embeddedworld2017
Exhibition&Conference
... it´s a smarter world

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

A Monitoring Based Safety Architecture for Multicore Microcontrollers

Thomas Barth (h-da)
Prof. Dr.-Ing. Peter Fromm (h-da)

Contents

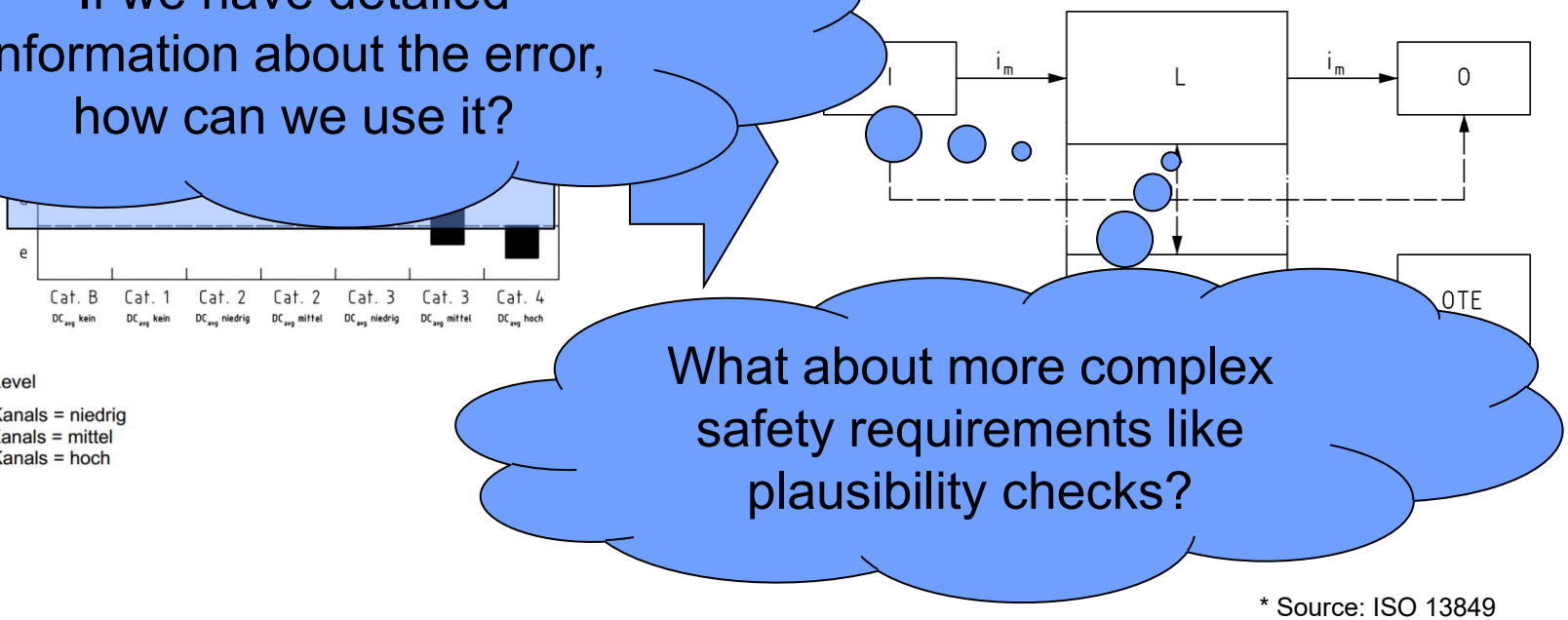
- Safety requirements
- Multicore Architectures
- Signal flow
- Error escalation
- Runtime Environment
- Timing behavior
- Conclusion

Safety requirements

It is not clear if multicore microcontrollers can be seen as redundant datapath.

Therefore focus on single path architecture with high diagnostic coverage.

If we have detailed information about the error, how can we use it?



Legende

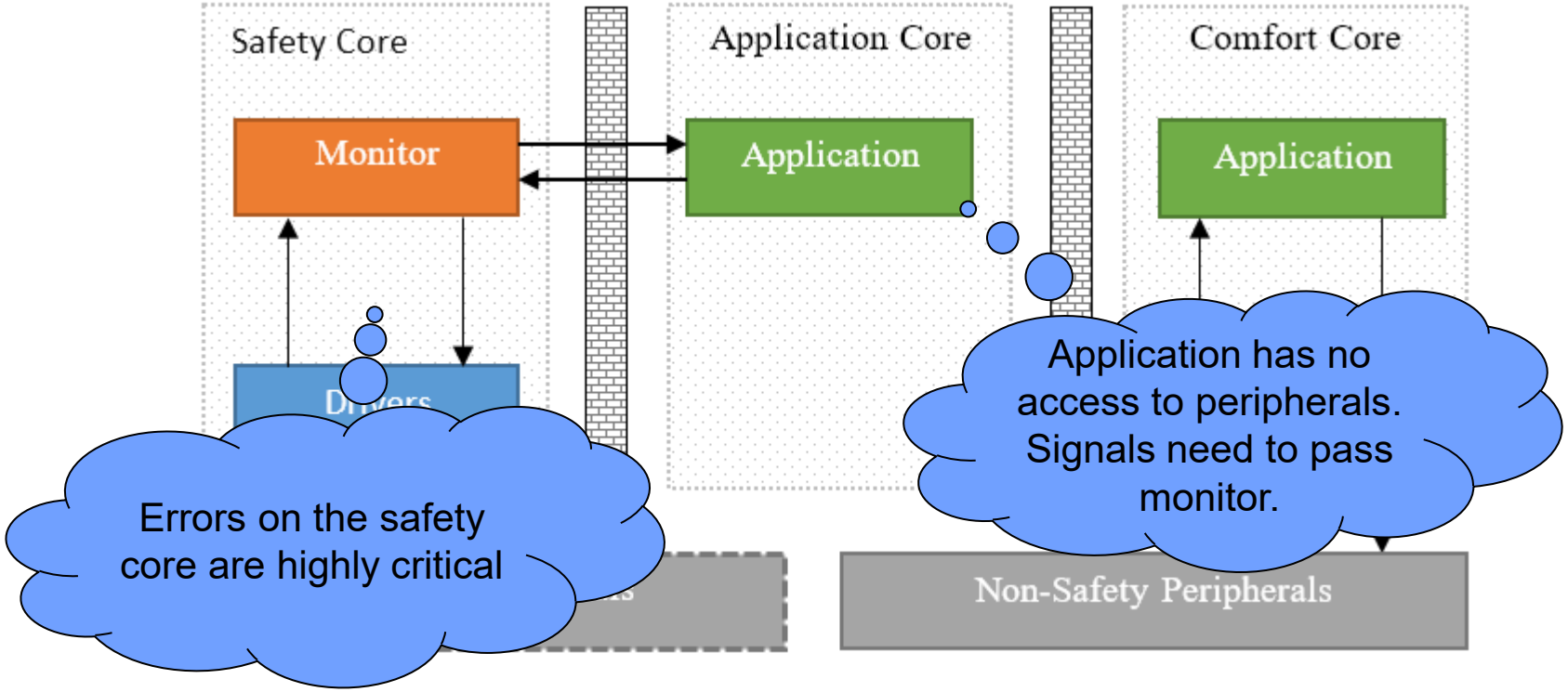
PL	Performance Level
1	$MTTF_d$ jedes Kanals = niedrig
2	$MTTF_d$ jedes Kanals = mittel
3	$MTTF_d$ jedes Kanals = hoch

What about more complex safety requirements like plausibility checks?

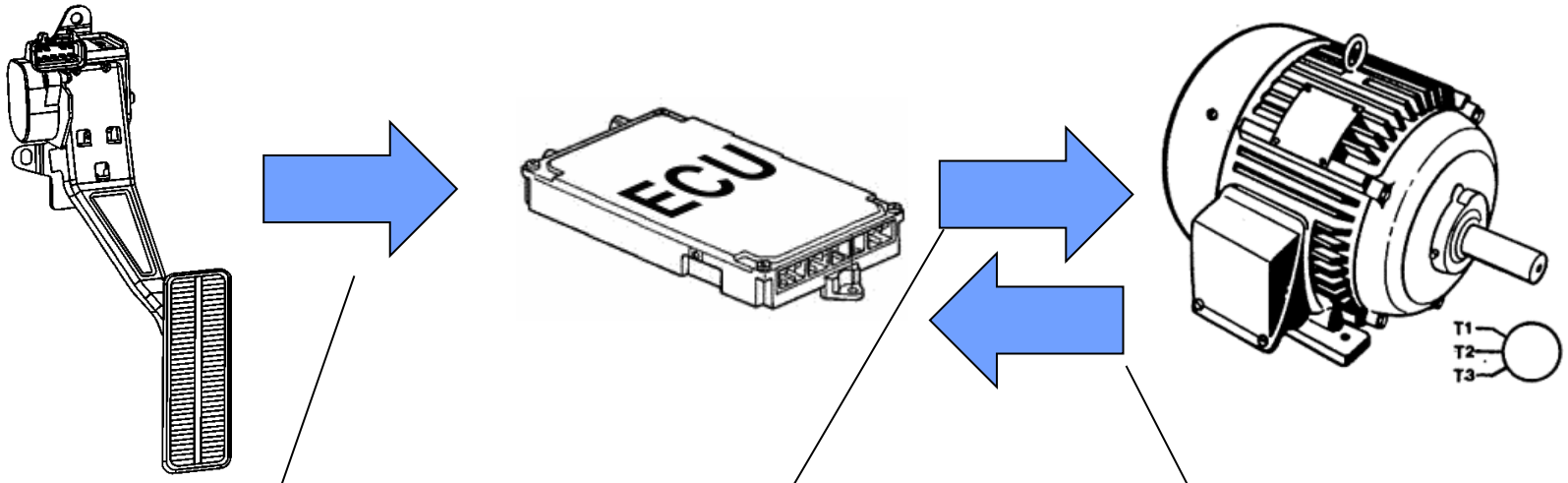
* Source: ISO 13849

Multicore Architectures

While on single core microcontrollers one core is responsible for executing safety and non-safety code, multicore microcontrollers allow assignment of specific tasks to dedicated cores.



Signalflow



Is the value in Range? Are complementary signals plausible?

Is the calculated output plausible in relation to the input Signal?

Was the setpoint received and accepted?

Are the ECU or other safety critical components faulty?

Safety Concept

The overall safety requirement is: No matter, what goes wrong with the hardware and / or the application software, the car may never get into a dangerous state – means, driving without control.

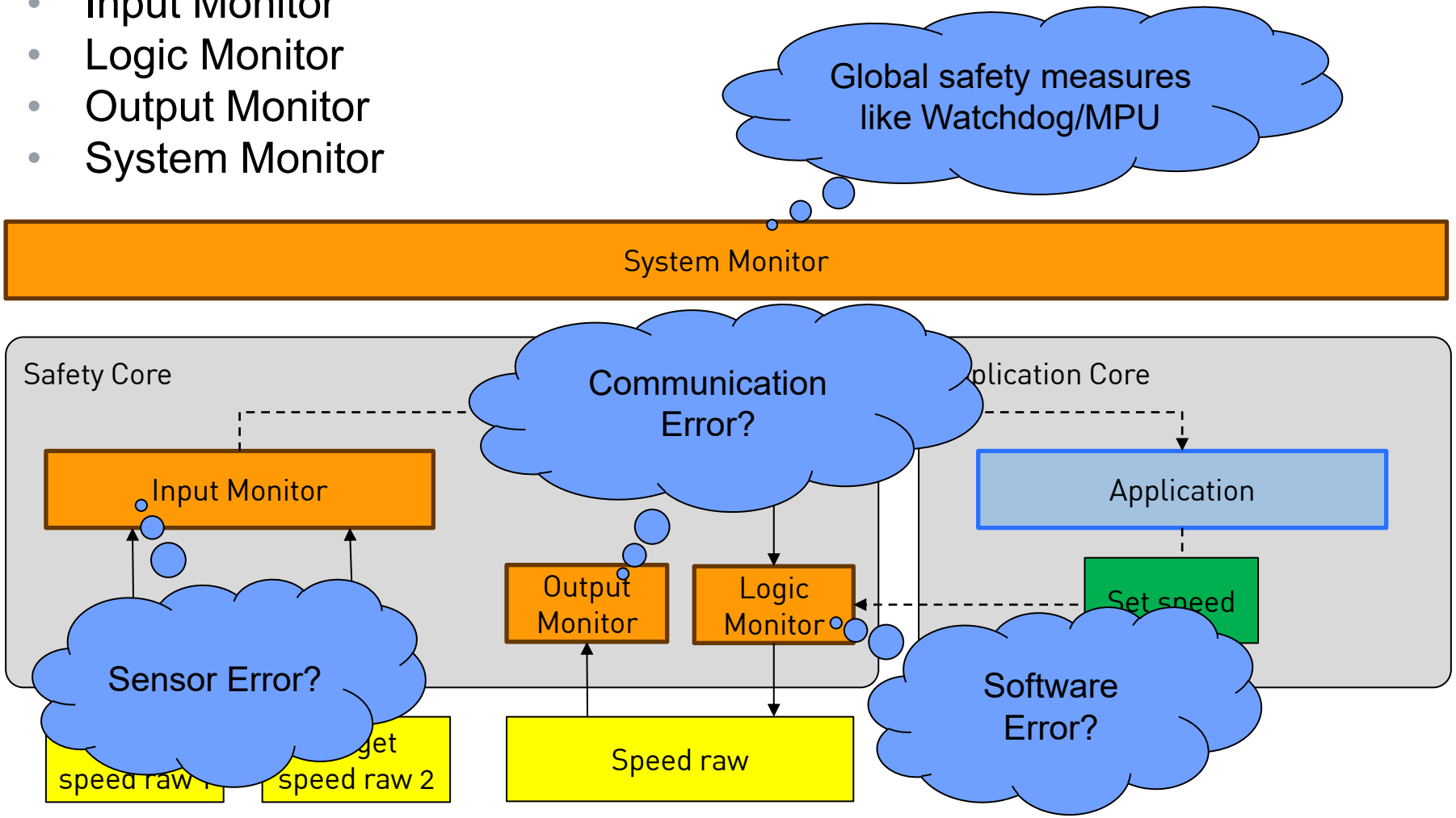
This is achieved by providing a high diagnostic coverage level by using 4 monitors (each containing various safety functions)

- The **Input Monitor** checks the validity of all sensor input signals and user commands
- The **Logic Monitor** checks the validity of all calculated control signals
- The **Output Monitor** compares the generated engine signal with the physical speed
- The **System Health Monitor** checks all system modules, e.g. battery, CAN state,...

Signalflow

Implemented Monitoring Architecture utilizes 2 cores and features:

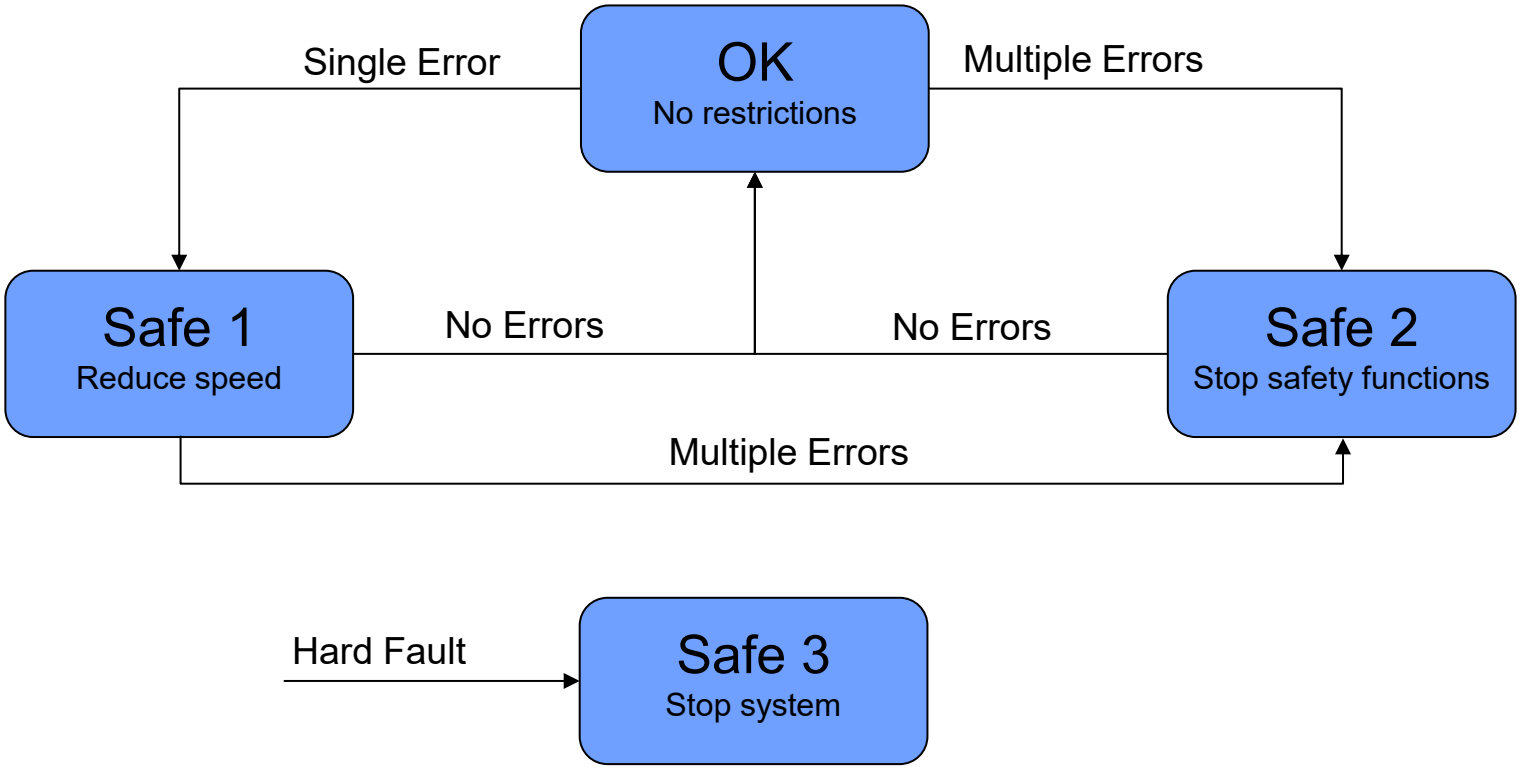
- Input Monitor
- Logic Monitor
- Output Monitor
- System Monitor



Exemplary error handling strategy

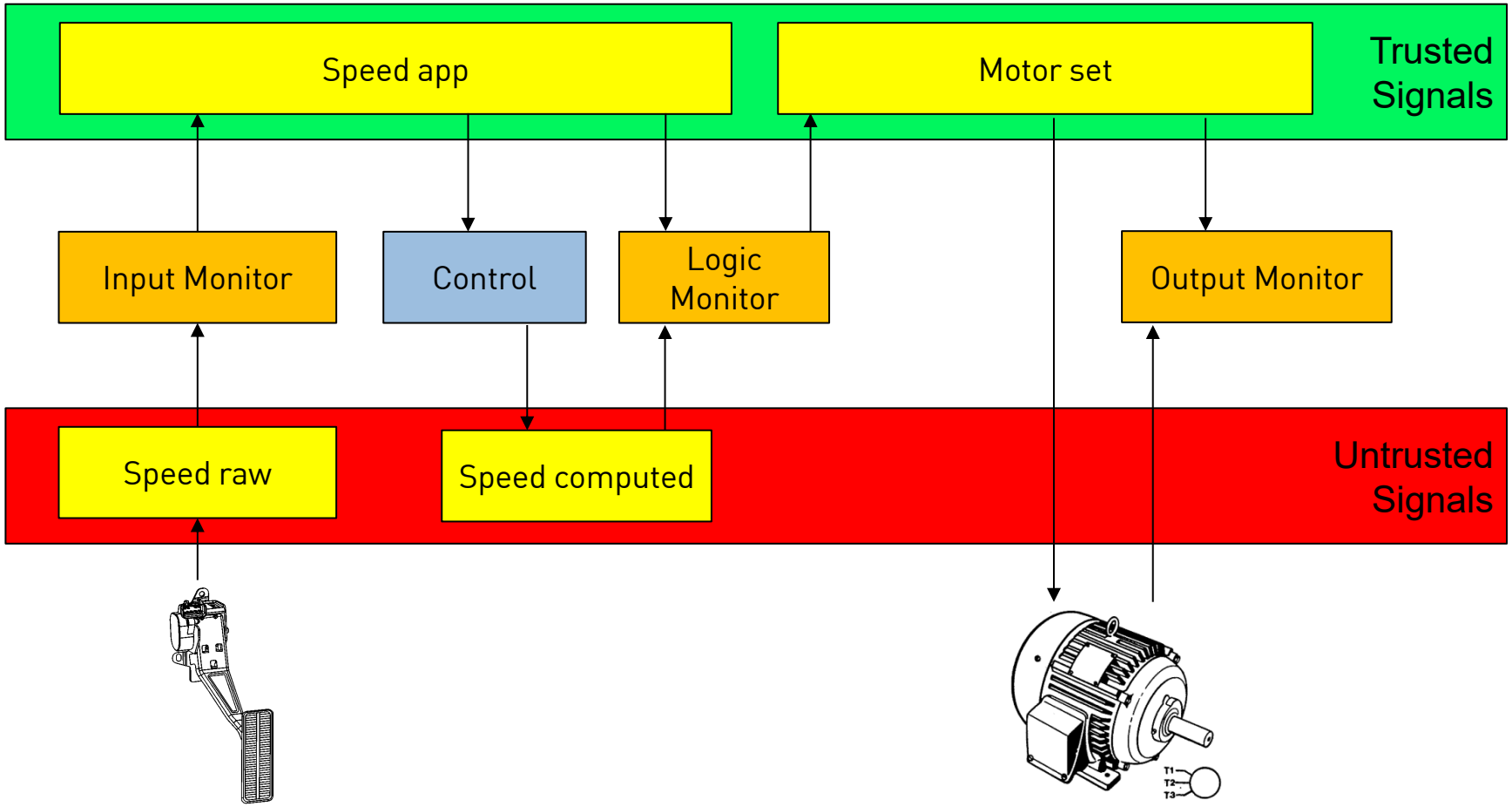
Safety requirements:

- Only calculate output if sensor values are valid
- Zero position of the accelerator is needed before vehicle moves



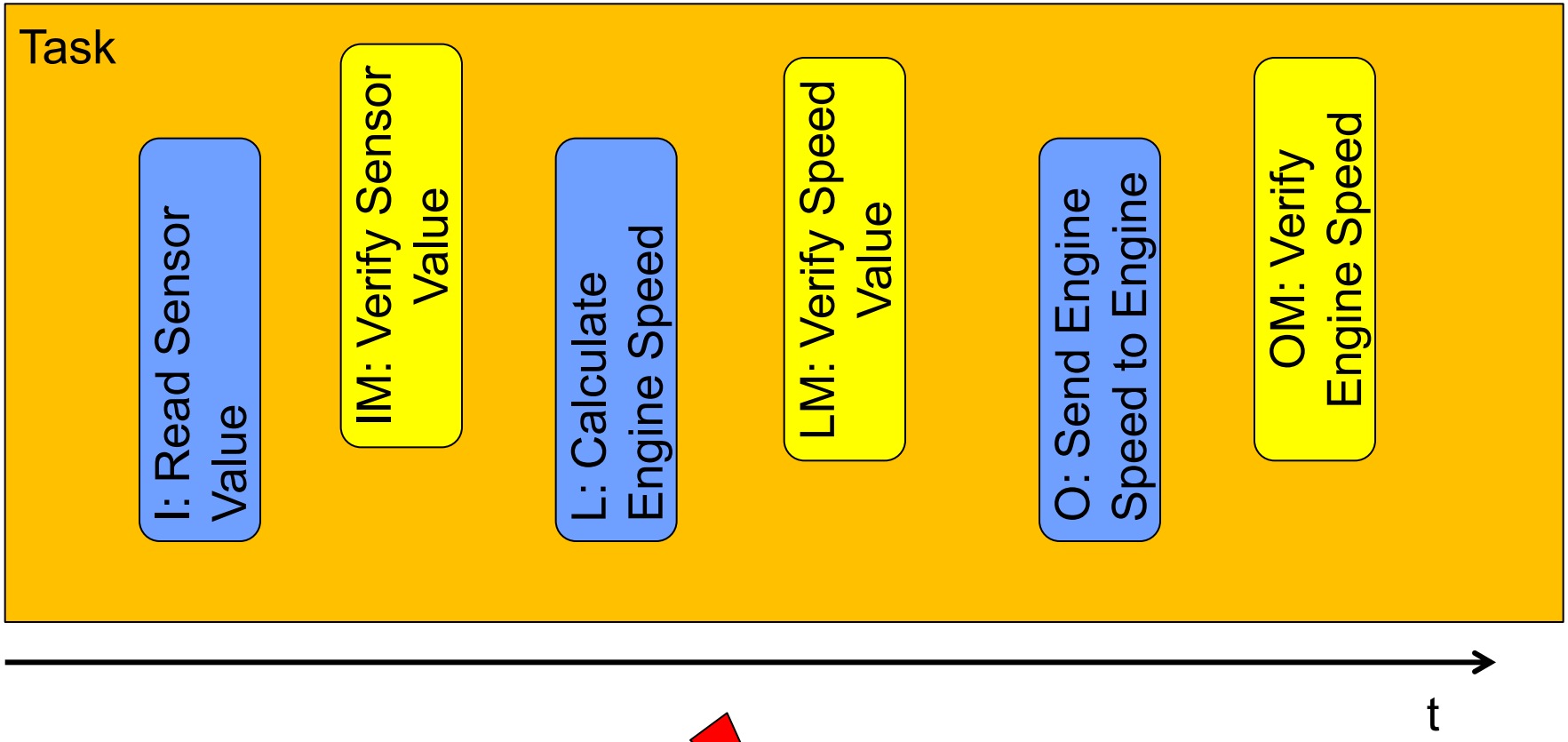
Runtime Environment – Signal Layer

Data transfer is abstracted by signals, which are organized in signal-layers.



Runtime Environment – Timing behaviour

Following the ILO cycle

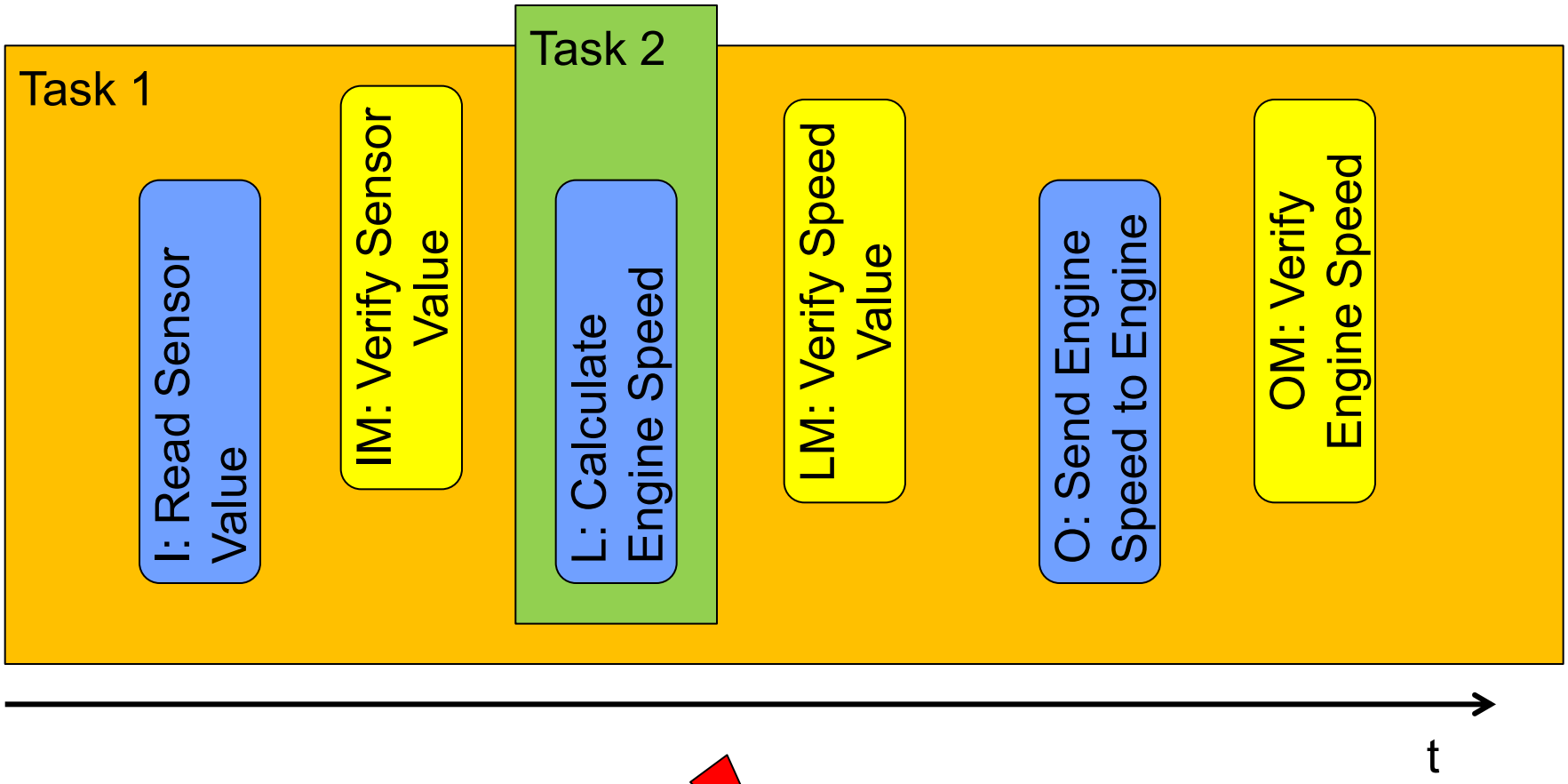


I: Input, IM: Input Monitor
L: Logic, LM Logic Monitor
O: Output, OM Output Monitor

Shared memory domain, control runnable may corrupt verified data

Runtime Environment – Timing behaviour

Separation of safety and non safety runnables

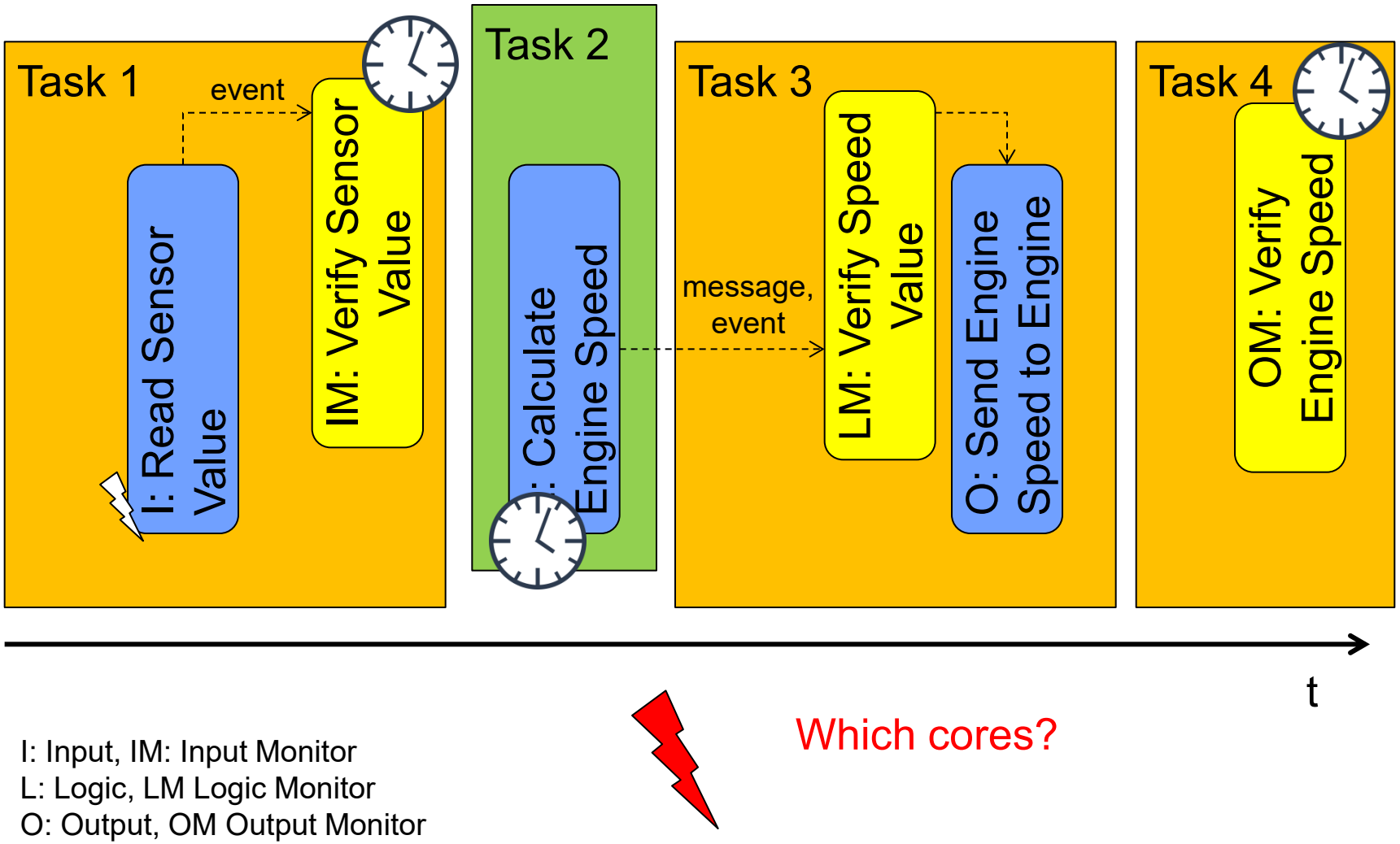


I: Input, IM: Input Monitor
L: Logic, LM Logic Monitor
O: Output, OM Output Monitor

Now the synchronization becomes a problem, how to deal with the age story?

Runtime Environment – Timing behaviour

Considering synchronization

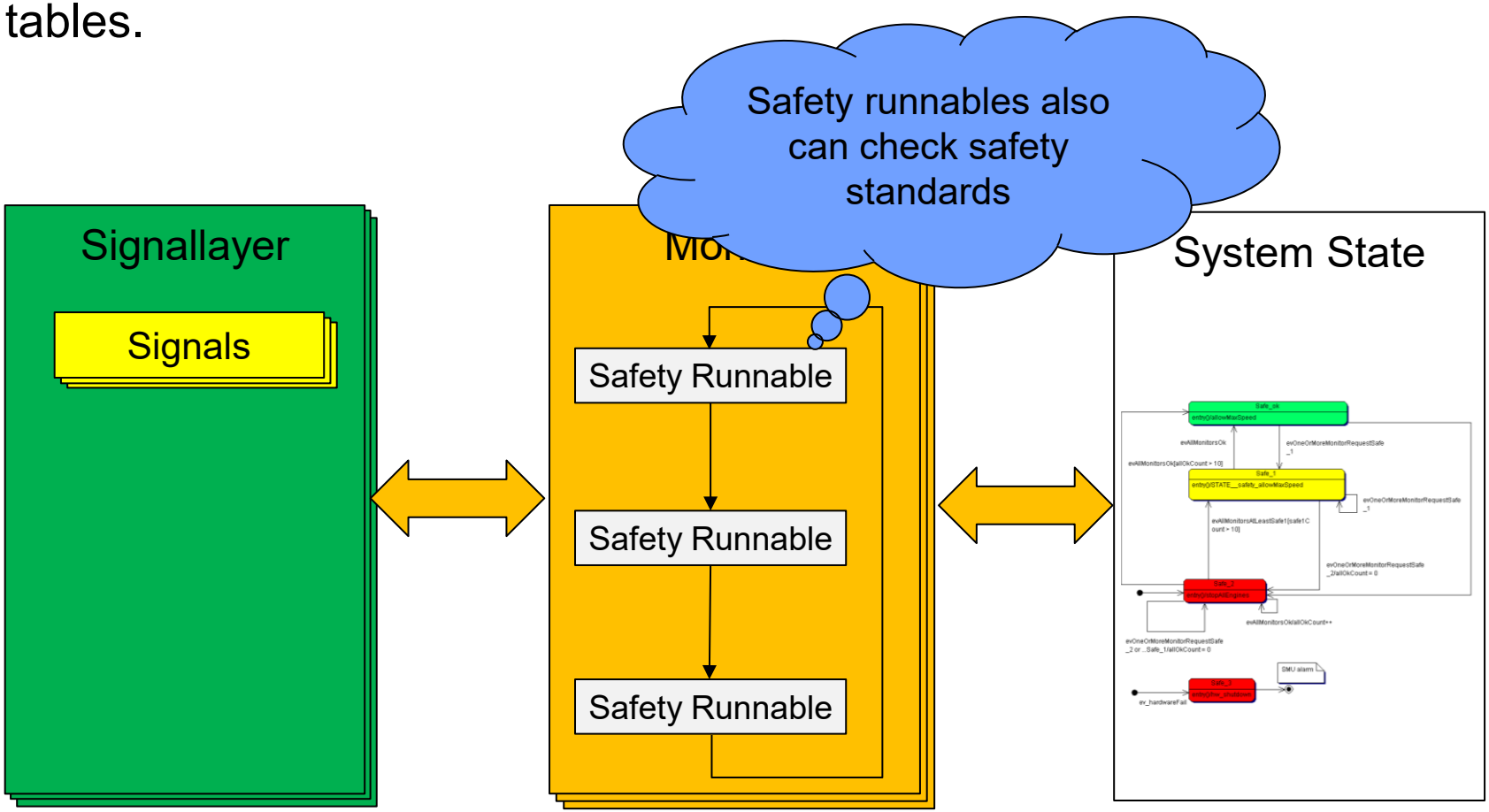


I: Input, IM: Input Monitor
L: Logic, LM Logic Monitor
O: Output, OM Output Monitor

Which cores?

Runtime Environment – Timing behaviour

(Safety) Runnables can be easily be incorporated by using runnable tables.



Pros and Cons

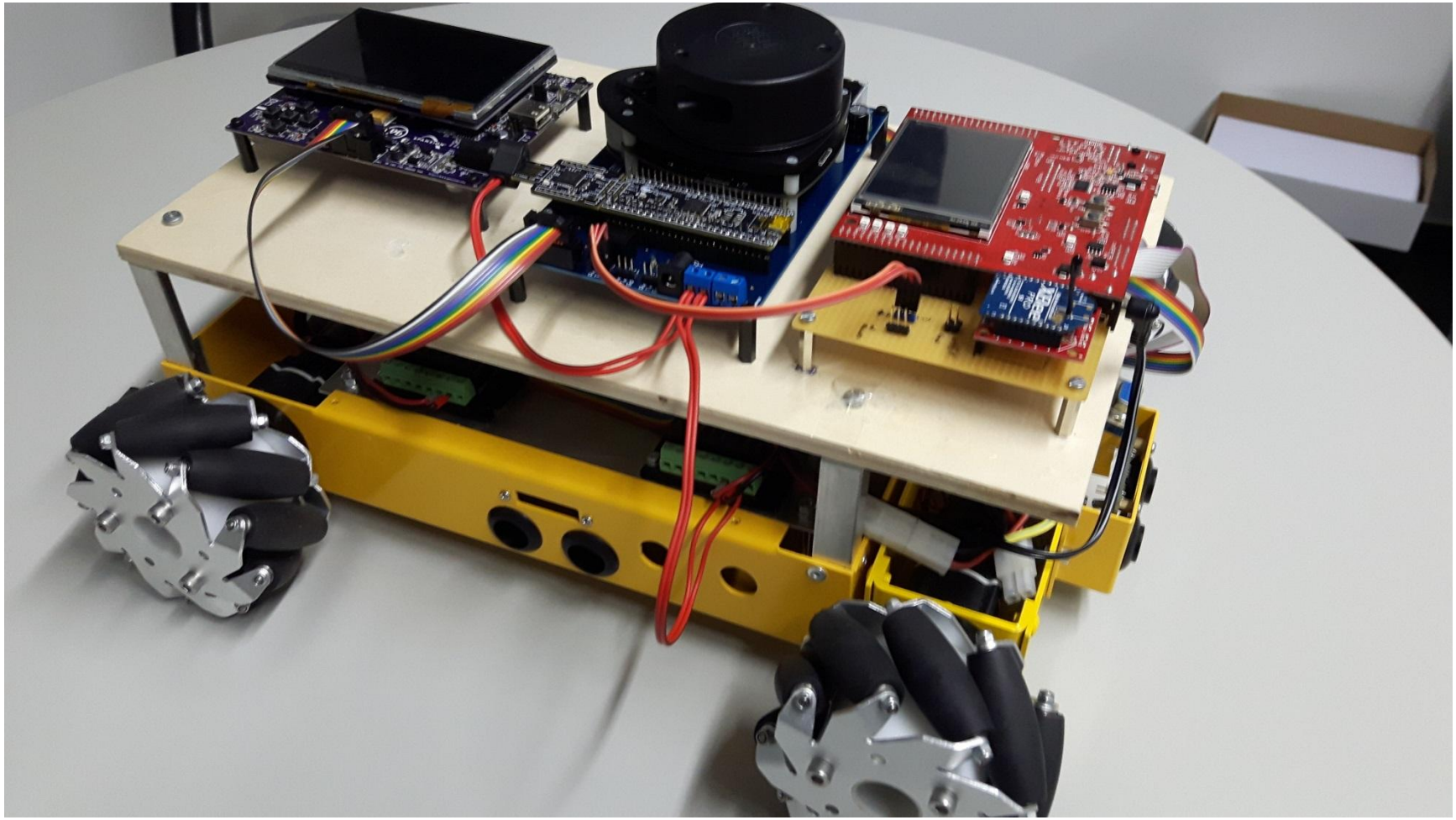
Benefits:

- Tests can be performed by stimulating signals
- 1 to 1 mapping to safety runnables
- (re)qualification is simplified by separating application and monitoring
- Possibility of module simulation
- Abstraction reduces system-complexity

Drawbacks:

- Error escalation across task boundaries
- Latencies accumulate if runnables are only triggered cyclically (events are supported)
- High demand of global RAM

Example Project - StudentCar



Conclusion

Lessons learned

- Multicore microcontrollers remain a big challenge
- A clear and safe system architecture is very important
- A runtime environment can decrease system complexity and ease (re)qualification

Outlook

- To reduce the complexity further, tools for model based system generation are under development.
- In research projects it was possible to integrate 5 sub-projects from 5 teams with 30 students with no multicore experience in 3 month.

Contact

M.Sc. Thomas Barth
Prof. Dr.-Ing. Peter Fromm

Hochschule Darmstadt - University of Applied Sciences
FB Elektrotechnik und Informationstechnik (EIT)
Birkenweg 8
64295 Darmstadt

Email: thomas.barth@h-da.de; peter.fromm@h-da.de
Web: www.eit.h-da.de