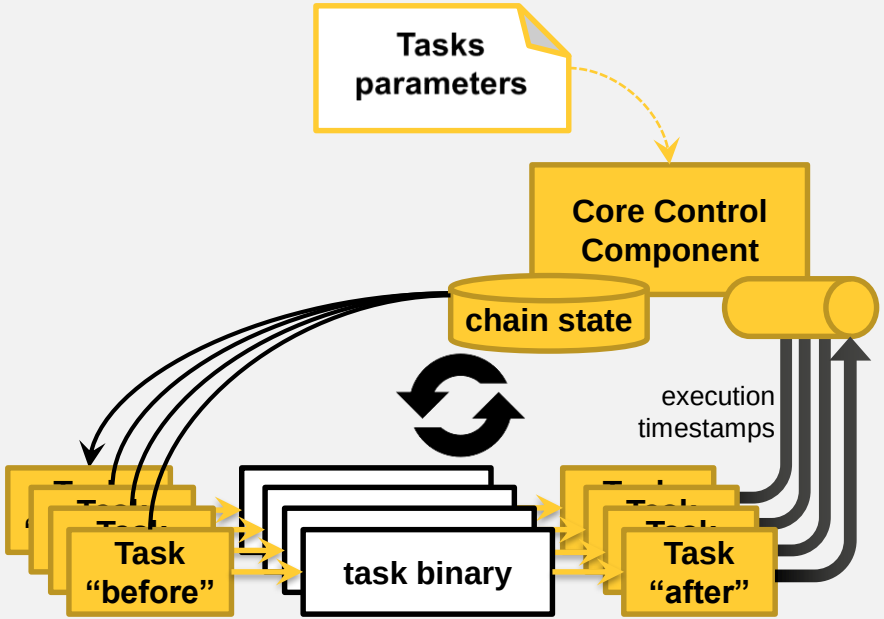
current exec. time

Switch time to backup

$$ET(t) + RWCET(t) + t_{SW} + W_{MAX} \geq D_c$$

Remaining Guaranteed time in backup

monitoring max. period



03

EXPERIMENTAL PLATFORM

- OBJECTIVES

- PRINCIPLE

- PRELIMINARY RESULTS

OBJECTIVES

Bench Platform

- Proof of concept
- Development process needs identification

Validation

- Task chain end-to-end timing guarantees
- Available computing resources for non-vital tasks
- Agent Scalability
  - No overheads

Analysis

- Comparison to other solutions (e.g. EDF)
- Behavior with different task set profiles
- Scheduling policy and tasks allocation influence

PLATFORM PRINCIPLES

MiBench tasks
FIFO / RM / RR Scheduling

Task Set Inputs

Activation model (periods)
Contraints (precedence)
Scheduling Policy
Close to "real" applications :
- complexity
- execution times
- functionnalities

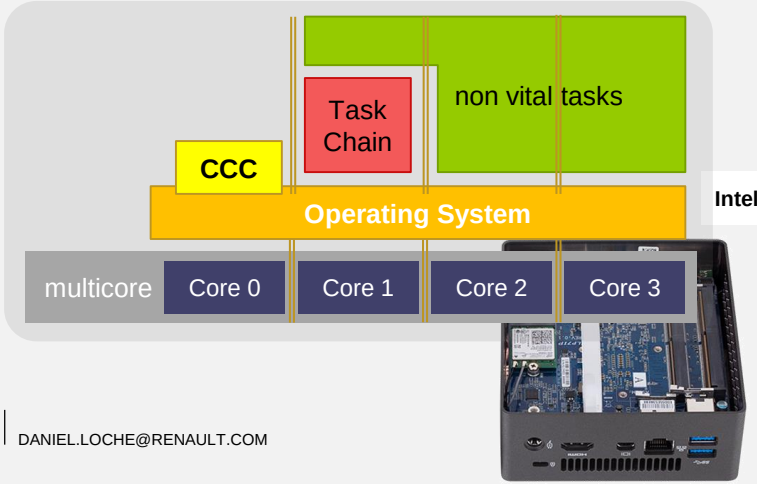
Hypothesis

vital task chain - Start & End events
non-vital tasks - no constrains
dual mixed-critical system

Experimental Platform

Outputs

Deadline misses
Execution time profiles
Task chain response time
CCC mode switch count

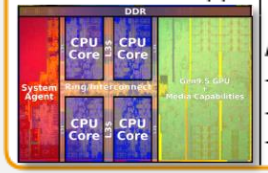


Intel I5-8250U

Support Platform

Hardware

multicore (min = 3)
for embedded app.



Software

Integration Tools

Available API :
- **Partitionning**
- Tasks monitoring
- Start / Stop signal

Linux + Xenomai RTOS

OFF-LINE ANALYSIS

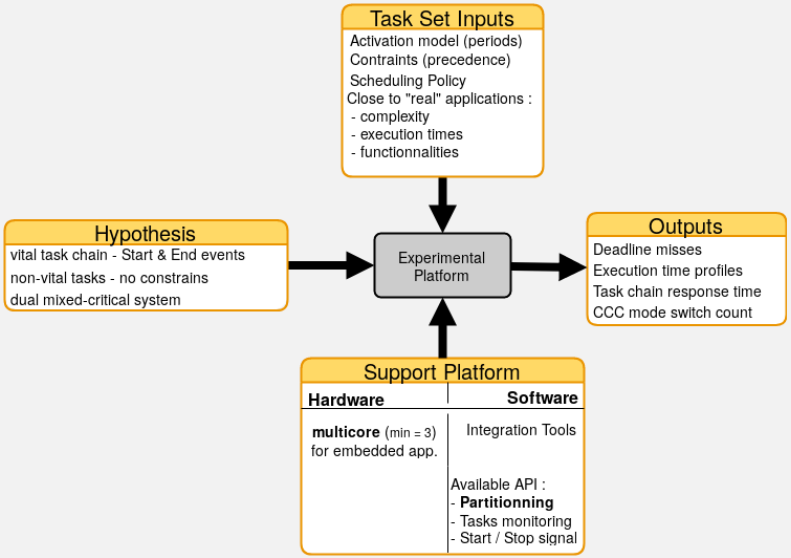
Preliminary Results

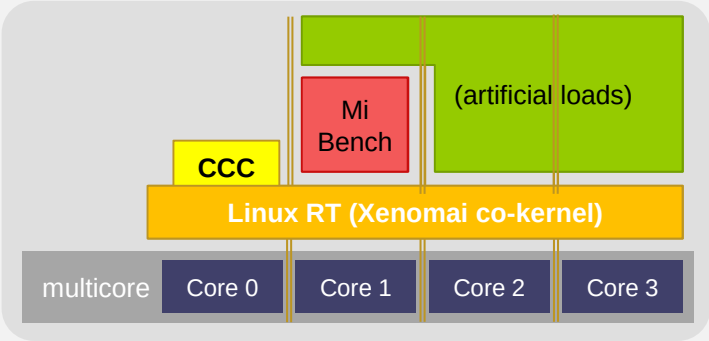
- Degraded mode switch time value

  t_{SW}=2ms
- Monitoring & Control period W_{MAX}

- (individual) Tasks Characterization
  - Execution time profiles
  - Experimental WCET

  } Task set specification
  - Strange behaviour from some MiBench tasks => not suitable, further tests needed





04

CONCLUSION & PERSPECTIVES

CONCLUSION

Conclusion

- Anticipation mechanism
- End-to-end deadline guarantee for a task-chain based critical function
- Expected fair computation sharing for non vital tasks
- On-going tests for performance analysis

Perspectives

- Industrial application tests
- **Multi-leveled degraded mode**
  - avoid stopping every non vital tasks at once
  - Requires more tasks information
- **Tasks allocation refinement**
  - Permit multiple vital task chains
  - Chose tasks allocation to improve performances

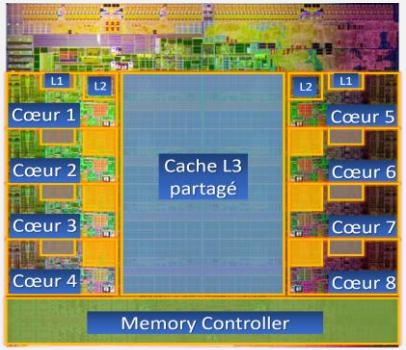
THANK YOU

MULTICORE DIFFICULTIES

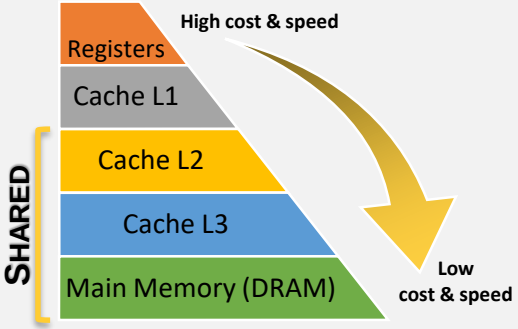
- **Predictability Issues**
  - Resource sharing & software concurrency
  - Tasks synchronisation
  - Memory limitations

➢ *Complex / Pessimistic WCET analysis*

- **Mixed-criticality Application**
  - Nominal execution
  - Transient faults due to interferences
  - Back-up mode to reduce interferences at a safe state



(a) Répartition des Caches sur un processeur Intel



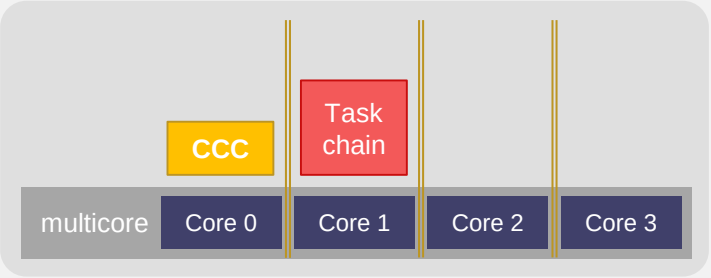
(b) Memory Space Hierarchy

BACK-UP MODE CHARACTERIZATION

- **Off-line characterization**

Vital task chain executed in back-up mode

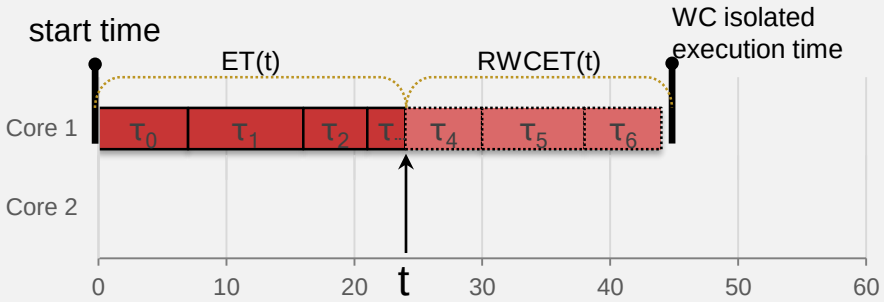
- on single core partition
- non-vital tasks stopped



- **$WCET_{ISO}$ measures**

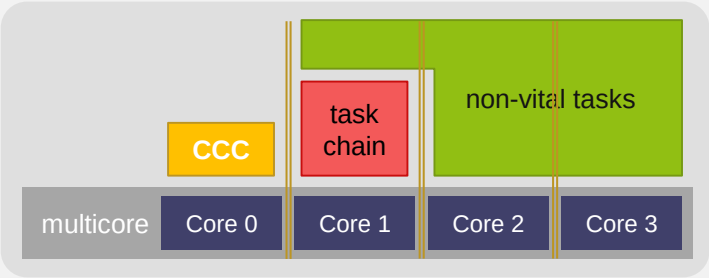
Worst-case execution time of each task

- **Chain state at time t**
  - executed tasks ET(t)
  - non executed tasks RWCET(t)
  - ➢ Worst-case isolated execution time.

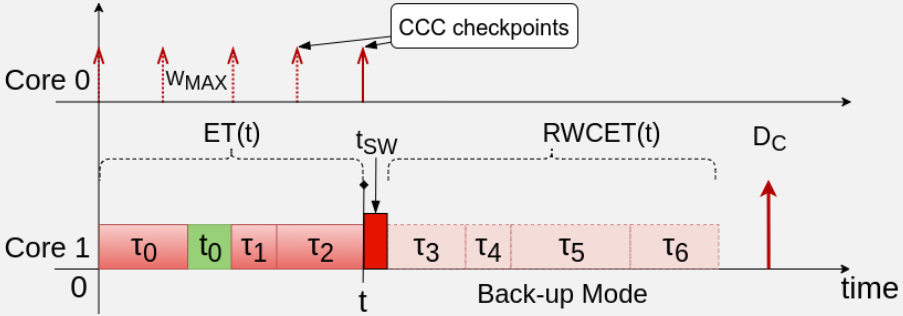


RUNTIME MONITORING

- **Switching latency t_{SW}**
  - implementation-dependent



- **At choosen period W_{MAX}**
  - too short
    ↳instability & CPU "waste"
  - too long
    ↳*high false-positive anticipation rates*

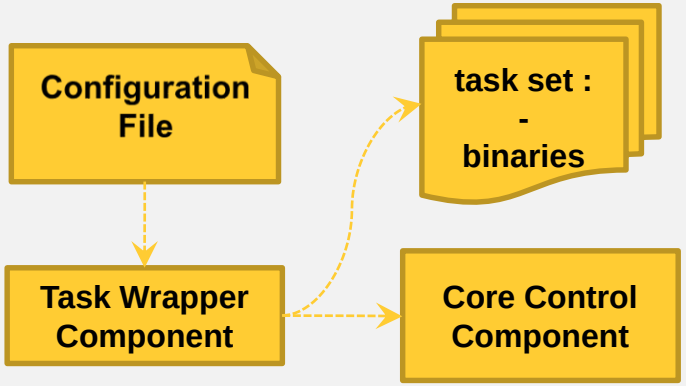


$$\boxed{ET(t) + RWCET(t) + t_{SW} + W_{MAX} \geq D_c}$$

anticipation

SETUP & RUNTIME ENVIRONMENT

- **Configuration file**
  - tasks parameters (periodicity, priority, core..)
  - Off-line characterization (WCET, W_{MAX}, t_{SW})
- **Task set binary files**



OS & Runtime environment

- **Linux ecosystem**
  - **Compatibility**
    - multimedia application
    - embedded software approximation
  - **Adaptability**
    - scheduling policies
    - partitionning and isolation features
  - **Xenomai Patched kernel**
    - soft real-time performances
    - API for Monitoring & Control Agent

EXPERIMENTAL SETUP - MIBENCH APPLICATION BENCHMARK

- **Multi-purpose benchmark**
- **Generate simple task chains**
  - imitate existing task chain applications
  - generate a pool of non-vital tasks from multimedia applications

    1. Signal Data Processing
       (bitcount, fft, sha, basicmath, qsort)
    2. Trajectory Computing
       (bitcount, CRC32, dijkstra, blowfish, basicmath, fft, fft-inv)
    3. Database Encryption
       (bitcount, CRC32, patricia, blowfish, basicmath)
    4. Voice Synthetiser
       (bitcount, gsm, basicmath, stringsearch)
    5. Image Processing
       (susan [smooth, edges, corners], jpeg, adpcm, bitcount, basicmath, dijkstra)

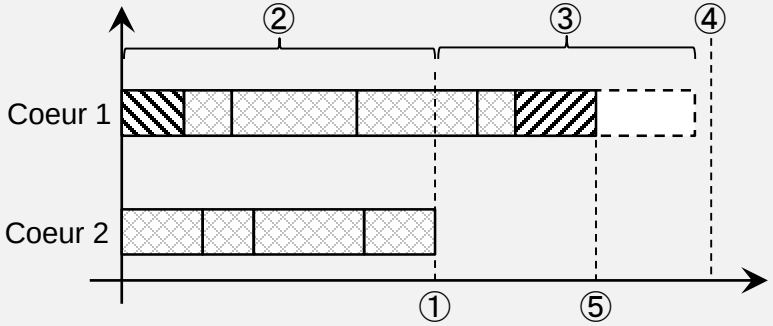
MiBench category	tasks
Automotive	basicmath, bitcount, qsort, susan
Network	dijkstra, patricia
Consumer	jpeg, typeset
Office	stringsearch
Security	blowfish, sha
Telecom	adpcm, CRC32, FFT, gsm

EXPERIMENTAL SETUP - MIBENCH APPLICATION BENCHMARK

TABLE II
MONITORING & CONTROL AGENT OVERHEADS SAMPLE

	Monitoring & Control ON		Monitoring & Control OFF	
Task	Median (ms)	Max (ms)	Median (ms)	Max (ms)
FFT (S)	33.80	35.63	32.72	34.01
Sha (L)	31.86	98.69	32.88	129.46
CRC32 (S)	34.70	36.56	28.63	29.80
rev FFT (S)	36.89	98.50	33.55	37.20
Bitcount (S)	13.61	31.41	12.02	13.00
dijktra (L)	51.12	63.38	38.13	67.55
Basicmath (S)	27.63	98.50	10.75	11.20

⓪ ① ② ③ ④ ⑤ ⑥ ⑦ ⑧ ⑨
⑩ ⑪ ⑫ ⑬ ⑭ ⑮ ⑯ ⑰ ⑱ ⑲ ⑳



① Point de contrôle
② Temps d'exécution de la chaine
③ Pire temps restant d'exécution
④ deadline (contrainte bout-en-bout)
⑤ Fin d'exécution réel de la chaine

autres tâches

tâche de début de chaine

tâche de fin de chaine