Concurrent Systems

3D4 ⟷ CS2016

Operating Systems

Andrew Butterfield
ORI.G39, Andrew.Butterfield@scss.tcd.ie

Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

with thanks to Mike Brady

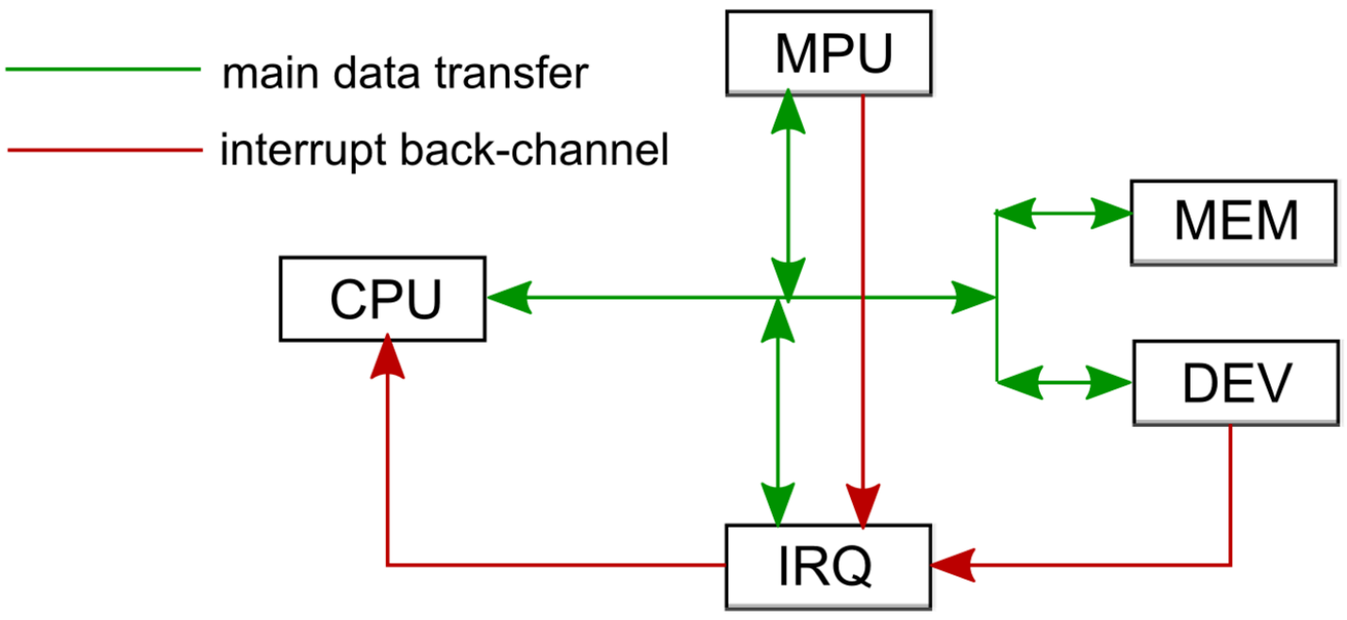
recent OS research @SCSS

- Applying formal methods to OS verification
- Why?
  - Testing is king, but requires test-setup repeatability.
  - OS behaviour is effectively non-deterministic:
    - mainly because of interrupt unpredictability
    - hard to repeat tests
    - Multicore makes things worse

From FMEIMAKQP(?) final presentation

Unavoidable Concurrency
– even with Single-Core!

- In a single-core system, the CPU is time-shared between the hypervisor and partition code with no parallelism.
- Flow control change is managed via traps (exceptions, interrupts, ...).
- However, we have an essentially concurrent system



- CPU executes instructions in a sequential manner on behalf of either the kernel or partition
- Memory (MEM) responds to CPU memory requests
- The MMU/MPU observes the bus traffic, raises memory fault interrupts when appropriate
- IO Devices (DEV) signal via interrupt when done
- Interrupt request hardware (IRQ) takes in interrupt requests from MMU/MPU/DEV/CPU and forwards the highest in priority to the CPU

Formal Methods?

- Use of mathematical logics to:
  - Model application domains
  - Give precise semantics to requirements, specification and programming languages
  - Allow proofs of program correctness
  - Provide a rigorous foundation for developing analysis tools
    - e.g,, model-checkers, like Promela/SPIN.

What kinds of OS?

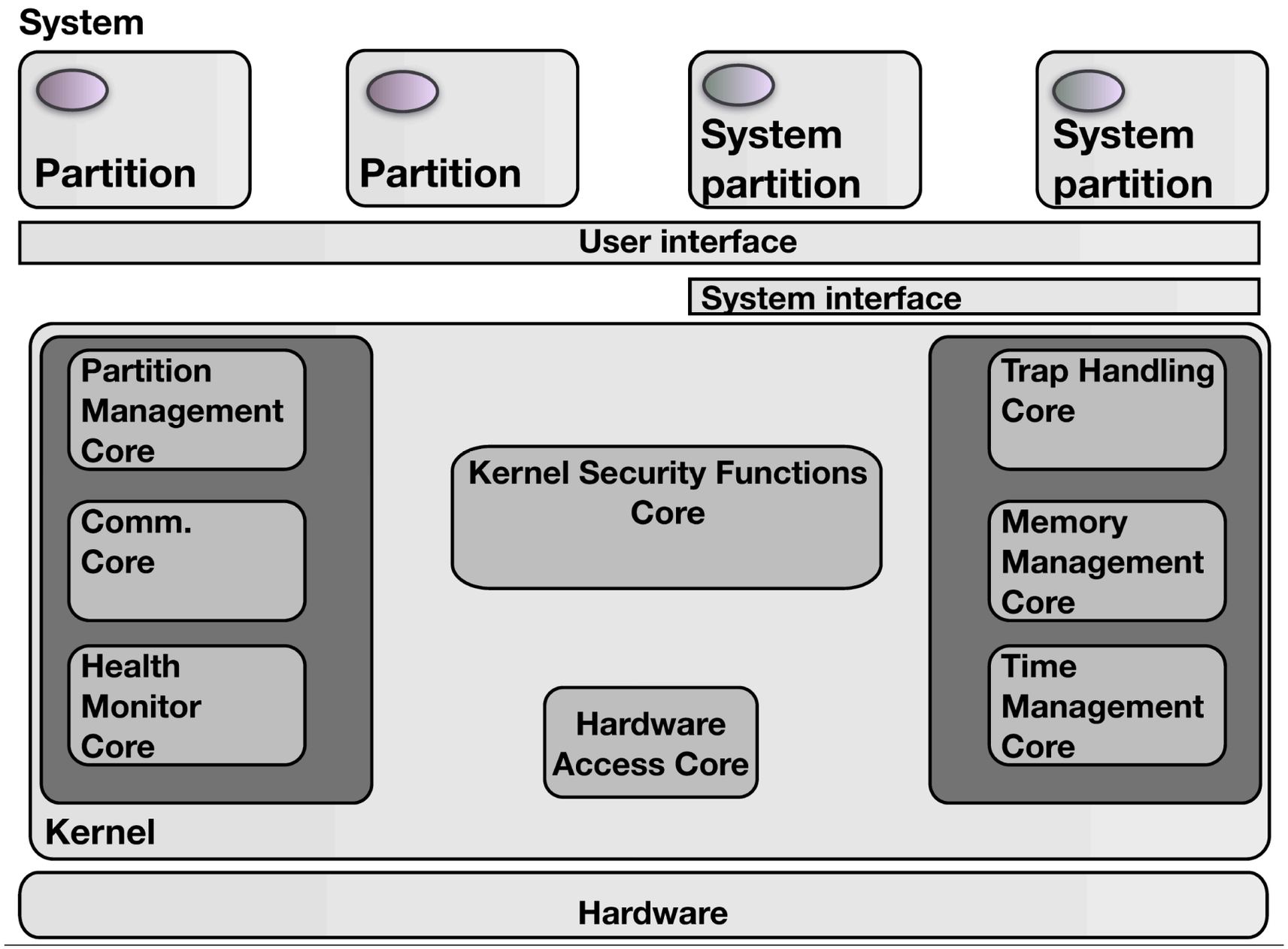
- Real-time OSes used in spacecraft
  - research/consultsncy for European Space Agency
- Time/Space Partitioning (TSP) Kernels
  - a form of “hypervisor”
  - two projects: MTOBSE, FMEIMAKQP
- Real-Time Executive for Multiprocessor Systems (RTEMS)
  - project starting: RTEMS-SMP

Lero - the Irish Software Research Centre

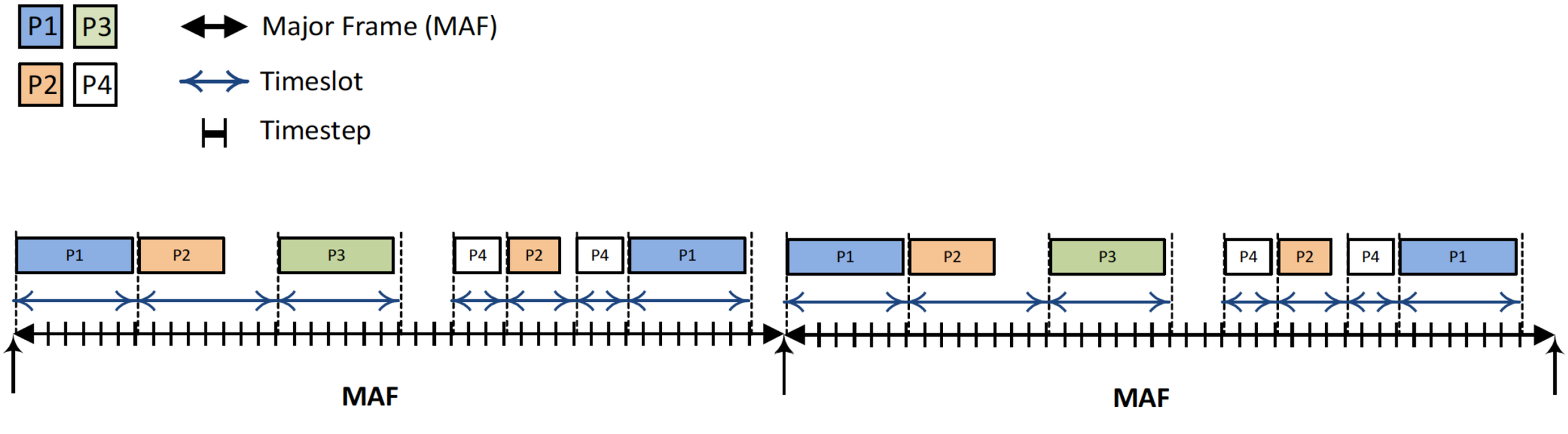
- SFI-funded Research Centre
- Involves all 7 universities in Ireland, as well as some of the IT sector
  - Headquarters in Limerick
- Strong industrial focus
- Main point of contact with ESA

Lero THE IRISH SOFTWARE RESEARCH CENTRE

Time/Space Partitioning Kernel



Fixed TSP Scheduling



"Schedulability" analysis is crucial in order to work out the fixed timings above!

IMA Separation Kernel Qualification Project
ESA Contract N°: 4000111495/14/NL/GLC/al
D02: Partitioning Kernel Requirements Baseline

Excerpts from an TSP formal model

```
type_synonym tfwWLength= nat
type_synonym partitionID = nat
type_synonym tfwOffset = nat
type_synonym timeFrameWindow = "(partitionID * tfwWLength * tfwOffset)"
type_synonym majorTimeFrame =  "timeFrameWindow list"
type_synonym TFWid = "nat"
datatype schedule = Sch TFWid majorTimeFrame
...
definition
  scheduler :: "unit s_monad"
where
  "scheduler ≡ do
     parMan ← gets partitionManager;
     schedule ←  return (partitionSchedule parMan);
     nextTFWNonIdle ← nextTFW(schedule);
     schedule' ← setTFWSchedule schedule nextTFWNonIdle;
     modify (λs . s(|partitionManager := (partitionManager s)(|partitionSchedule := schedule'|)|));
      s ← get;
     archSwitchToPartition
      getCurrentTFW(partitionSchedule(partitionManager s))
   od"
...
lemma sanitization:
"∃ s, s' ∈ state . s' = scheduler s ⇒ ∀ r ∈ (cpu_regs (cpu_state s')) . r=0"
by (simp_all add:archSwitchToPartition add:scheduler)
```

Using CSP to model hardware

CSP Model of MMU

```
MMU(blocked)
=   tick -> ( TRANSFER(blocked)
            [] BLOCK_MEM_REGION(blocked)
            [] UNBLOCK_MEM_REGION(blocked)
            [] MMU(blocked) )

TRANSFER(blocked)
= bus?dir?addr ->  ( if member(addr, blocked)
                     then ( badaccess -> raise!memfault -> MMU(blocked) )
                     else ( mmuOK -> MMU(blocked) )
                   )

BLOCK_MEM_REGION(blocked)
= mmuBlock?a -> MMU(union(blocked,{a}))

UNBLOCK_MEM_REGION(blocked)
= mmuUnBlock?a -> MMU(diff(blocked,{a}))
```

- Synchronise with global clock (`tick`)
- Accepts bus read or write (`bus?dir?addr`)
- Permits operation if address not blocked (`blocked, mmuOK`)
- Objects if address is blocked (`blocked,badaccess ,raise!memfault`)

CSP: Communicating Sequential Processes, developed by C.A.R. Hoare

Has powerful model-checker called FDR (Failures-Divergences Refinement)

RTEMS, "improved"

- RTEMS: open-source real-time operating system (rtems.org)
  - Widely used
- RTEMS Improvement
  - A version of single-core RTEMS qualified for spaceflight
    - done for ESA by Thales Edisoft (Portugal)
  - Code in RTEMS repository, qualification data owned by Edisoft.

RTEMS-SMP

- In 2013, two good realtime scheduling algorithms for multicore emerged
  - Multiprocessor resource sharing Protocol (MrsP)
    - A. Burns and A.J. Wellings, U. of York
  - O(m) independence preserving protocol (OMIP)
    - Bjorn B. Brandenburg, Max Planck Institute for Software Systems
- Within a few years, an RTEMS version was written by an employee of a German company, Embedded Brains
  - called RTEMS-SMP, part of the RTEMS repo, to be in the next major release.

RTEMS-SMP pre-qualification

- ESA want a flight qualified version of RTEMS-SMP
- Edisoft and Embedded Brains, along with a German instrument maker called Jena Optronik formed a consortium to do this.
- ESA invited key formal methods researchers from *Lero*
  - Andrew Butterfield Lero@TCD
  - Mike Hinchey Lero@UL, also NASA consultant.
- 2-year project, started mid-Feb 2019

Formal Aspects of RTEMS-SMP

- Main focus: the use of MrsP and OMIP in RTEMS
  - static analysis tools (coverity, infer,…)
  - model checking (SPIN, TLC, FDR)
  - Correctness Proofs (Frama-C, TLA+, …)