CORES TG – January 28 2021

Arjan Bink

Jérôme Quevremont

Davide Schiavone

Agenda

- CV32E40P RTL Freeze
- Silicon Labs view of Core priorities
  - CV32E40S
  - CV32E40X
  - CV32E20
  - View on roadmap
- CV32A6

Updates on CV32E40P

- ‘RTL Freeze’ achieved for CV32E40P
  - RV32IMC extensions verified
  - Interrupts and Debug
    - 40 RTL bugs found! Congrats to everyone

- We want to move the Documentation to the Core repository

- We want to build a CI flow based on GitHub to guarantee:
  - logical equivalence (it does guarantee backwards functionality)
  - This will prevent changes to the core unless for BUGs or another set of PARAMETERS

CV32E40P – next step

- XPULP instructions on verification
  - RV32Xpulp extension to be verified
  - First we need to relocate them to a custom instruction space
    - The current encoding is not RISC-V compliant, this difficult to be pushed to mainstream GCC and LLVM
  - Random Instruction Generator needs to be extended for Xpulp

- RISC-V Debug Module to moved to OpenHW Group and verified
  - Standalone or linked to the core? TBD

- Imperas RVVI: implementing verification interface to the core

CV32E40S/CV32E40X/CV32E20

Silicon Labs view of Core priorities

Arjan Bink (Silicon Laboratories)

CV32E40S/CV32E40X/CV32E20

- CV32E40S
  - 4-stage RISC-V core aimed at security
  - Key features: ePMP, Machine + User mode, anti-tampering features
- CV32E40X
  - 4-stage RISC-V core aimed at compute intensive applications
  - Key features: P, B, F, general purpose accelerator interface
- CV32E20
  - 2-stage RISC-V core aimed at control applications
  - Key features: Low cost, low interrupt latency, Zce

CV32E40S – Secure core

Arjan Bink (Silicon Laboratories)

CV32E40S – Secure core (1/2)

- CV32E40S is a 4-stage secure RISC-V core supporting the RV32IMCXsecureZce_Zicsr_Zifencei instruction set
- Key application areas
  - Security
- High level security features
  - RISC-V standard features (ePMP, U)
  - Anti-tampering features
    - Protection against glitch attacks
    - Control flow integrity
    - Autonomous (hardware-based, low latency) response mechanisms
  - Reduction of side channel leakage
- Compared to Ibex
  - Additional anti-tamper features
  - Higher performance
  - Comparable area for same feature set
  - Based on CV32E40P

CV32E40S – Secure core (2/2)

- CV32E40S key features
  - RV32IMCXsecureZicsr_Zifencei_Zce
  - 4-stage pipeline
  - M/U-mode
  - Enhanced PMP (ePMP)
  - CLINT
  - OBI
- Security features (Xsecure)
  - Security alert outputs
  - Data independent timing
  - Dummy instruction insertion
  - Register file ECC
  - Hardened PC
  - Hardened CSRs
  - Control flow hardening
  - Functional unit hardening
  - Bus interface hardening
  - Reduction of profiling infrastructure
  - Etc.
- Code size reduction extension (Zce)
- Bound RVFI interface
  - Formal verification
  - Enable standard ISS lock step compare
- Bus error support
- Extended Debug Trigger (0.14)
  - Multiple breakpoints,
  - Data interface related breakpoints
  - Support for etrigger
  - Optional exception instead of halt for triggers
- Simplified pipeline and controller
  - Debug FSM simplification
  - Complete removal of non-security related custom features (PULP, APU)
- Performance, area and power optimizations
  - Limit prefetch depth
  - Register file optimization
    - Remove 2nd registerfile write port
    - Remove 3rd read port
  - ALU clean up
- Other improvements
  - fence.i interface
  - mtval implementation
- Dependencies and scope
  - Zce, ePMP, 0.14 of Debug pending on timely ratification

CV32E40X - eXtendable compute core

Arjan Bink (Silicon Laboratories)

CV32E40X – Extendable compute core (1/2)

- CV32E40X is a 4-stage RISC-V core aimed at compute intensive applications supporting the RV32IM[A][F]C[B][P]X Zce_Zicount_Zicsr_Zifencei[_Zfinx] instruction set
- Key application areas
  - Medium performance compute intensive applications
  - Accelerator interfacing
- High level compute features
  - P, B, F extensions
  - Accelerator interface (X)

CV32E40X – Extendable compute core (2/2)

- CV32E40X key features
  - RV32IM[A][F]C[B][P]XZce_Zicount_Zicsr_Zifencei[_Zfinx]
  - 4-stage pipeline
  - M-mode
  - CLINT
  - OBI
- No custom instructions
- General purpose eXtension itf
  - Generic (applicable to ALU type instructions, loads/stores)
  - Tightly integrated (e.g. providing read and write access to register file, bypass signals, stall signals, etc.)
  - Low latency (instructions same latency as can be expected when adding the custom instruction directly into the core)
- Code size reduction extension (Zce)
- Bound RVFI interface
  - Formal verification
  - Enable standard ISS lock step compare
- Extended Debug Trigger (0.14)
  - Multiple breakpoints,
  - Data interface related breakpoints
  - Support for etrigger
  - Optional exception instead of halt for triggers
- Bus error support
- Simplified pipeline and controller
  - Debug FSM simplification
  - Removal of custom (PULP) features
- Performance, area and power optimizations
  - Limit prefetch depth
  - Register file optimization (if P, B excluded)
    - Remove 2nd registerfile write port
    - Remove 3rd read port
  - Faster divide
  - ALU clean up
- Lower worst case interrupt latency
  - Interruptable div/divu/rem/remu
  - Improve by ~32 cycles
- Other improvements
  - fence.i interface
  - mtval implementation
- Based on CV32E40S
- Dependencies and scope
  - Zfinx, Zce, P, B, 0.14 of Debug pending on timely ratification

Benefits of X interface

- Does not claim opcodes for non-used functionality
- Does not spend area on non-used functionality
- Enables adding custom extensions without releasing the extensions themselves
- Limits verification and documentation effort to the X interface itself as opposed to the custom instructions
  - Instead burden is put on the user interested in such custom instructions
- Enables a business/support model for tool vendors (e.g. compiler, ISS, assertions, etc.) to provide support for added custom extensions

CV32E20 – Low cost control core

Arjan Bink (Silicon Laboratories)

CV32E20 – Low cost control core (1/2)

- CV32E20 is a 2-stage RISC-V core aimed at low cost control applications supporting the RV32[I|E][M]CZce_Zicsr_Zifencei instruction set

- Key application areas
  - Low cost control applications
  - Interrupt intensive applications

- High level features
  - Low cost: E, C, Zce extensions
  - Low latency interrupts: CLIC

CV32E20 – Low cost control core (2/2)

- CV32E20 key features
  - RV32[I|E][M]CZce_Zicsr_Zifencei
  - 2-stage pipeline
  - M-mode
  - CLINT or CLIC
  - OBI
- Based on Ibex (former zero-riscy)
  - Completely remove unused features (PMP, User mode, NMI, etc.)
- Focus
  - Low cost
  - Fast interrupts
- No custom instructions
- Standard RISC-V debug (0.14)
- Support for bus errors
- Single cycle branch penalty
- Code size reduction extension (Zce)
- Bound RVFI interface
  - Formal verification
  - Enable standard ISS lock step compare
- Pin compatible with CV32E40P, except
  - Removal of APU interface
  - Addition of bus error pins
- Dependencies and scope
  - E, Zce, CLIC pending on timely ratification

Core-V roadmap – Silabs view (not approved)

CORE-V

Misc

- RISCV-DBG
- FP NEW
- CLIC-EXT

CVA6

CVA6
RV{32|64}IMAC[FD]Zicsr
6-stage, M/S/U-mode, CLINT, AXI, MMU, I$, D$
FPGA-optimized flavor for CV32A6, eXtension interface considered, FENCE.T instruction considered

CVE4

CV32E40P
RV32IM[F]C[PULP_XPULP]Zicount_Zicsr_
Zifencei[PULP_CLUSTER][PULP_ZFINX]
4-stage, M-mode, CLINT, OBI

CV32E40P (‘v2’)
Completion of of PULP_XPULP, F

CV32E40S
RV32IMC[Xsecure]Zicsr_Zifencei[_Zce]
4-stage, M/U-mode, CLINT, OBI, PMP, Security

CV32E40X
RV32IM[A][F]C[B][P][X]Zicount_Zicsr_Zifencei[_Zce]
4-stage, M-mode, CLINT, OBI, eXtension interface

CVE2

CV32E20
RV32[I|E][M]CZicount_Zicsr_Zifencei[_Zce]
2-stage, M-mode, [CLINT|CLIC], OBI

20Q4 | 21Q2 | 21Q4 | 22Q2 | 22Q4

OPENHW GROUP™ PROVEN PROCESSOR IP

Approved project | Proposed project (PPL/PL) | Ongoing (will be donated to OpenHW soon) | Concept

Interested in participating on E40S/E40X/E20?

- Work on spec/doc/design verification is in progress within Silabs
  - Will convert into projects in OpenHW a.s.a.p.
  - Projects are being run in an agile manner (E40X/E40S first)
  - Focus on stability
    - Continuous integration
      - Verification
      - Synthesis (area, power, timing)
      - Formal
    - Limit (or completely remove) uncontrolled dependencies
      - Avoid core-v-verif interference to/from CVA6 and CV32E40P

- Focus on shared base architecture for CV32E40S and CV32E40X
  - Based on CV32E40P

- Interested in contributing?
  - Please contact me at arjan.bink@silabs.com

CVA6
Project Launch (PL gate)

JQ, Thales

Technical WG, 2020-01-25

CVA6
Project Launch (PL gate)

Jérôme, Thales

Technical WG, 2020-01-25

Introduction

- Preliminary Project Launch (PPL gate) approved on 2020-09-28
  - core-v-docs/CVA6 preliminary project proposal.md at master · openhwgroup/core-v-docs · GitHub

- Time to have our Project Launch (PL):
  - Resources joining Thales and OpenHW staff
  - More OpenHW bandwidth after CV32E40P RTL freeze

- This PL presentation:
  - Additional details that complete the PPL document
  - Switched to slides for more visual content

Main evolutions (vs. PPL)

- Spun-off projects:
  - LLVM support
    - Can be run in a standalone fashion as CVA6 sticks to RISC-V ISA
  - Core-v-verif
    - Common environment for future CORE-V cores
- Features
  - Considering the addition of a coprocessor interface
    - To be standardized among CORE-V cores
    - FPU could connect to this interface but would presumably be kept internal (and optional) for performance
  - Added RVFI interface

Documentation

- Clarified document structure
  - Document names can evolve
  - Main documents below
- Core:
  - Specification
    - Identifies features agreed upon
    - "What" defined as requirements with identifiers
    - Main input for design and verification work
    - Some sections can be short (references to RISC-V ISA, AXI specs…)
    - Best example: Open Bus Interface
  - Users' guide
    - Includes the specification
    - For CVA6 integrators and users: HW, SW, ASIC, FPGA… viewpoints
    - Need it soon enough
  - Design document
    - Explains the "How": design choices…
    - Not prescriptive, written during or after the design. Useful for next projects.
    - Best example: ARIANE pipeline

Input to

Input to

- Verification
  - Verification Environment Specification
    - User-manual for the verification environment (testbenches, testcases, verification components, etc.)
    - Description of the testbench structure and theory of operation.
    - Best examples: lowRISC IBEX Documentation and the core-v-verif Verification Strategy.
  - Design Verification Plan
    - DVplan, Verification Plan, Vplan: same meaning
    - Feature-by-feature listing of the Device Under Test
      - and a description of how it will be verified
      - and how we know when it is verified (coverage).
    - Examples in: https://github.com/openhwgroup/core-v-docs/tree/master/verif/CV32E40P/SimulationVerificationPlan

Verification

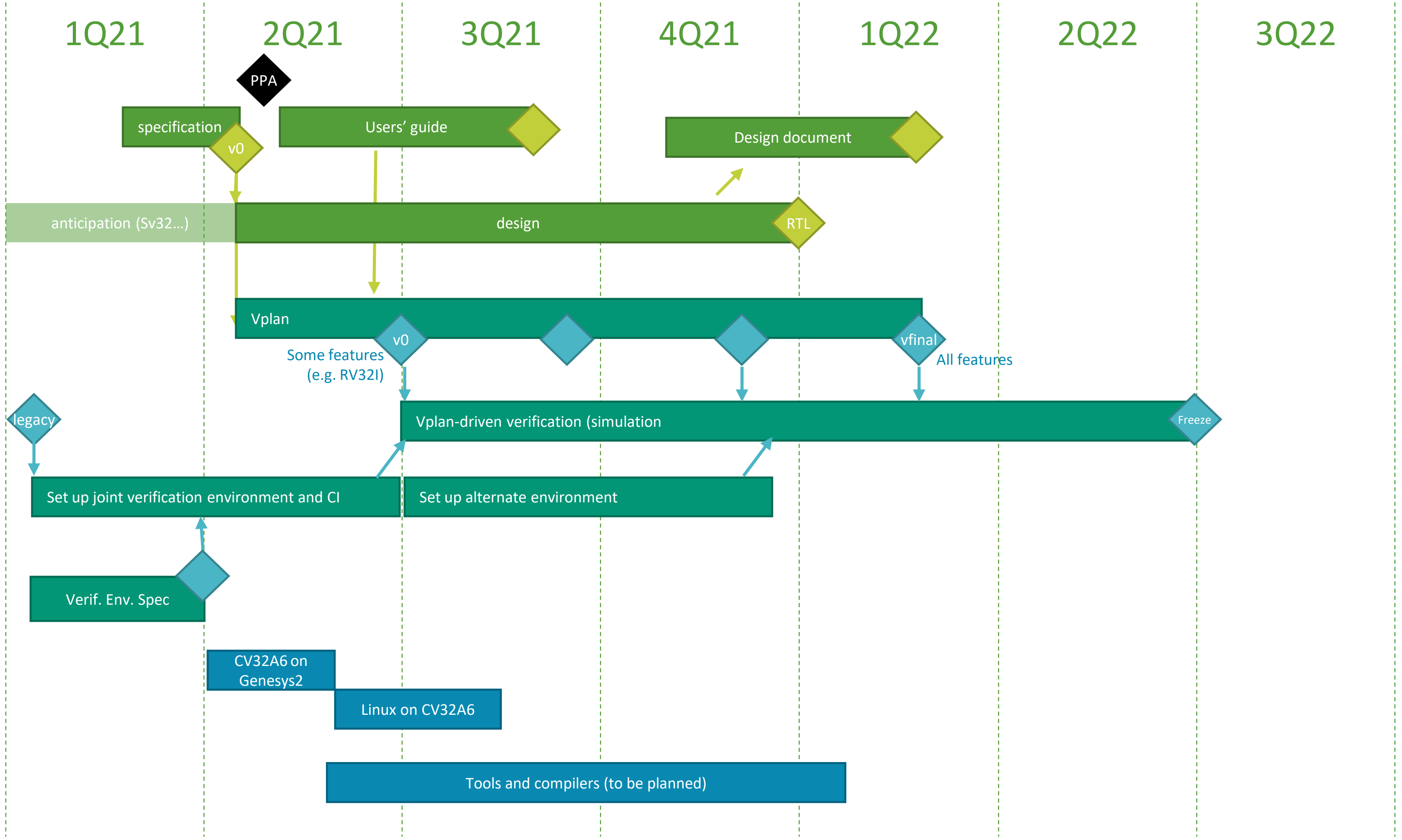
- Two verification environments supported:
  - "Reference" environment with Imperas ISS, UVM step and compare…
  - "Alternative" "sustainable" open-source environment with Spike ISS and Verilator support
- Current gap:
  - CV64A6 verified in Travis environment (ETHZ legacy)
  - CV32A6 verified in Thales-originated bench (https://github.com/openhwgroup/core-v-verif/tree/master/cva6
  - Delays for commits
  - Commit to "legacy" CV64A6 sometimes break CV32A6
  - Some 2020 commits have introduced significant CoreMark decrease
- Priority: set up a joint testbench/CI/commit process
  - Mike has already started ☺

Resources and tasks

- Thales TRT in 2021
  - Jérome: CVA6 TPL, coordinate specification
  - Sébastien:
    - Add Sv32 support (CV32A6), port to Genesys2, support Linux on CV32A6
    - FPGA frequency/resources optimizations
  - Emeric (part time): make WT cache more robust, add a few features
- Thales India: recent hires of senior engineers
  - Pranay: add FPU support to CV32A6, investigate modularity, FPU optimization for FPGA
  - Ranjan (expected e/o March): verification
  - Anjali: toolchain, Linux in cooperation with Sébastien
- Thales INVIA:
  - Fix CV32A6 bugs
  - Coprocessor interface
  - CVA6 LLVM
  - Help transition from CV32A6 testbench
- OpenHW staff
  - Mike:
    - Coordinate verification
    - Verification environment specification
    - Started working on testbench
  - Gianmarco: MEng and PhD student
    - Focus on design
    - First steps could be on documentation
    - Contribution yet to decide
      - Maybe performance/resource optimization for ASIC&FPGA
  - Florian:
    - ?
- Other members?

More verification resources wanted!

Master planning



Actual duration will depend on the available resources.

Focus on some critical and short term tasks. Detailed planning with all tasks (as in PPL) deferred to PPA gate.

Coordination until the PPA gate

- CVA6 meetings every 2 weeks
  - Cross-TG: Specification, verification
  - Participants:
    - CVA6 contributors, including involved OpenHW staff
    - OpenHW chairs and members welcome

- Dedicated technical meetings when needed

- Reporting to task groups and TWG (short)

Thank you!