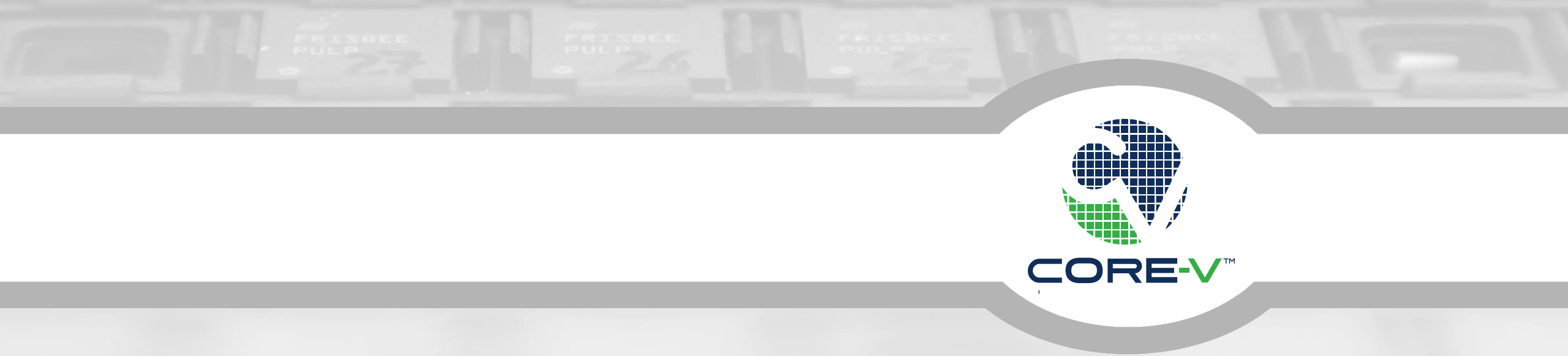


CORES TG – March 1 2021

Arjan Bink

Jérôme Quevremont

Davide Schiavone

Agenda

- CORE-V Family
- CV32A6 status
- Misc topics
  - Bus faults

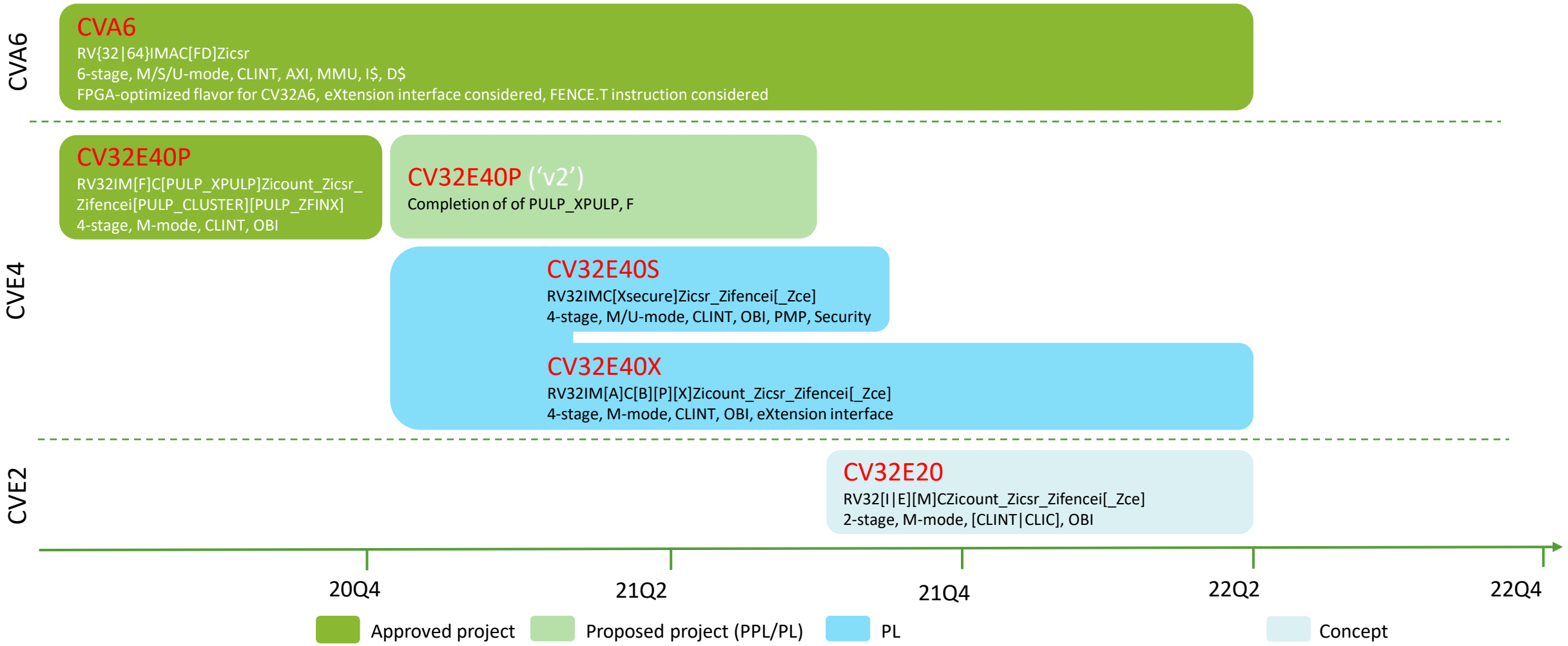
OpenHW Group CORE-V family

Davide Schiavone

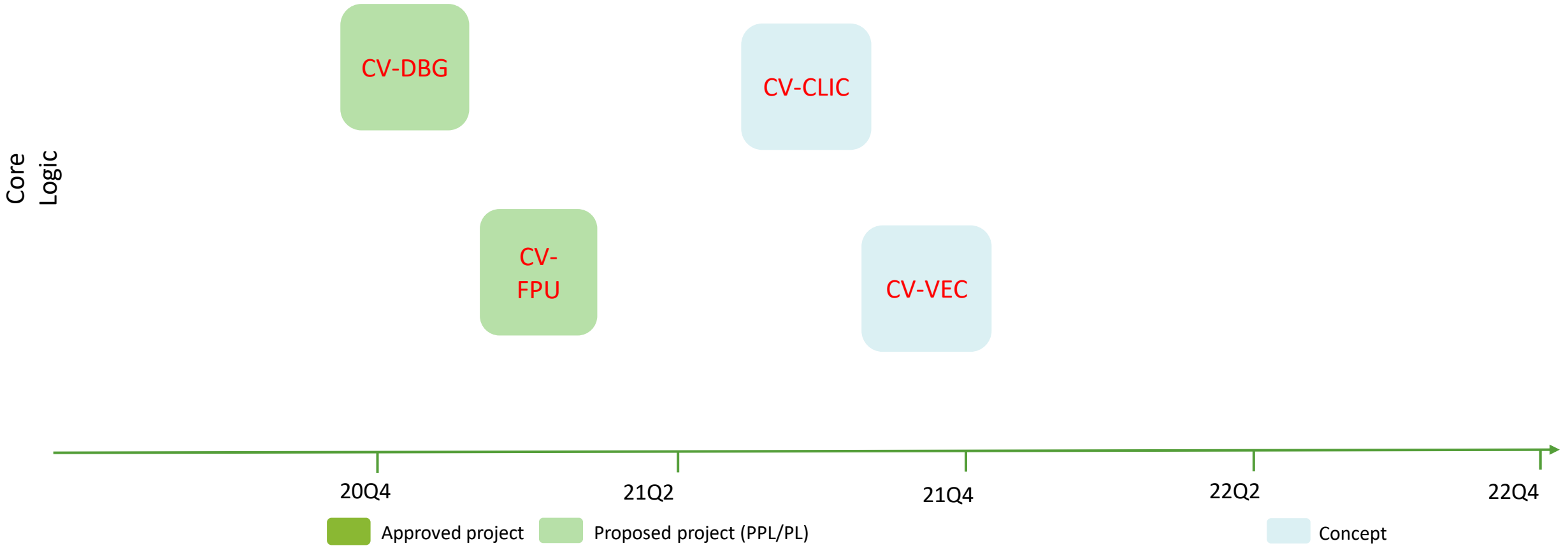
davide@openhwgroup.org

OPENHW GROUP™
PROVEN PROCESSOR IP

CORE-V RISC-V Cores

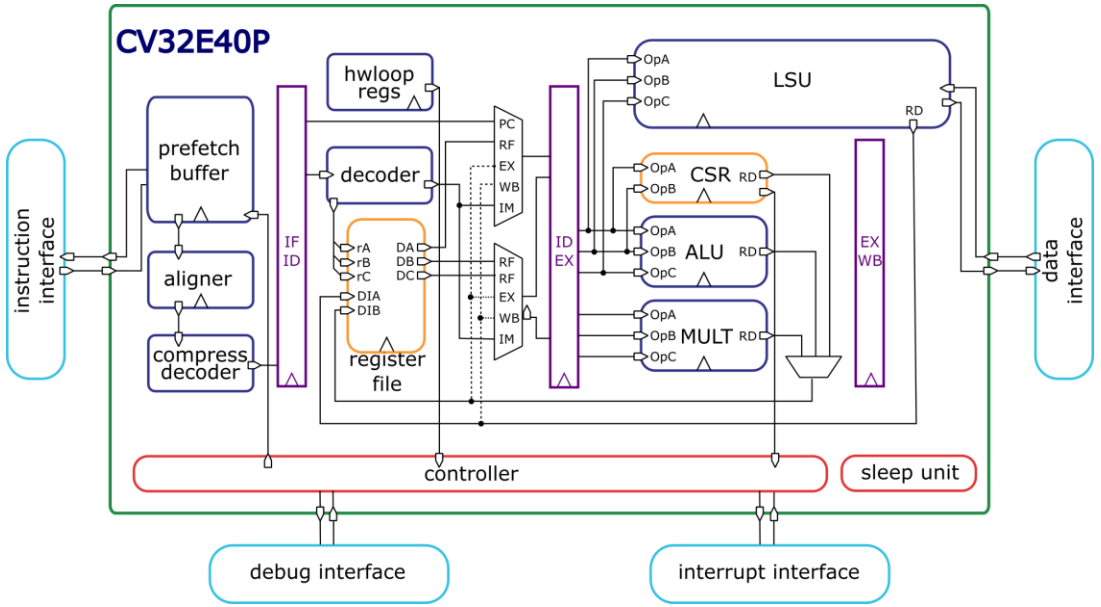


CORE-V Core Logic Blocks



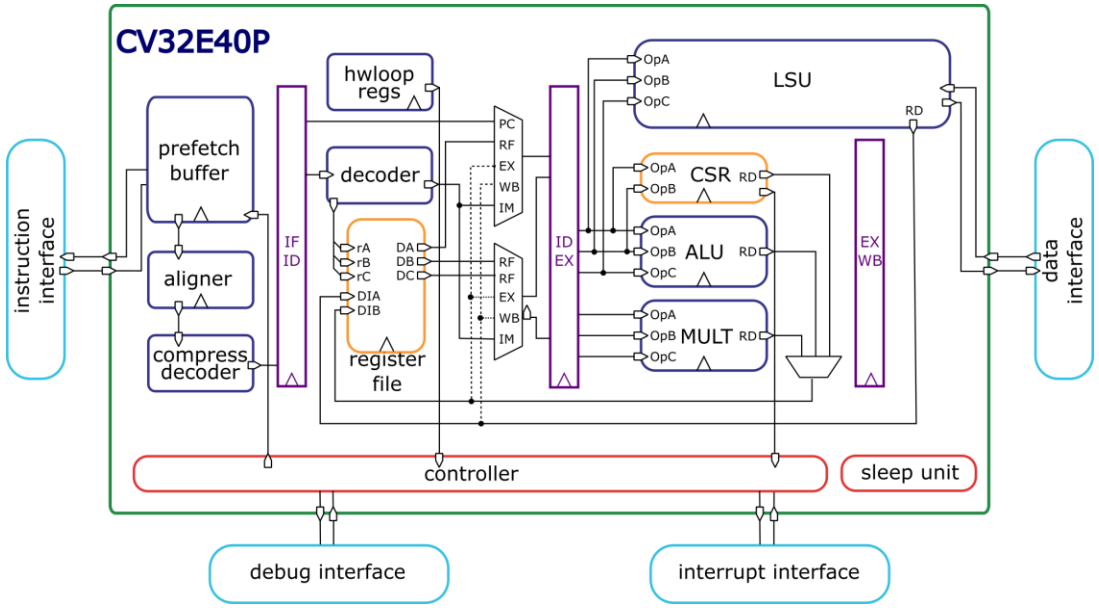
CV32E40P – PC

- ‘RTL Freeze’ achieved
  - RV32IMC extensions verified
  - Interrupts and Debug
- In production within CORE-V MCU
  - Microcontroller based on the ETH Pulpissimo
  - FPGA implementation based on Xilinx Genesys2
  - ASIC implementation based on Globalfoundries GF22FDX
    - MCU extended with an embedded FPGA fabric (ETH arnold chip)
- Activities on RVFI interface
  - Facilitating sim-based step&compare verification FSM and formal verification



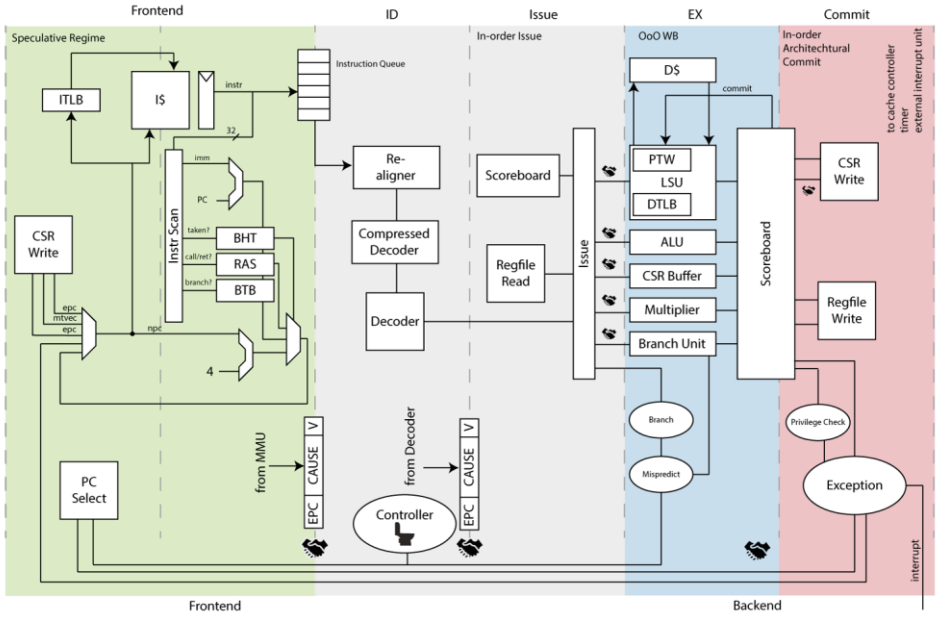
CV32E40Pv2 – PPL

- RV32Xpulp extensions
  - Verification and Reference Model
    - Extending the ISS with the Xpulp extensions
  - Moving the existing instructions to the RISC-V custom space
    - The current Xpulp extensions are in ***reserved*** space
    - Need to move them to be compliant with RISC-V
  - SW support with upstream GCC and LLVM compiler
    - OpenHW Group working on the compilers to support the Xpulp extensions
- RV32F extensions
  - Verification and documentation of the IP
    - See later
- CV-DBG
  - Moving, documenting, and verifying the external RISC-V Debug module



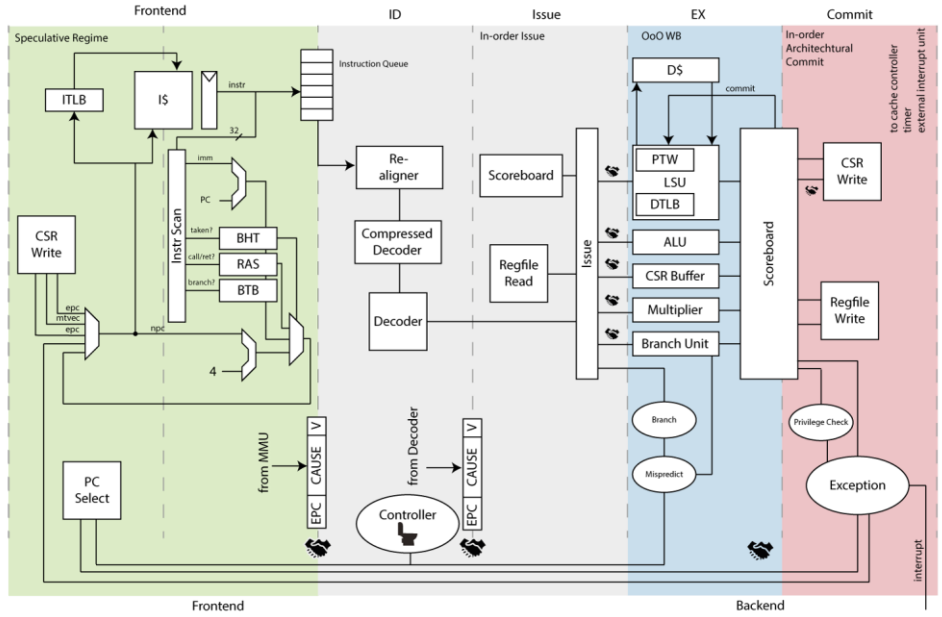
CV64A6 – (Q2 2022, PL stage)

- New documentation
  - reStructured text-based as for the other cores
- Verification
  - RV64GC extensions to verify (RV64IMAFDC)
  - sim-based Step&Compare, formal verification
  - RVFI interface
- FPGA optimizations
  - Target the Xilinx Genesys2, but not a Soft-Core!



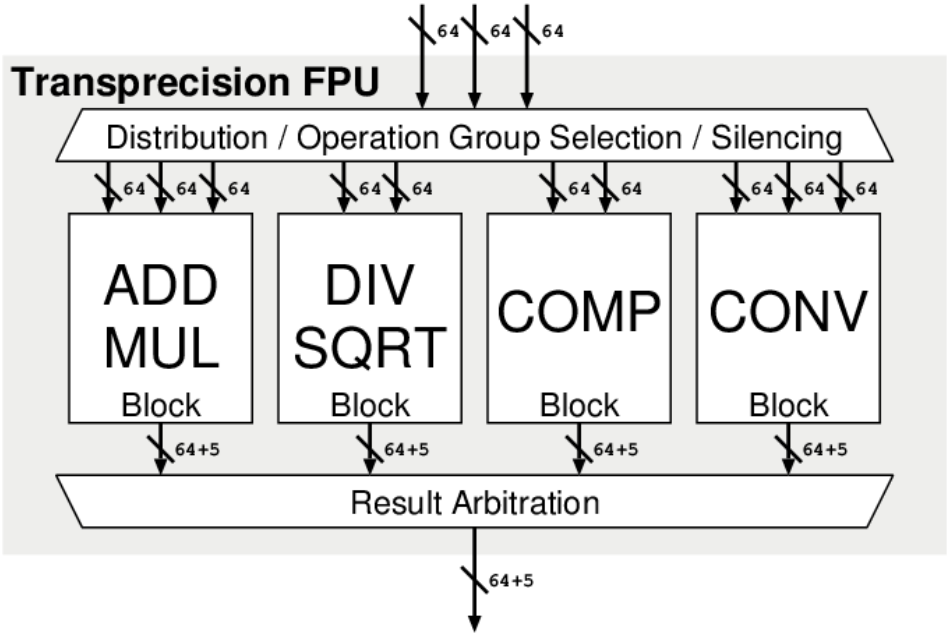
CV32A6 – (Q2 2022, PL stage)

- 32b version of cv64a6
  - Support for MMU
  - Support for RV32FD
  - Linux porting
- Verification
  - RV32GC extensions to verify (RV32IMAFDC)
  - sim-based Step&Compare, formal verification
  - RVFI interface
- PPA optimizations
  - For both FPGA and ASIC



CV-FPU – (Q2 2022, PL stage)

- IP that computes floating-point RISC-V RVF and RVD extensions
  - Included in different projects as
    - CV32E40Pv2, CVA6
- Starting point: ETH *fpnew*
  - Documentation
    - resTructured text documentation
- Verification
  - Verification stand-alone or as part of the cores



CV32E40S – (Q4 2021, PL stage)

- Secure core
  - Support for User-Mode and enhanced PMP
  - Anti-tampering features
    - Protection against glitch attacks, Control flow integrity, Autonomous response mechanisms
  - Reduction of side-channel attacks
  - Zce extension (if ratified)
  - PPA optimizations w.r.t. CV32E40P
  - Support for bus error
  - Support for PMA

- Verification
  - ISS model required for step&compare, formal verification
  - RVFI interface

CV32E40X – (Q4 2022, PL stage)

- Compute intensive core with CV-X-IF interface
  - RV32IM[A]C core with CV-X-IF interface to offload custom extensions
  - RV32B and RV32P extensions (if ratified)
  - Zce extension (if ratified)
  - PPA optimizations w.r.t. CV32E40P
  - Support for bus error
  - Support for PMA

- Verification
  - ISS model required for step&compare, formal verification
  - RVFI interface

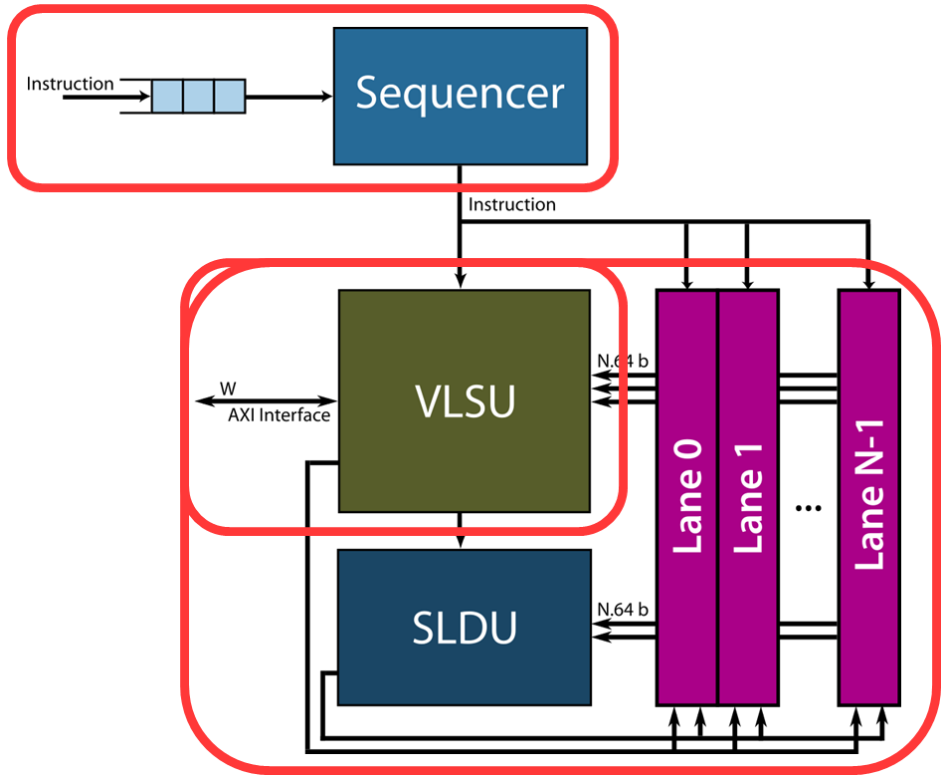
CV32E20 – (Q2 2022)

- Low area core
  - RV32{I,E}[M]C only extensions
  - Optimized power and area for control-oriented applications
  - Starting point lowRISC Ibex (which started from ETH zero-riscy)
    - Clean-up parameters
    - Aligning IP interface with CV32E40* cores

- Verification
  - RVFI interface already available

CV-VEC

- IP that computes vectorial instructions based on RISC-V RVV extensions
  - compatible with
    - CV64A6
- Starting point: ETH ara
  - Documentation
    - resTructured text documentation
- Verification
  - Verification stand-alone or as part of the cores
- Join research activity between OpenHW Group, ETH, and PolyMTL



CV-X-IF Specification

- Bus spec to allow RISC-V cores to offload RV extensions
  - CV32E40X will leverage this interface the most
  - CV32E40P will use it for RV32F extensions
  - CVA6 may use it for RVV extensions
- Verification
  - Formal verification needed to test the spec
- Allows the cores to be agnostic about the custom extension definition

CVA6 status

Jérôme Quevremont

CVA6

- PL on 2020-01-25
  - Start specification phase
  - Clarified documentation architecture
  - Need to attract more participants, especially on verification
  - Project meeting every 2 weeks (next: 2020-03-05 2pm CET, see https://calendar.google.com/calendar/u/0/embed?src=meetings@openhwgroup.org)
- Topics currently addressed:
  - Project management: hybrid approach (waterfall then sprints)
  - Specification tooling (editor, need requirement tracing?)
  - FPGA optimization
    - Benchmarks show room for improvement (frequency, resource utilization)
    - PhD starting at ETH Zürich
  - On-going design
    - Sv32 MMU PR submitted, started porting CV32A6 to Genesys2, goal to support Linux
    - CV32A6 floating point support on-going
  - Two verification environments supported:
    - "Reference" environment with Imperas ISS, UVM step and compare…, targeting 100% coverage
    - "Alternative" "sustainable" open-source environment with Spike ISS (and Verilator) support
    - How to combine them?

Misc topics – Bus Faults

Arjan Bink

Bus errors on RISC-V

- Behavior for bus errors is not specified for RISC-V
- Community uses diverse methods, e.g.
  - SiFive uses a ‘bus error unit’ that converts bus errors into regular interrupts
  - Ibex implements precise bus errors and causes exceptions using RISC-V defined mcause exception codes (i.e. instruction access fault (exception code 1), load access fault (exception code 5), store/AMO access fault (exception code 7) (no way to differentiate between PMP errors and bus faults)
  - SweRV-EL2 maps imprecise bus errors onto custom NMIs (and they also have precise bus errors).

Bus error proposal (custom exception codes)

Fault	mcause	mepc	mtval	Write back	Priority	Comment
Instruction bus fault (precise) *	interrupt = 0, exception code 24	PC of faulting instruction	Faulting address	-	Same priority as instruction access fault	
Load bus fault (precise)	interrupt = 0, exception code 25	PC of faulting (load) instruction	Faulting address (used on data bus)	No	Same priority as load access fault	
Store bus fault (precise)	interrupt = 0, exception code 26	PC of faulting (store) instruction	Faulting address (used on data bus)	-	Same priority as store access fault	
Load bus fault (imprecise) *	interrupt = 1, exception code 64	PC of next instruction to be executed at the time the NMI is taken	- (faulting address can be made available in platform defined peripheral)	Yes (invalid data can enter register file and be used; not using the invalid data seems equally bad)	NMI	NMI to nmi_vec_i. Fatal, not resumable.
Store bus fault (imprecise) *	interrupt = 1, exception code 65	PC of next instruction to be executed at the time the NMI is taken	- (faulting address can be made available in platform defined peripheral)	-	NMI	NMI to nmi_vec_i. Fatal, not resumable.

* CV32E40X / CV32E40S will use precise instruction bus fault, imprecise load bus fault, imprecise store bus fault