MASTER'S THESIS · WORK IN PROGRESS · HEICHIPS 2026 SUMMER SCHOOL, HEIDELBERG

A deterministic tile-based RISC-V multicore SoC

Simon Veser · Christian Burwick · Alexander Frank · veser@ims-chips.de
Institut für Mikroelektronik Stuttgart (IMS CHIPS) · in exchange with DLR

Universität Stuttgart · IMS CHIPS

The interference delay between cores is exactly zero, not merely bounded. The multicore WCET is the single-core WCET.

No shared memory, no cache, no coherence. Wait-free core ports plus fixed-length TDM slots mean one core's behaviour cannot change another core's timing at all. The usual decomposition **WCET_multi = WCET_single + interference** becomes exact instead of over-approximated.

0 interference delay · **O(N)** fabric cost, N = 2…32 · **125 MHz** placed & routed up to N = 32

1 Why this exists

Certification argues from a **proven** worst-case execution time, not from measurements. On a shared-memory multicore, one core's bus and cache traffic stretches another core's execution time by an amount nobody can bound tightly: **unbounded interference**.

So the highest civil-avionics criticality level (DAL A) has sat at quad-core for roughly 15 years, frequently with all but one core switched off. This work asks what the hardware has to look like if the bound is to be **trivial** instead of pessimistic.

2 Design rules, and one deviation

Design rule	This architecture	
Dedicated instruction memory per core	Private I-SPM per tile	met
Core-local data scratchpad, not shared RAM	Private D-SPM; no shared RAM exists	met
Stack in the core-local scratchpad	Stack in D-SPM	met
Each I/O channel owned by one core	UART, LEDs, buttons on the I/O tile	met
Bypass caches, disable branch prediction	Ibex has neither to begin with	n/a
No core reaches into another's scratchpad	The TDM NI writes the target's RX window	deviation

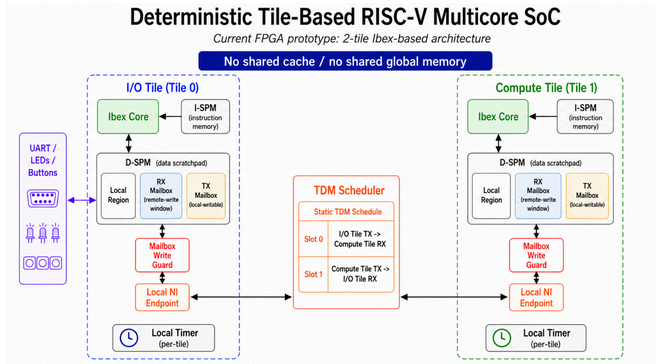
Rule set from S. Wegener, *Towards Multicore WCET Analysis*, WCET 2017.

3 The deviation is deliberate

The network interface writes into a foreign tile's RX window through port A of a dual-port BRAM. That is a real departure from the rule set, and it buys the whole communication mechanism.

- **Timing is untouched.** Port B, the core's port, stays wait-free: **gnt = req**, one cycle, independent of any NI access.
- **Collisions are defined, not undefined.** An O(1)-per-tile guard merges write/write per byte (NI wins) and forwards write/read. 73 FF per tile, at every N.

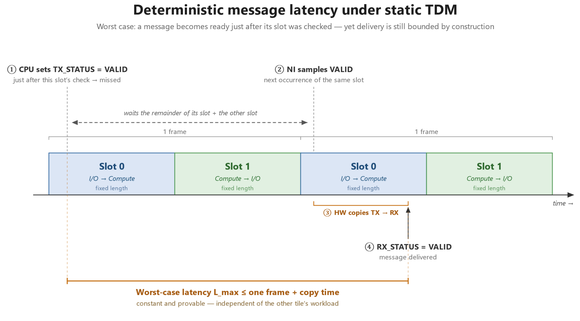
4 The architecture



- **Tile** = Ibex core (RV32IMC, 2-stage in-order, no cache, no branch prediction) + 8 KB private I-SPM + 8 KB private D-SPM + local timer.
- **The only cross-tile path** is the guarded NI port; everything else it could touch is blocked and counted in hardware.
- **Message passing, not shared memory.** Single-writer mailboxes with an ownership handshake, and no atomics anywhere; no shared mutable state exists.
- **The communication graph is configuration, not RTL.** Star, all-to-all or anything in between: the graph, the slots per frame and the TDM schedule are generated from a config file and tailored to the application.

5 Why the delay is exactly zero

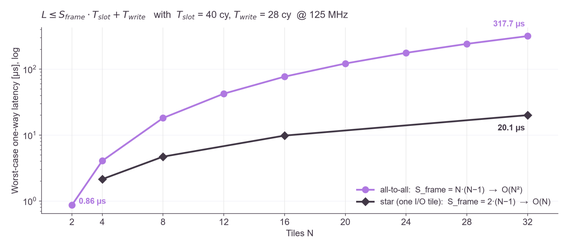
Every slot has **the same fixed length whether or not it carries a message**, and a core is never made to wait for the interconnect.



6 Latency is set by the schedule

$$L \le S_{frame} \cdot T_{slot} + T_{write}$$

slot length T_{slot} and copy time T_{write} are design constants, independent of traffic



Latency depends only on the schedule, never on what the other tiles are doing. S_{frame} comes from the communication graph: all-to-all is O(N²), a star around one I/O tile is O(N).

7 How software uses it

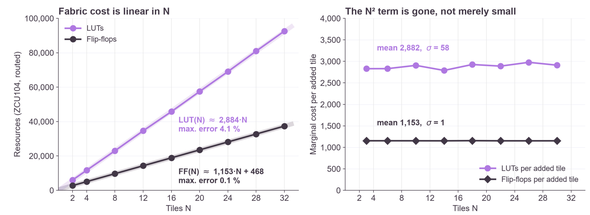
1. **Sender** writes its own TX mailbox, fences, sets **TX_STATUS = VALID**.
2. **Hardware** checks in that link's slot: sender ready, receiver **RX_STATUS == EMPTY**?
3. **Only then** copy TX → RX, the write guard confining the access.
4. **Ownership transfers**, and that handshake *is* the synchronisation primitive. No shared state to protect, no atomics in the ISA.

Backpressure is wait-free and visible. This is a central point: if the receiver has not emptied its mailbox, the transfer simply waits for a later slot. The sender core never stalls, the slot keeps its fixed length, and a hardware counter records every occurrence.

8 Where this sits

- **RCOM,** the predecessor at DLR: replicate the scratchpad once per core. Single-cycle reads, but the O(N²) replication scales badly in hardware.
- **T-CREST / Argo:** one network interface per node on a 2D-mesh NoC. The NIs alone are 74.6 % of a 4×4 NoC, on a custom VLIW ISA and toolchain.
- **This work:** one central NI, static TDM, private scratchpads. O(N) fabric on a stock RISC-V core, latency set by the schedule.

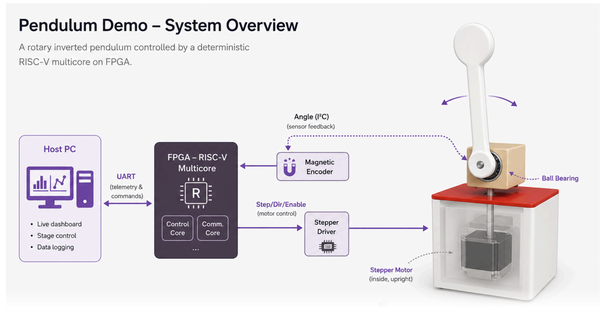
9 The hardware cost is linear in N



ZCU104 (xczu7ev), all-to-all, placed and routed, 13 points from N = 2 to 32. The N = 32 system is synthesised and running on the board; verification is in progress.

The entire deterministic communication layer, NI, write guard and mailboxes together, is a **flat 7.6 % of the LUTs at N = 32**. Determinism here is cheap, and it stays cheap.

10 Live demonstrator (in bring-up)



A rotary inverted pendulum, balancing while the other tiles are put under load. The same control task runs in three configurations:

1. **Single core:** the control loop shares one core with background load. It misses deadlines and the pendulum falls.
2. **Multicore, shared memory:** control gets its own core, but bus interference still stretches its timing. It falls.
3. **Multicore, isolated (this work):** full load on every other tile, and the pendulum stays balanced.

All three run the **identical control binary** on the same FPGA; only the platform integration changes.

Thesis in progress, come talk to me. Currently verifying the N = 32 board run and bringing up the pendulum. Ideas on WCET analysis and tooling for this architecture are especially welcome.

Simon Veser · veser@ims-chips.de
prototype on Xilinx ZCU104 (xczu7ev)