Towards power efficient mixed criticals systems

OSPERT Workshop, 9th July 2013, Paris

Florian Broekaert, Laurent San, Agnes Fritsch (Thales Communications & Security)
Sergey Tverdyshev (SYSGO)

Agenda

1. Introduction
   1. Context
   2. System integrators motivations
   3. Power management techniques overview

2. A Low-Power scheduler implementation
   1. Low-Power scheduling algorithm example
   2. End-User usage
   3. Implementation - A portable LowPower scheduler framework

3. PikeOS - scheduling for mixed critical systems
   1. PikeOS
   2. PikeOS scheduling with time partitionning

4. Low Power scheduling perspectives for mixed critical systems
   1. Extensions of PikeOS scheduler
   2. Integration at user VM
   3. PikeOS extensions

SYSGO EMBEDDING INNOVATIONS THALES

Introduction - Context

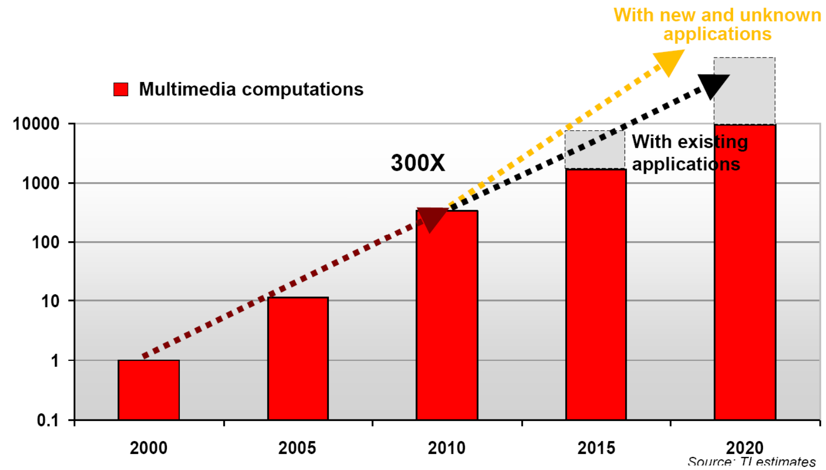
Context & Trends:

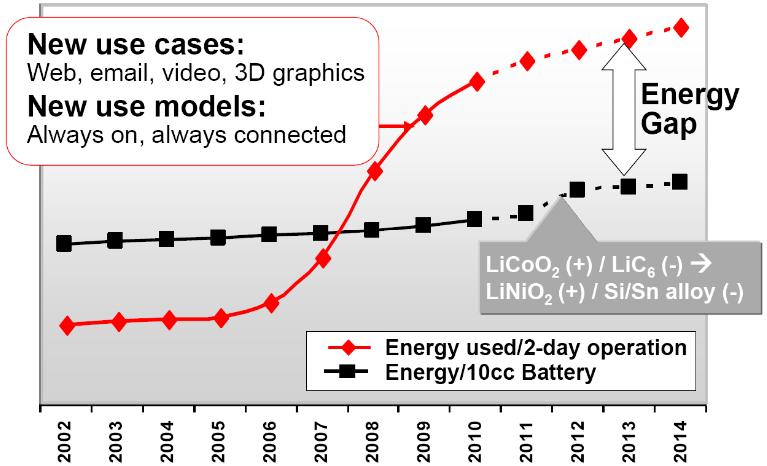
- Ever increasing functionalities & complexity

→ More performances required!

→ New SoC architectures are emerging:

Multicores, GP-GPU, Manycores…





Source: Öfversten [IWPC'08], Samms [IWPC'08]

- Battery technology evolution is slower

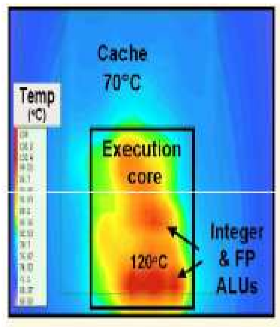
→ Power Management becomes a must!

SYSGO EMBEDDING INNOVATIONS

THALES

Power consumption impacts:

- Thermal dissipation
- Components reliability
- Cost



→ **Issue for the whole system!**

Power consumption reduction → Thermal dissipation reduction → Cooling devices reductions → Cost reduction (development & production)

Cooling devices reductions → Reliability increase → Reduction of maintenance cost

Cooling devices reductions → Weight reduction

SYSGO EMBEDDING INNOVATIONS

THALES

Introduction - TCS motivations

Objectives

Power consumption reduction of mono & multi processors embedded systems hosting real-time applications

Contribution to ↗ autonomy / ↘ consumption / ↘ thermal embedded products

Approach

Safe usage of HW platforms power saving knobs

Specification of OS services to monitor and control power consumption

Use a modular runtime environment with minimum impact on the SW application development

Environment enabling development of power mngt policies

Portable design on various OS and HW platforms

CMOS circuit power consumption breakdown:

$$\text{Power} = P = P_{Static} + P_{Dynamic}$$

Energy required for executing a Task during a time T

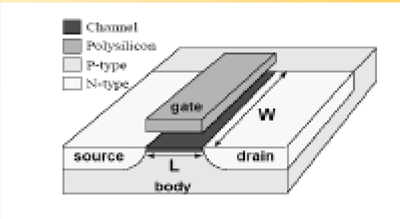
$$E = P \times T \propto V^2$$

Knowing: $T \propto F^{-1}$

$$P_{Static} \propto V_{dd} \times I_{leak}$$

With $I_{leak} \propto (W/L)e^{(-V_{TH}/T)}$

Evolve with CMOS technology and temperature (T)



⇒ Gains ∝ V

$$P_{Dynamic} \propto \alpha \times C \times V_{dd}^2 \times F$$

With
- α **: switching factor**
- **C : Capacitance**
- V_{dd} **: Supply voltage**
- **F : Frequency**

⇒ Gains ∝ V².F

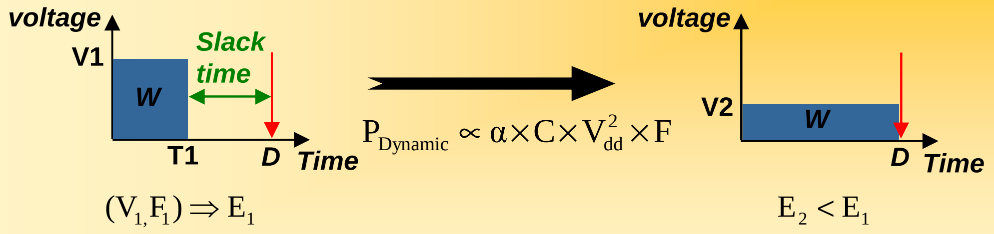
SYSGO EMBEDDING INNOVATIONS THALES

Introduction – power management techniques

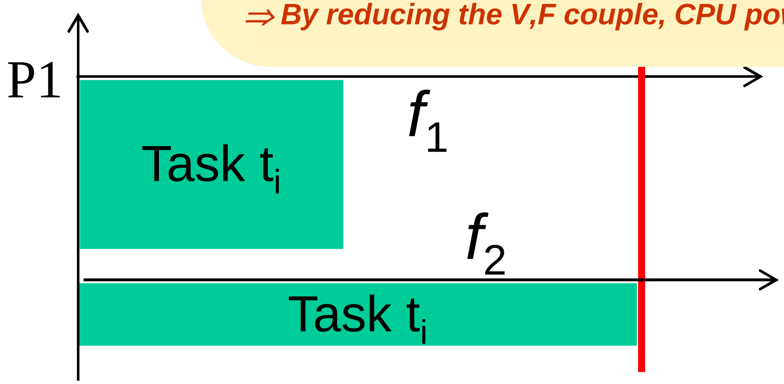
DVFS (Dynamic Voltage and Frequency Scaling) knobs:

- **HW mechanism consisting in varying the voltage/frequency couple**
- **Technique enabling the reduction of the processor power consumption by providing just the necessary energy to execute a job**

Theorical example: a task with a workload W which execution must be finished before its deadline D



⇒ By reducing the V,F couple, CPU power consumption can be reduced!



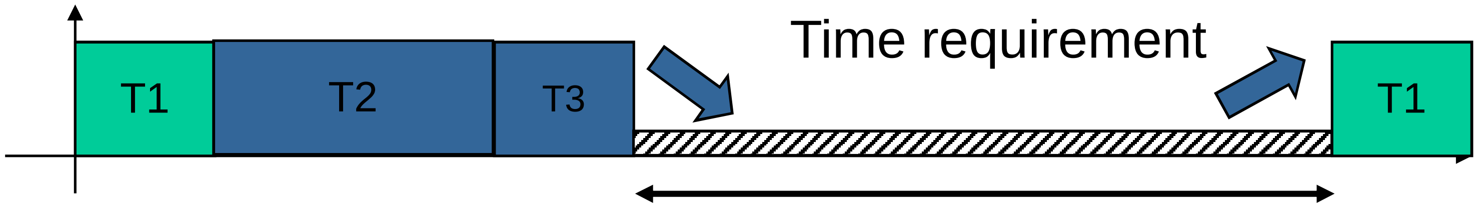
Slow down factor:
$S = f_1 / f_2$

Offline strategy
Global frequency (for all task) : based on the schedulability test according to the Worst Case Execution Time (WCET) of a task set

Online strategy
Local frequency (for each job) : based on the schedulability test,
The Worst Case Execution Time is substituted by the actual execution time → Lot of switch

DPM (Dynamic Power Management) knobs:

- Input voltage of the component is turned off
- Several low-power mode
  - Idle: data in cache are saved
  - Sleep: data are lost



T > Tsleep + Twakeup,

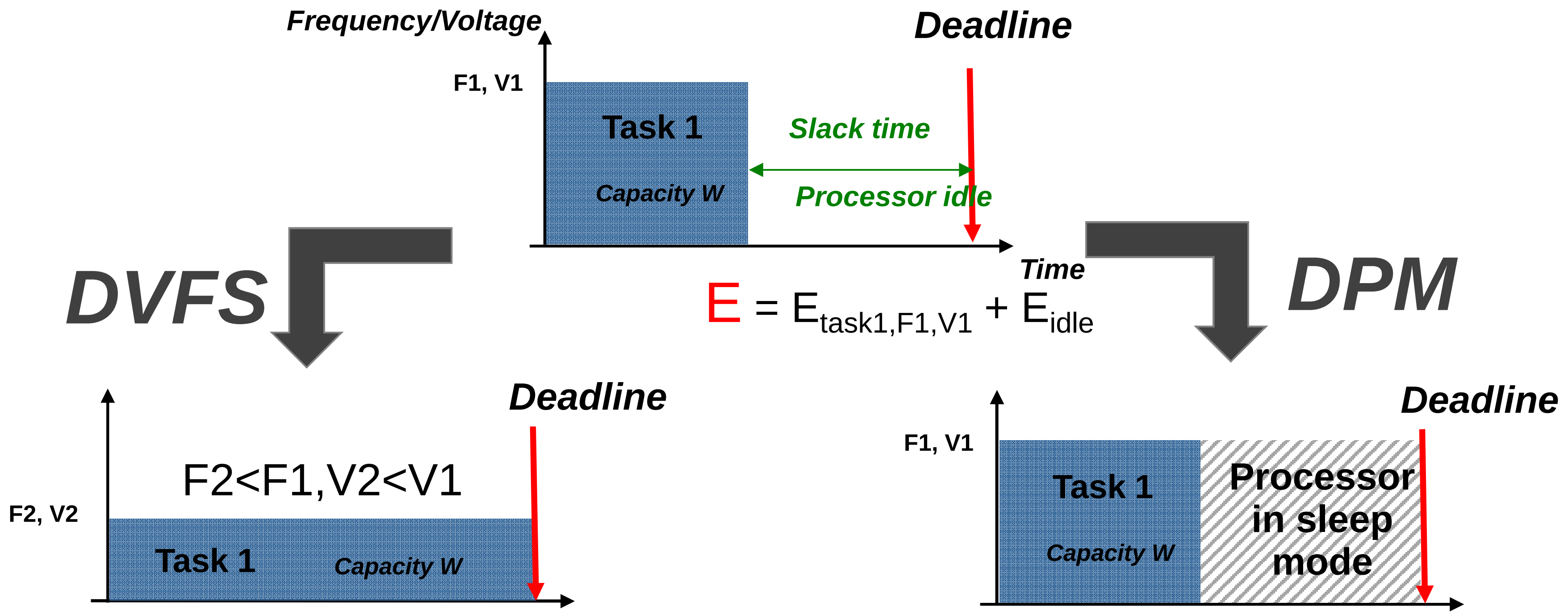
Data in the cache memory ?

Real time system :
Offline or online policy based on the schedulabitity test

Soft system :
Probability based estimations of inactivity time, period, duration

SYSGO EMBEDDING INNOVATIONS THALES

DVFS or DPM ?



$$E = E_{task1,F1,V1} + E_{idle}$$

$$E' = E_{task1,F2,V2} + E_{switch}$$
$$E' < E_{task1,F1,V1} < E$$

$$E'' = E_{task1,F1,V1} + E_{switch} + E_{sleep}$$
$$E'' > E_{task1,F1,V1} (>E') < E$$

.

Introduction – power management techniques

Application driven solution:

- Intrusive, not flexible, multi-application issues

OS driven:

- General purpose OS (Linux/Windows) solutions:
  - OS heuristics evaluate the processor load for a given past period and take actions if the processor reaches a specific workload threshold
  - No modifications are required for the applications to use this framework

But,

  - Not suited for real-time (soft & hard) applications
  - Usually disabled by system integrators to avoid system misbehaviour

→ System integrators need power management solutions compatible with real-time and critical systems! No standard solutions exist

SYSGO EMBEDDING INNOVATIONS THALES

Agenda

1. Introduction
    1. Context
    2. System integrators motivations
    3. Power management techniques overview

2. A Low-Power scheduler implementation
    1. Low-Power scheduling algorithm example
    2. End-User usage
    3. Implementation - A portable LowPower scheduler framework

3. PikeOS - scheduling for mixed critical systems
    1. PikeOS
    2. PikeOS scheduling with time partitionning

4. Low Power scheduling perspectives for mixed critical systems
    1. Extensions of PikeOS scheduler
    2. Integration at user VM
    3. PikeOS extensions

SYSGO EMBEDDING INNOVATIONS

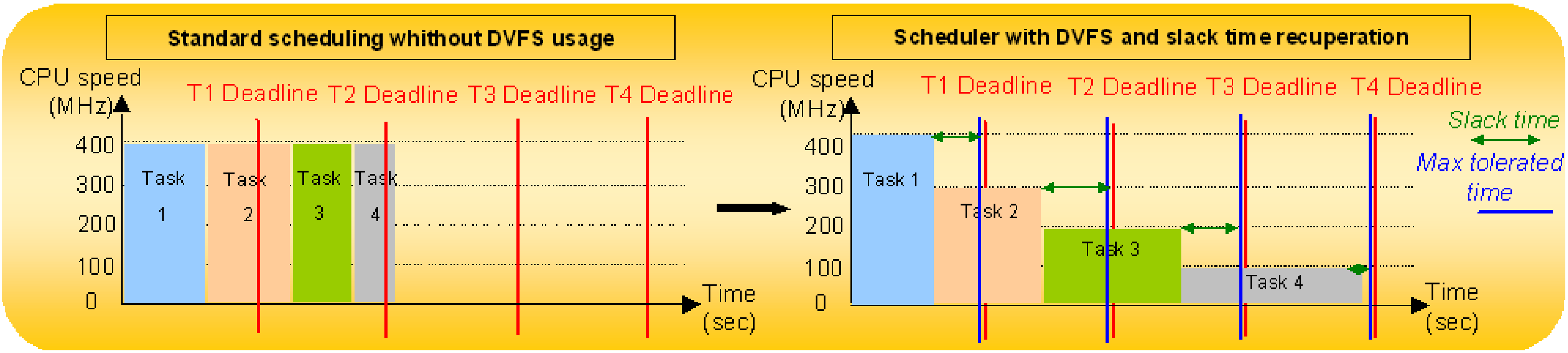
THALES

Low-Power scheduler implementation – algorithm example

Example of a Low-Power scheduling algorithm:

→ **EDF (Earliest Deadline First) policy combined with online DVFS usage**

- Task WCET is known/estimated for each CPU power modes
- Extend task execution upon a maximum tolerated time (deadline) and find a compliant CPU power mode
- At runtime: Slack time recuperation to take benefit of lower CPU power modes



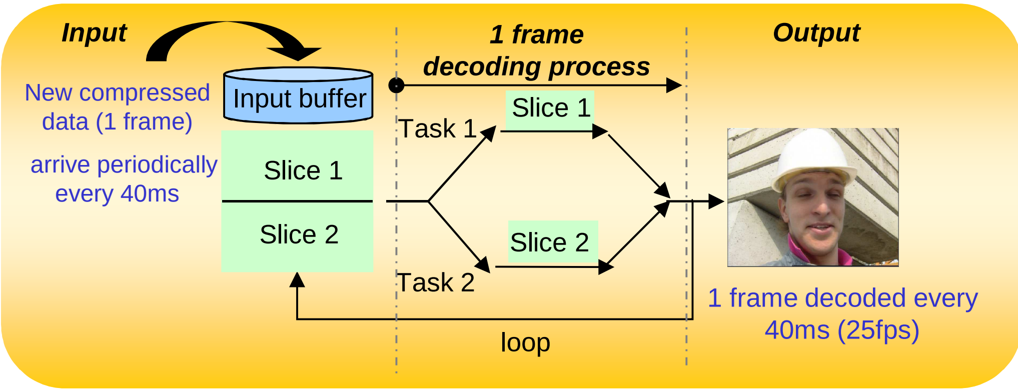
SYSGO EMBEDDING INNOVATIONS
THALES

Low-Power scheduler implementation – usage and limitations

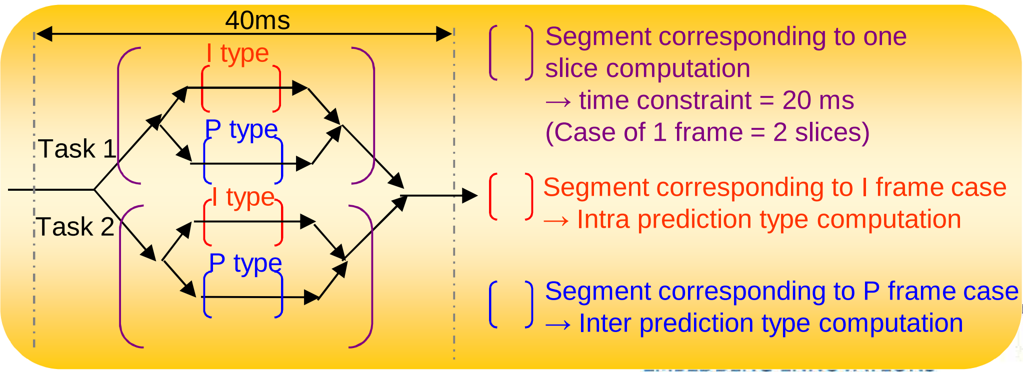
User interfaces:

- A proposal for POSIX extension to be able to hold extra information about timings for helping resource management policy development
- POSIX limitations:
  - Priority based only
  - doesn't take into account execution times
- Needs:
  - Concept to associate duration to concurrent flows: The thread segment

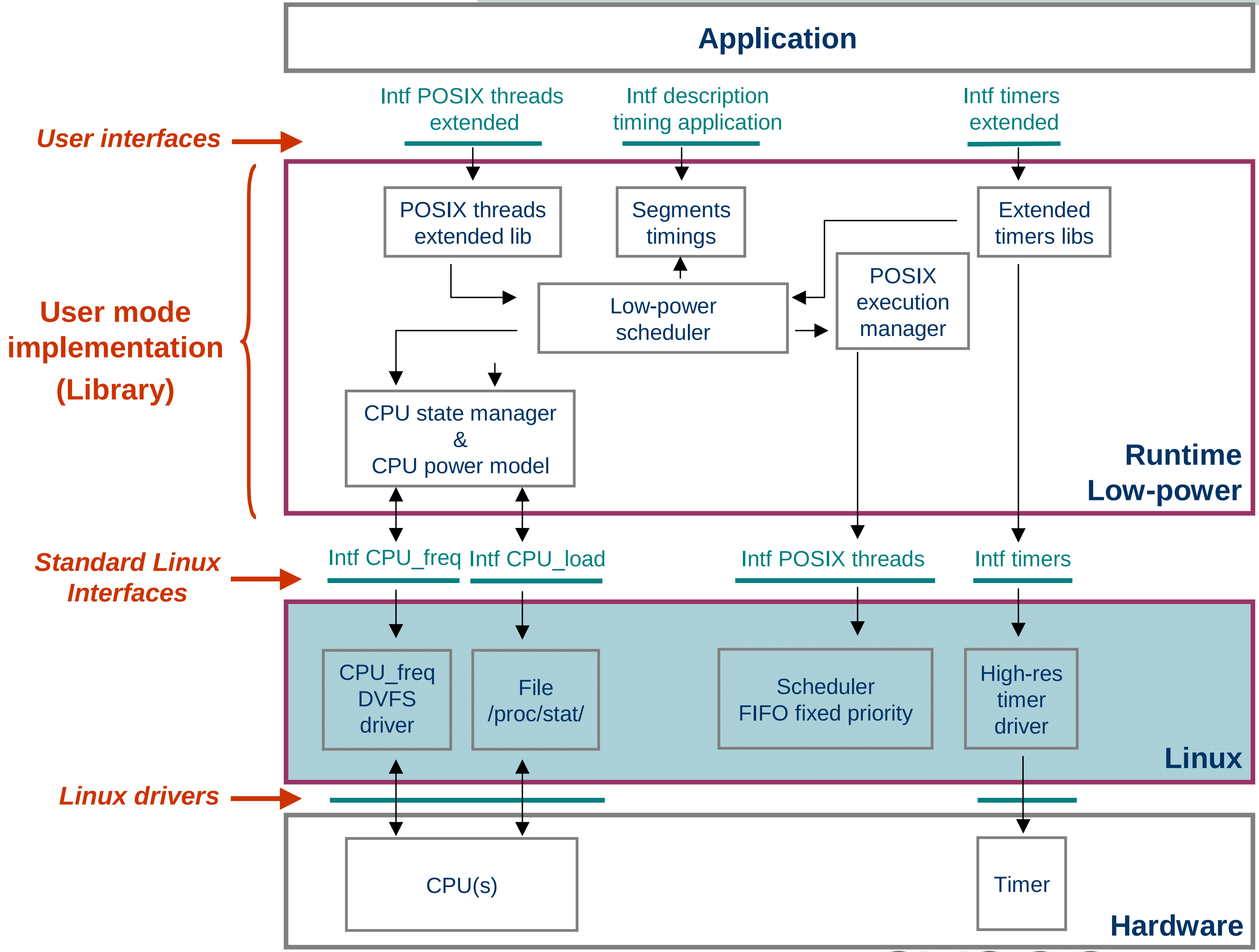
2 slices video streams overall decoding process



Application instrumentation Identify segments



Low-Power scheduler implementation - Low Power scheduler Framework (Patent)



Plugable approach

©FR2969776

SYSGO EMBEDDING INNOVATIONS

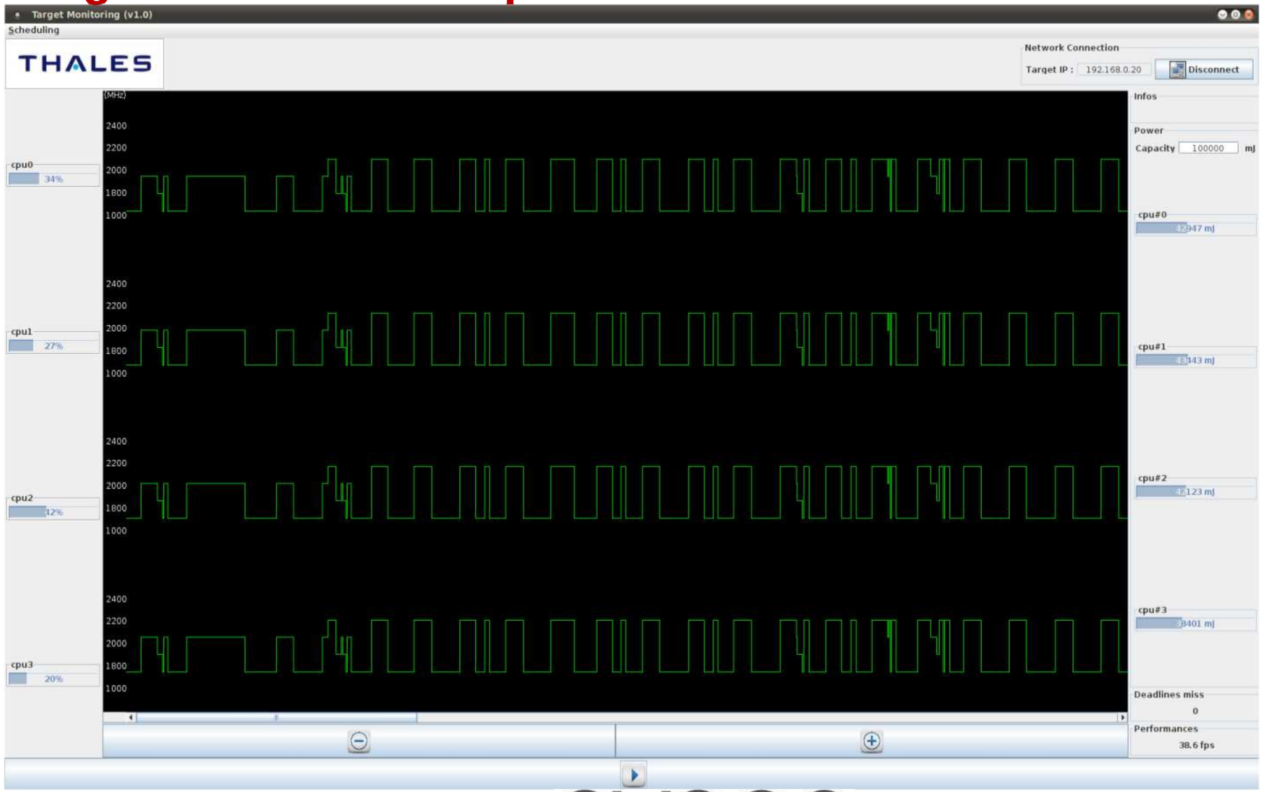
THALES

Low-Power scheduler implementation – results and limitations

Results on OMAP3530 with H.264 video codecs:

- ~30% power consumption reduction measured on OMAP3530
- Satisfying results on soft real-time applications
- Multiprocess/Multi application issues

→ **Future work: Integration in Kernel space or as a service of RTOS**



SYSGO EMBEDDING INNOVATIONS THALES

Agenda

1. Introduction
   1. Context
   2. System integrators motivations
   3. Power management techniques overview

2. A Low-Power scheduler implementation
   1. Low-Power scheduling algorithm example
   2. End-User usage
   3. Implementation - A portable LowPower scheduler framework

3. PikeOS - scheduling for mixed critical systems
   1. PikeOS
   2. PikeOS scheduling with time partitionning

4. Low Power scheduling perspectives for mixed critical systems
   1. Extensions of PikeOS scheduler
   2. Integration at user VM
   3. PikeOS extensions

PikeOS: real-time virtualisation plattform

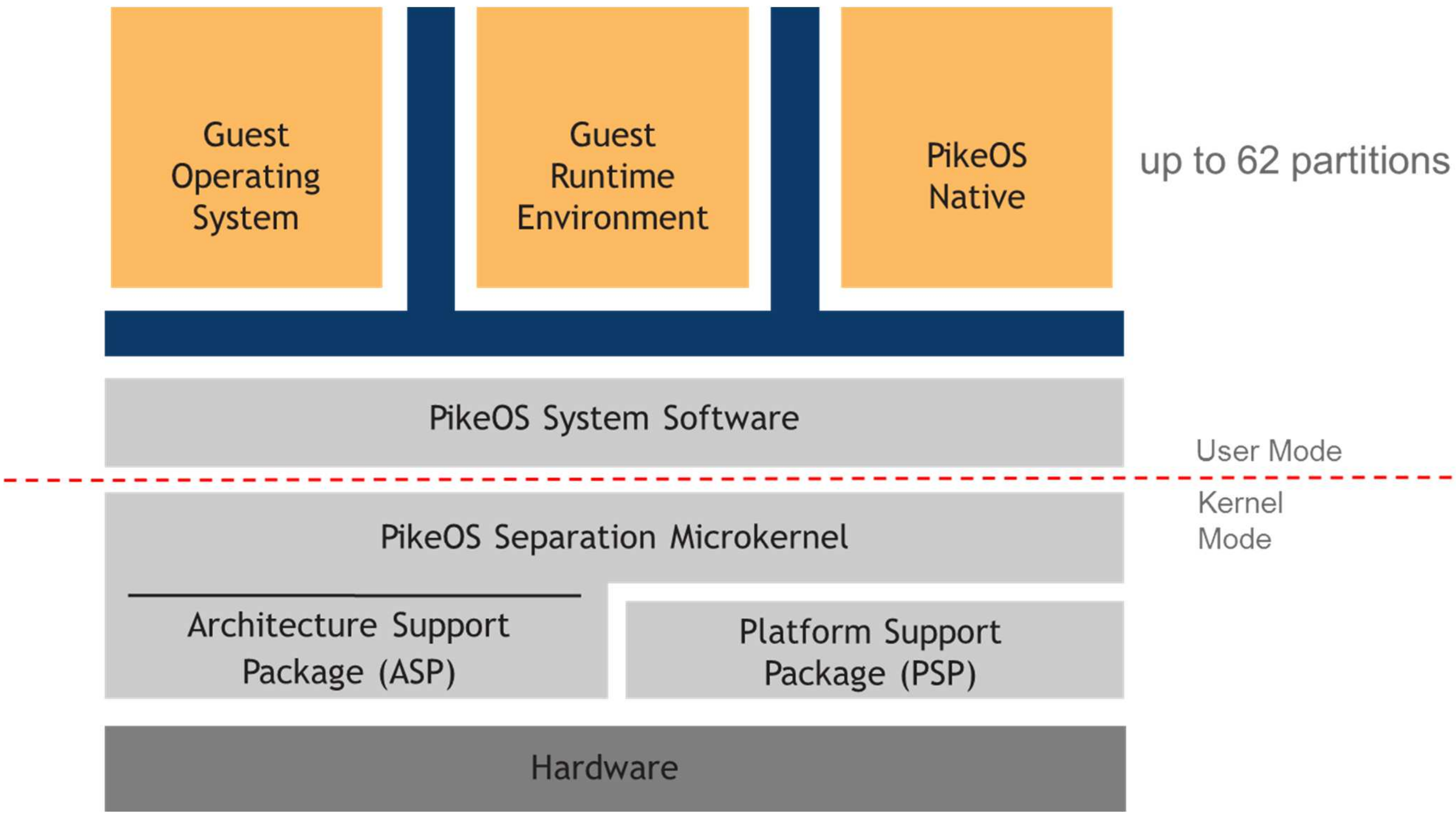
The safe and secure virtualization (SSV) RTOS:

- runs concurrently software of different safety and security levels …
- can provide multiple API, run time environments and guest operating systems …
- enables a mixture of hard real-time and non real-time applications …

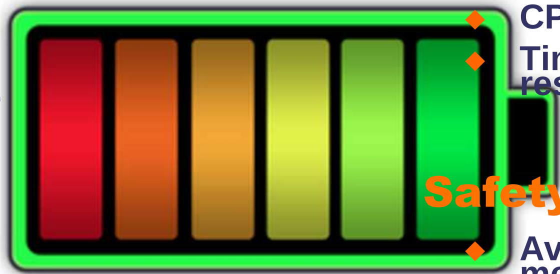
… on a single embedded device

- Certified technology
- Enables modular certification according to highest industrial standards
- Runs on numerous HW architectures and platforms
- Provides multi-core functionally

PikeOS Architecture



POWER is also shared



Challenge: Resources sharing

Resources

- CPUs
- Memory, IO memory
- Flies, drivers, devices, buses…

Safety

- Integrity, availability
- Isolation, application errors, fail safe

Security

- Integrity, availability, confidentiality
- Possible side channels via shared resources
- Resources and API are attack surface

PikeOS Solution

Resource Partitioning

Challenge: Time sharing

Time

- CPU cycles
- Time effects of accessing shared resources, e.g. buses…

Safety

- Availability, deterministic behavior, meeting deadlines
- Right balance between time- and event-triggered tasks

Security

- Availability, confidentiality
- Possible timing side channels via shared resources, e.g. caches, busses
- Time is the attack surface

PikeOS Solution

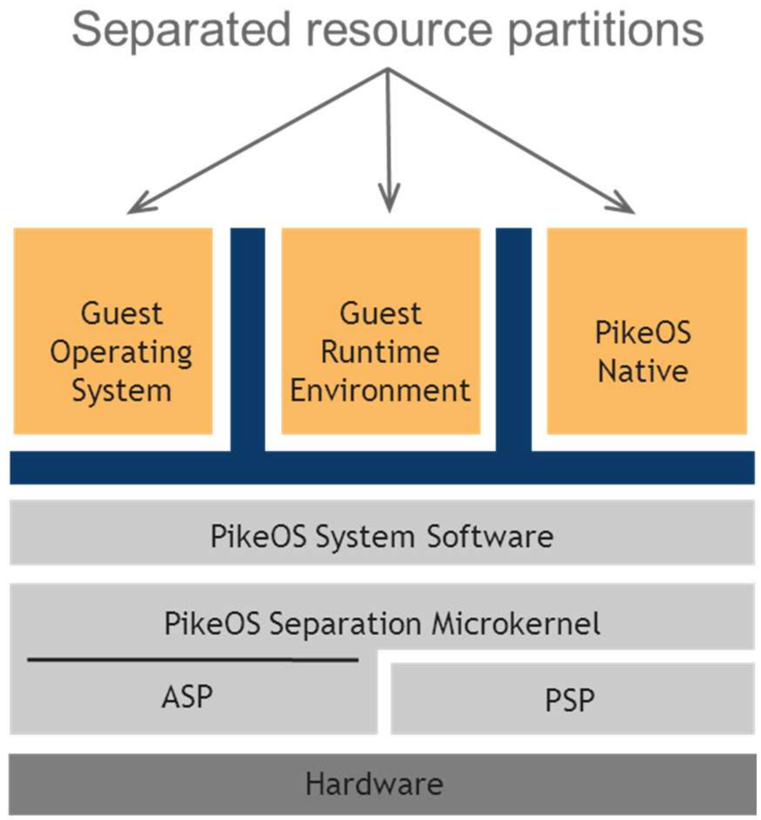
Time Partitioning

SYSGO EMBEDDING INNOVATIONS

THALES

Resource Partitioning - Spatial Separation

- Static allocation of all system resources
- Application has guaranteed access to assigned resources
- Applications cannot access resources of other partitions if not explicitly configured otherwise
- No error propagation throughout other partitions
- Memory protection enforcement using Hardware (MMU)
- All partitions execute in user modes

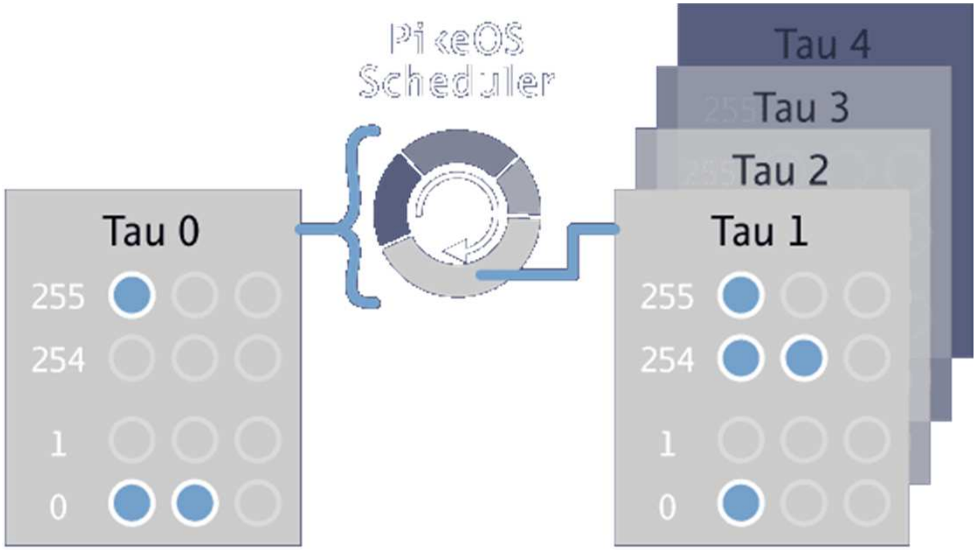


SYSGO EMBEDDING INNOVATIONS
THALES

Time Partitioning - Temporal Separation

Time partition principles

- **ARINC 653 compliant**
  - Static configuration of execution order and duration
- **Future ARINC 653 requirements**
  - Support for multiple scheduling schemes
  - Scheduling schemes can be switched during runtime
- **Partition '0' offers additional functionality**
  - Threads with high priority can preempt active partition
  - Threads with low priority can act as global idle-job



SYSGO EMBEDDING INNOVATIONS THALES

Agenda

1. Introduction
   1. Context
   2. System integrators motivations
   3. Power management techniques overview

2. A Low-Power scheduler implementation
   1. Low-Power scheduling algorithm example
   2. End-User usage
   3. Implementation - A portable LowPower scheduler framework

3. PikeOS - scheduling for mixed critical systems
   1. PikeOS
   2. PikeOS scheduling with time partitionning

4. Low Power scheduling perspectives for mixed critical systems
   1. Extensions of PikeOS scheduler
   2. Integration at user VM
   3. PikeOS extensions

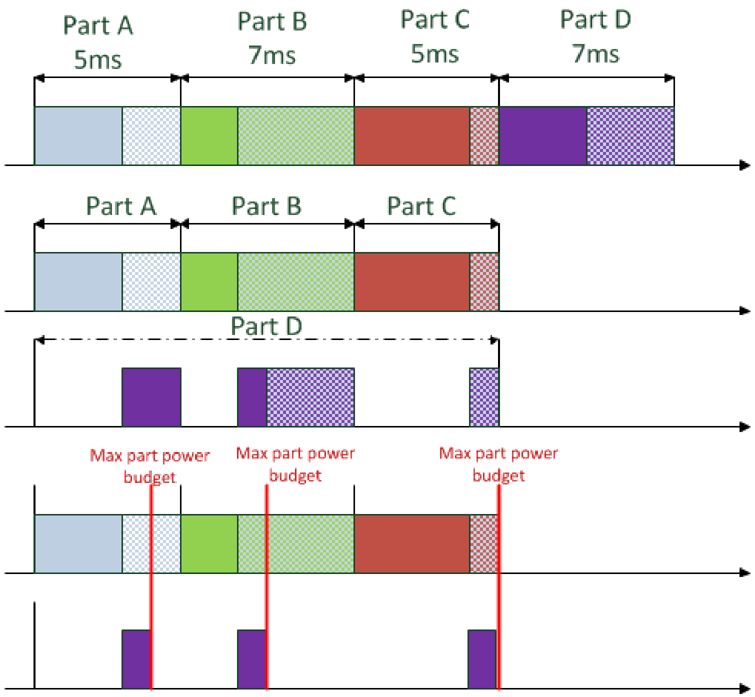
Option 1: Integration as extensions of PikeOS scheduler

- **A new parameter must be introduced for the VMs and handled by PikeOS scheduler:**
  - → Max power budget to be consumed for a VMs during its time partition
  - → It only impacts the background time partition execution duration, and/or low priority VMs

Traditional ARINC653 scheduling

Scheduling with background time partition

Scheduling with background time partition and limited power budget



SYSGO EMBEDDING INNOVATIONS THALES

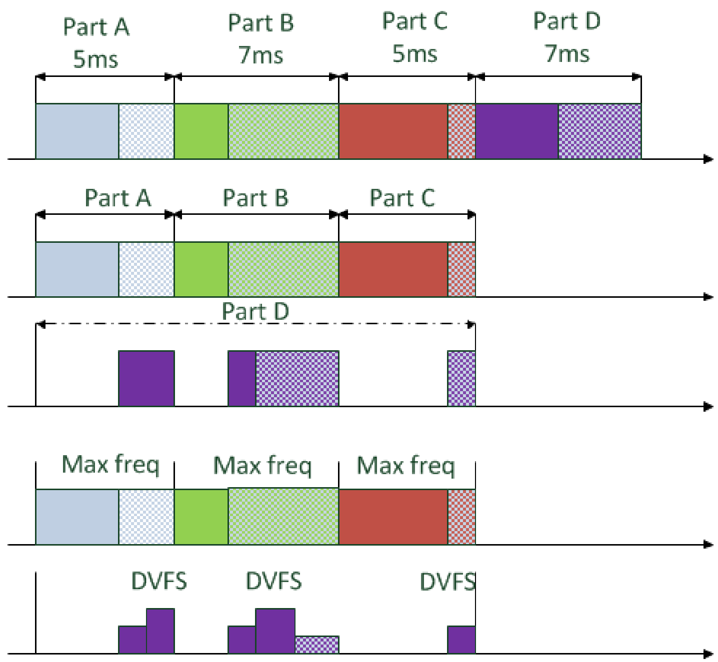
Option 2: Integration at level of user VM

- **Apply DVFS/DPM only on Low Priority VMs**
  - Low-priority VMs are allocated to background time partition
- **A new parameter must be handled by PikeOS scheduler: CPU frequency**
  - This is required to ensure the right power mode is used when High priority VMs are scheduled

Traditional ARINC653 scheduling

Scheduling with background time partition

Scheduling with background time partition and DVFS management on this low-priority VMs



SYSGO EMBEDDING INNOVATIONS THALES

Option 3: Integration at the level of PikeOS extensions

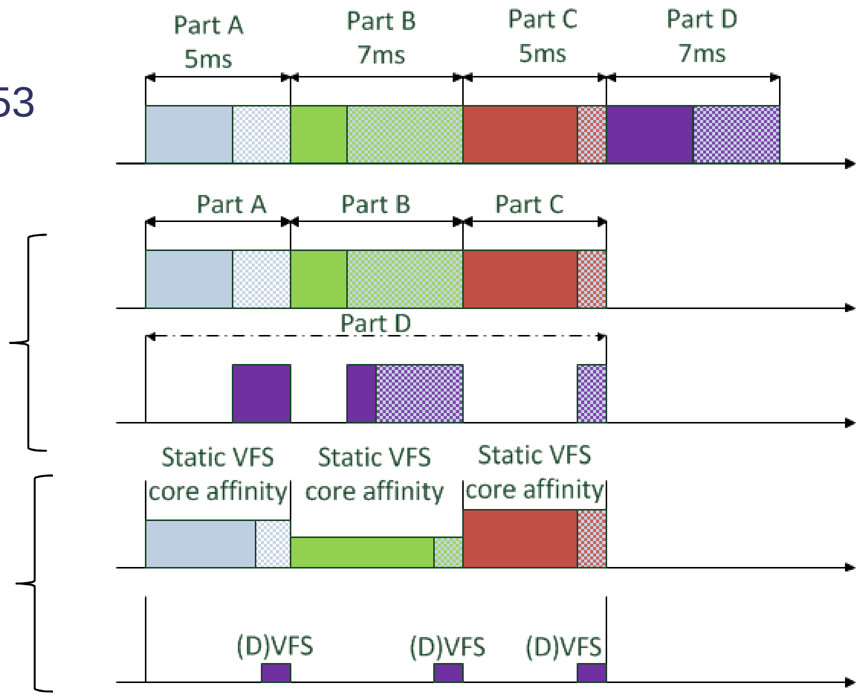
- Allocate high performance cores to critical VMs
- Allocate low performance cores to low priority VMs

→**The system integrator must provide the number of cores and/or frequencies to be used for each partitions to ensure deterministic behavior of the application**

Traditional ARINC653 scheduling

Scheduling with background time partition

Scheduling with background time partition and static VFS and core affinity for high priotiy VMs



SYSGO EMBEDDING INNOVATIONS THALES

Thank you for your attention!

SYSGO EMBEDDING INNOVATIONS

THALES