2023 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION JAPAN

# How to overcome the hurdle of customizing RISC-V with formal

Pascal Gouédo, Dolphin Design
Seiya Nakagawa, Siemens EDA
Nicolae Tusinschi, Siemens EDA
Salaheddin Hetalani, Siemens EDA

# QOS* Processor introduction

Questa OneSpin Solutions

## Challenges of processor verification

ISA, architecture and specification verification deliver cores with integrity

**Complex architecture**

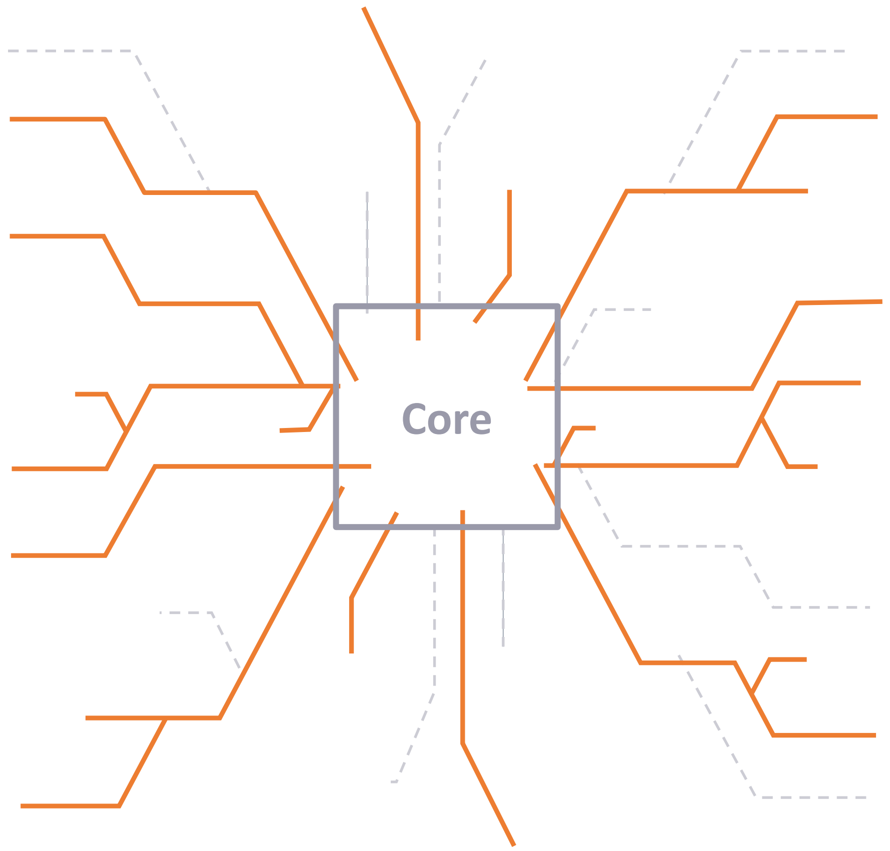
- (Custom) extensions
- Exceptions/ Interrupts

**Very complex µArchitecture**

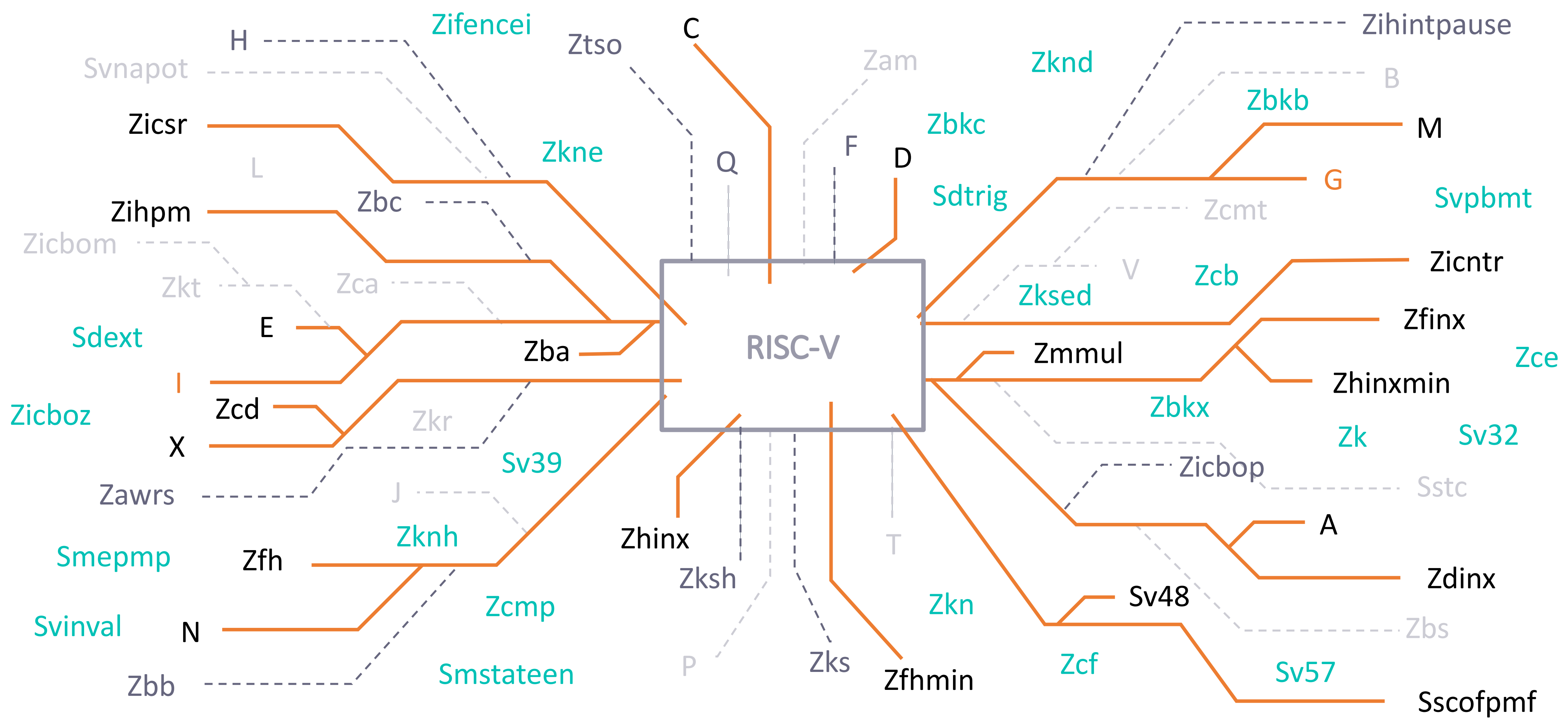
- Continuous PPA optimizations
- Pipelined implementation

**Verification – high effort task**

- Writing functional coverage model
- Simulation cannot hit all pipeline corner-cases
- Slow debug process
- Functional and structural coverage closure
- Customization introduces bugs in existing functionality



# Ultimate freedom of RISC-V



## QOS Processor

Accelerate, automate and increase quality of processor verification

### Verification Speed-up

- No writing of testbench
- Find RTL issues earlier than in UVM flow
- Accelerate coverage closure
- Optimized formal engines
- Pinpoint bugs systematically
- Quick fix check

### High degree of automation

- No writing of functional coverage model
- Designed for custom extensions
- µArchitecture extraction
- Assertion generation
- Initial value abstraction
- Disassembler annotation
- Trace analysis

### Exhaustive & complete verification

- No undocumented RTL
- 100% functional coverage
- Unbounded proofs
- Finds bugs other technologies can't
- Essential for state-of-the art processor DV
- ISA & privileged ISA compliance

## Siemens EDA's Industry proven solutions

Codasip — How the Right **Mindset** Increases **Quality** in RISC-V Verification — 2022 DVCon Europe, Munich, Germany, December 6 - 7, 2022

Edaptive Computing, Inc. — **Complete** Formal Verification of **RISC-V Processor IPs** for **Trojan-Free Trusted ICs** — GOMACTech 2019

Bosch — **Complete** Formal Verification of a Family of **Automotive DSPs** — 2016 DVCon Europe

Renesas — Formal Verification Applied to the Renesas **MCU Design Platform** — 51 Design Automation Conference

Infineon — **Complete** Formal Verification of **TriCore2** and Other Processors — *The content of this article was presented at DVCon 2007 and is posted with DVCon's permission.*

**~20 years**

of cutting-edge formal processor verification solutions

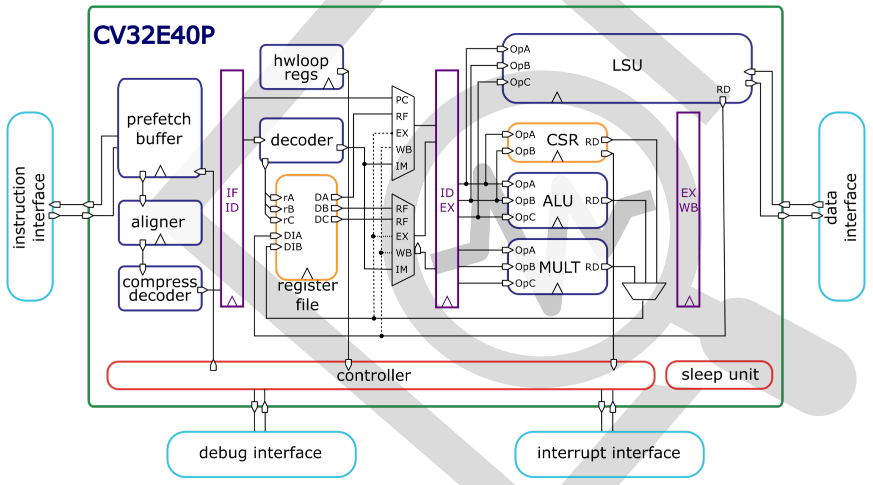
## QOS Processor application results

## Application example
CV32E40Pv1

**Design specification**
- 4-stage single-issue in-order pipeline
- OBI protocol memory interfaces
- Standard external debug and interrupt support
- Partial support for privileged spec 1.10
  - User Mode & physical memory protection

**Selection of issues reported**
- #132 Fetch side exception influences earlier instruction
- #136 Missed illegal exceptions
- #159 Wrong PMP computation
- #185 Exceptions update CSRs while in debug mode
- #533 Illegal instruction retires

**RV32IMC_Zicsr_Zifencei**



32-bit CV32E40Pv1 (RI5CY)

**23** bugs

**5** Standard extensions **21** Standard CSRs

**84** Standard instructions

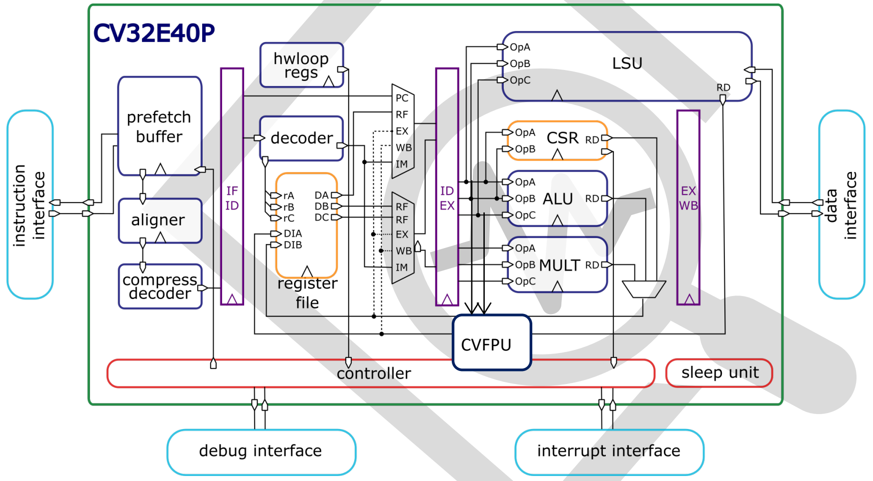
# Application example

CV32E40Pv2

## Design specification

- **+ Floating point & X custom instruction set extensions**
  - Post-incrementing load & store
  - ALU & Multiply accumulate
  - Single instruction multiple data (SIMD)
  - Hardware loops (zero-cycle branch)

**RV32**IMF**C**_Zicsr_Zifencei**_Zfinx_Xpulp_Xcluster**



32-bit CV32E40Pv2

## Selection of issues reported

- #722 Wrong instruction fetch caused by multicycle F instructions
- #723 Misaligned memory instructions set wrong memory access
- #725 No illegal instruction exception raised for non-Zfinx instructions
- #729 FMUL.S sets underflow flag of fflags wrongly
- #731 Custom Xpulp memory instructions set extra memory access
- #742 Simultaneous register file update by custom instructions

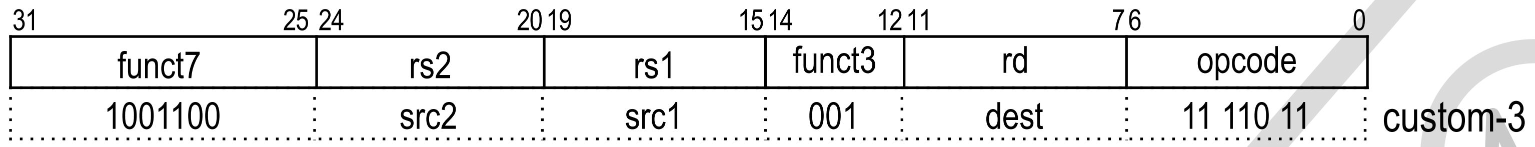
**31** bugs

| | | | |
|---|---|---|---|
| **+2** | Standard extensions | **+8** | Custom CSRs |
| **+2** | Custom extensions | **+320** | Custom instructions |

# X-extension verification effort is down to adding its specification

- **Example instruction: Sum of dot product on 2 vectors of four unsigned 8-b data**

cv.sdotup.b rd, rs1, rs2



rd = rd +

$$\sum_{k=0}^{3} rs1[8*(k+1)-1:8*k] \; * \; rs2[8*(k+1)-1:8*k]$$

- **User required input: provided using app's JSON format for regression runs**

| Name | Decoding | Execution | Restrictions |
|---|---|---|---|
| CV.SDOTUP.B | 1001100 rs2 rs1 001 rd/rs3 1111011 | X(rd) = X(rs3) +<br>X(rs1)[7..0] * X(rs2)[7..0] +<br>X(rs1)[15..8] * X(rs2)[15..8] +<br>X(rs1)[23..16] * X(rs2)[23..16] +<br>X(rs1)[31..24] * X(rs2)[31..24] | |

# Application example

## CV32E40Pv2

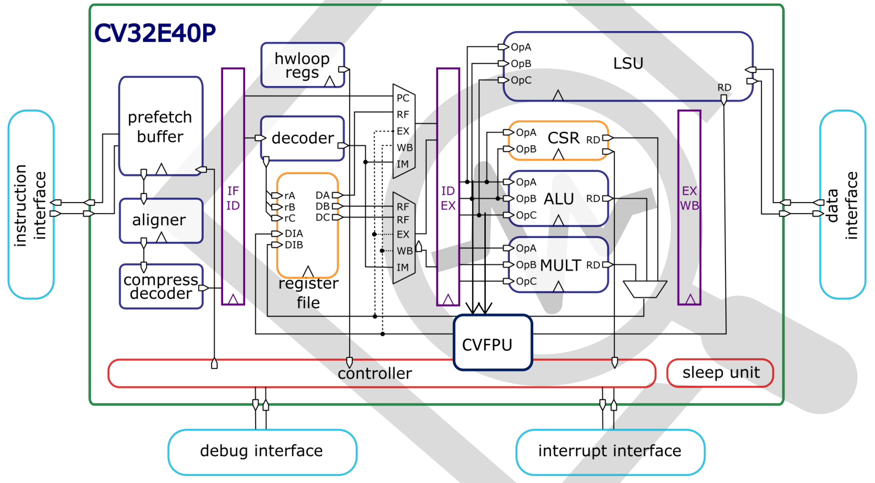
**5**
Configurations

**~400**
Assertions per CFG

**~2 hour**
Runtime of 70% of assertions per CFG

**100%**
Unbounded Proofs

**RV32**IMFC_Zicsr_Zifencei**_Zfinx_Xpulp_Xcluster**

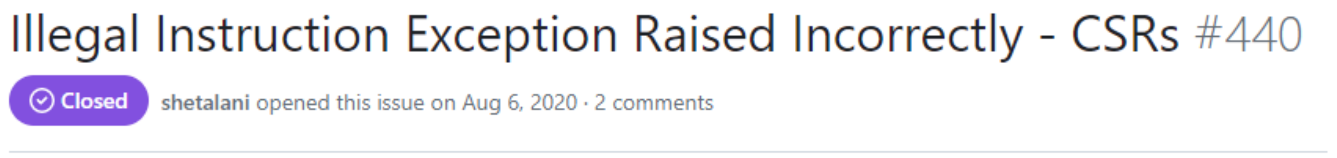


32-bit CV32E40Pv2

**31 bugs**

| | | | |
|---|---|---|---|
| **+2** | Standard extensions | **+8** | Custom CSRs |
| **+2** | Custom extensions | **+320** | Custom instructions |

# Bug case study

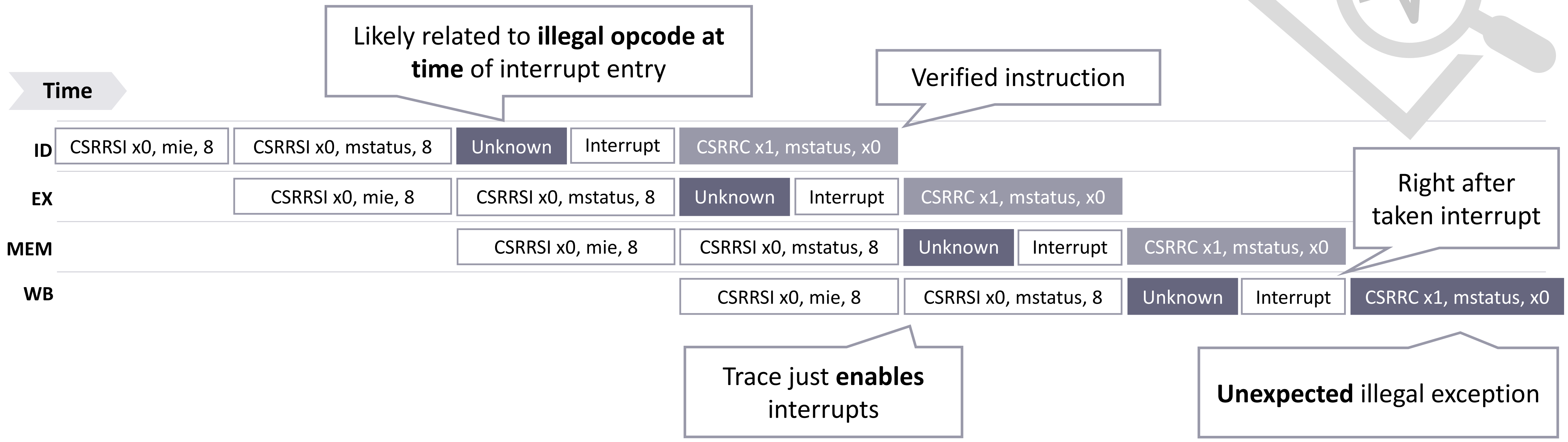


Illegal Instruction Exception Raised Incorrectly - CSRs #440

Closed shetalani opened this issue on Aug 6, 2020 · 2 comments

CV32E40Pv1

5 min



# Bug case study

Custom Xpulp memory instructions update register file wrongly | Simultaneous port writes #742

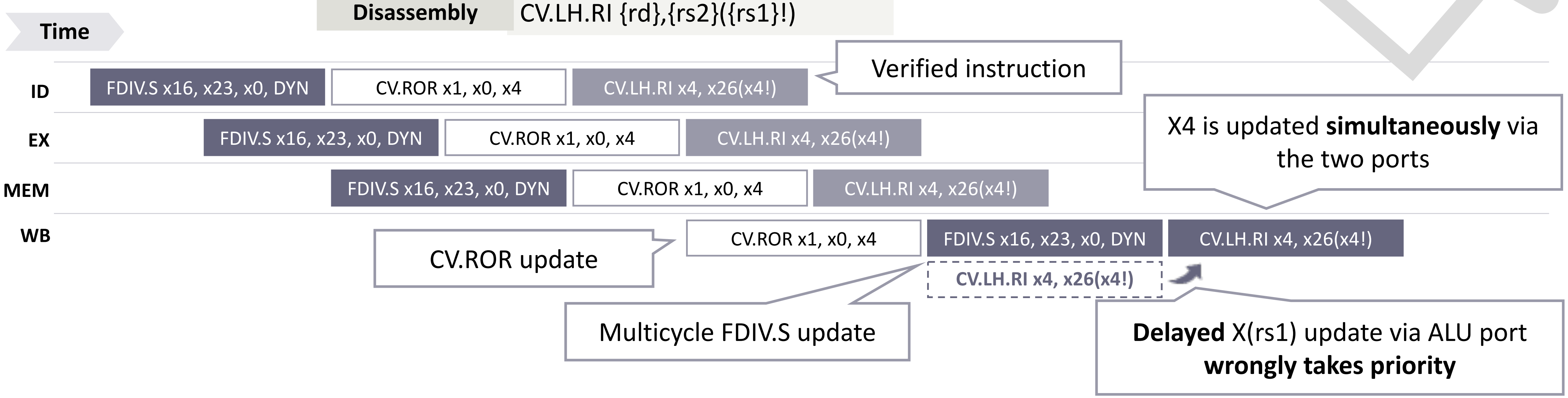
Open salaheddinhetalani opened this issue on Nov 28, 2022 · 0 comments

CV32E40Pv2

31 sec

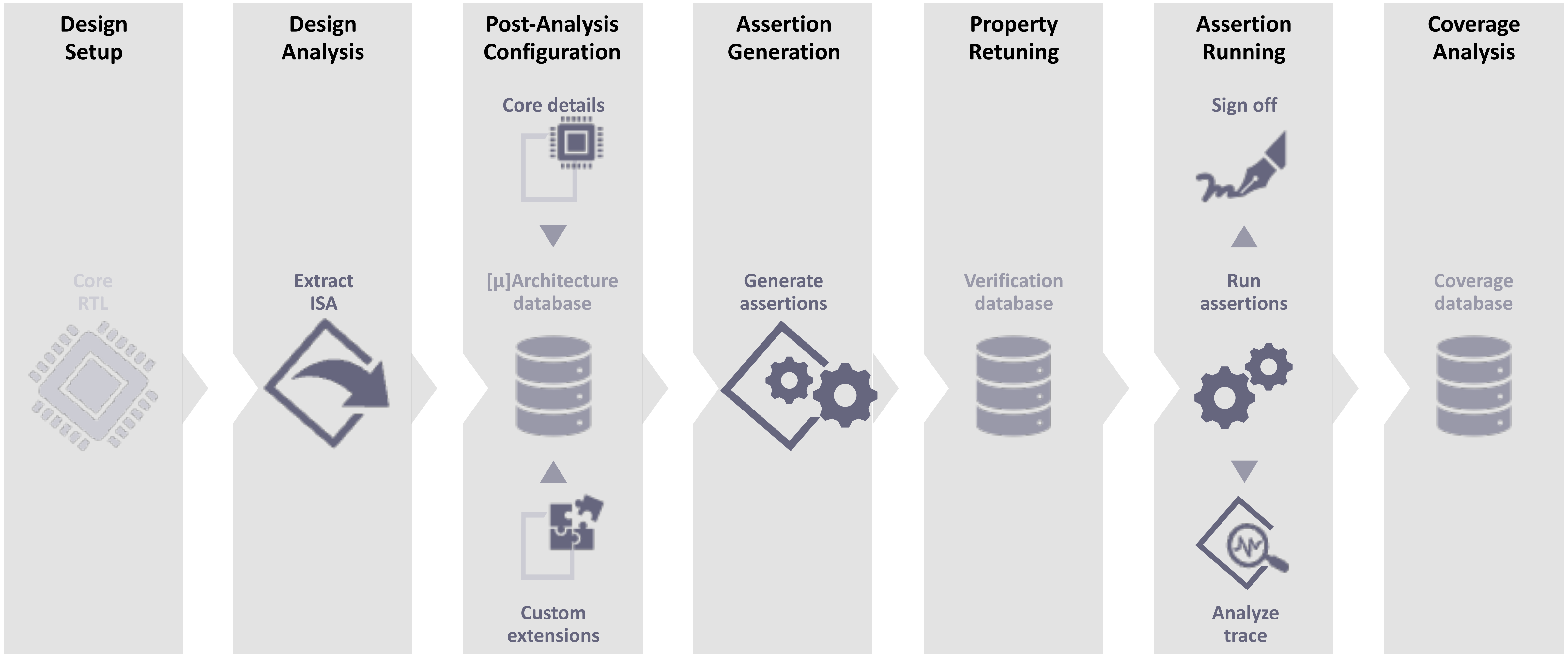
| | | |
|---|---|---|
| **Execution** | X(rs1) += X(rs2) | X4: Updated **first** via ALU port |
| | X(rd) = EXTS(mem16(X(rs1))) | X4: Updated **later** via LSU port |

**Disassembly** CV.LH.RI {rd},{rs2}({rs1}!)



## QOS Processor flow

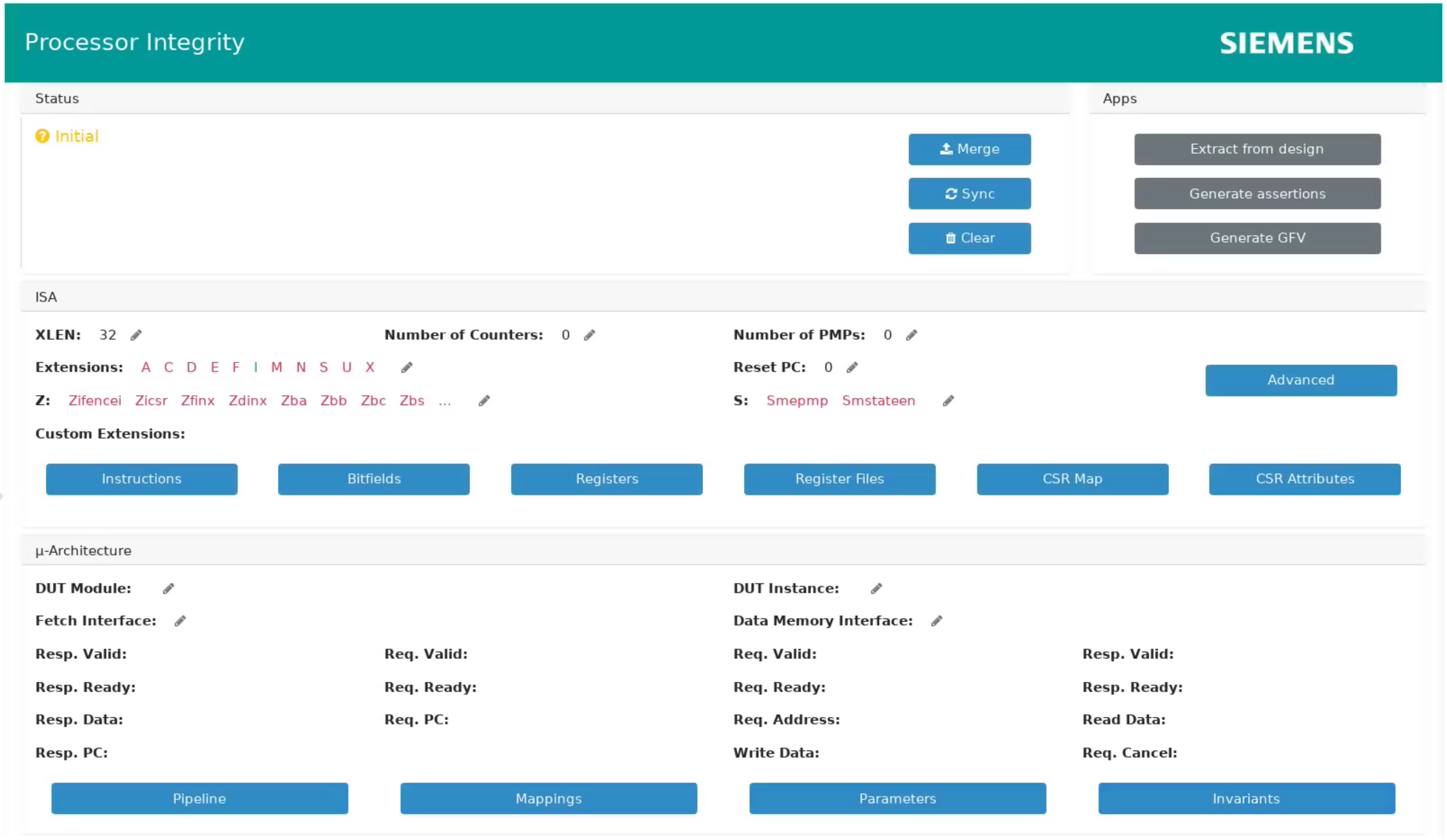
## App flow


Design Setup
Core RTL
Design Analysis
Extract ISA
Post-Analysis Configuration
Core details
[µ]Architecture database
Custom extensions
Assertion Generation
Generate assertions
Property Retuning
Verification database
Assertion Running
Sign off
Run assertions
Analyze trace
Coverage Analysis
Coverage database

# App GUI

**Design Setup**

Core RTL



Processor Integrity

SIEMENS

Status

Initial

Apps

ISA

**XLEN:** 32

**Number of Counters:** 0

**Number of PMPs:** 0

**Extensions:** A C D E F I M N S U X

**Reset PC:** 0

**Z:** Zifencei Zicsr Zfinx Zdinx Zba Zbb Zbc Zbs ...

**S:** Smepmp Smstateen

**Custom Extensions:**

μ-Architecture

**DUT Module:**

**DUT Instance:**

**Fetch Interface:**

**Data Memory Interface:**

| | | | |
|---|---|---|---|
| **Resp. Valid:** | **Req. Valid:** | **Req. Valid:** | **Resp. Valid:** |
| **Resp. Ready:** | **Req. Ready:** | **Req. Ready:** | **Resp. Ready:** |
| **Resp. Data:** | **Req. PC:** | **Req. Address:** | **Read Data:** |
| **Resp. PC:** | | **Write Data:** | **Req. Cancel:** |

Processor apps

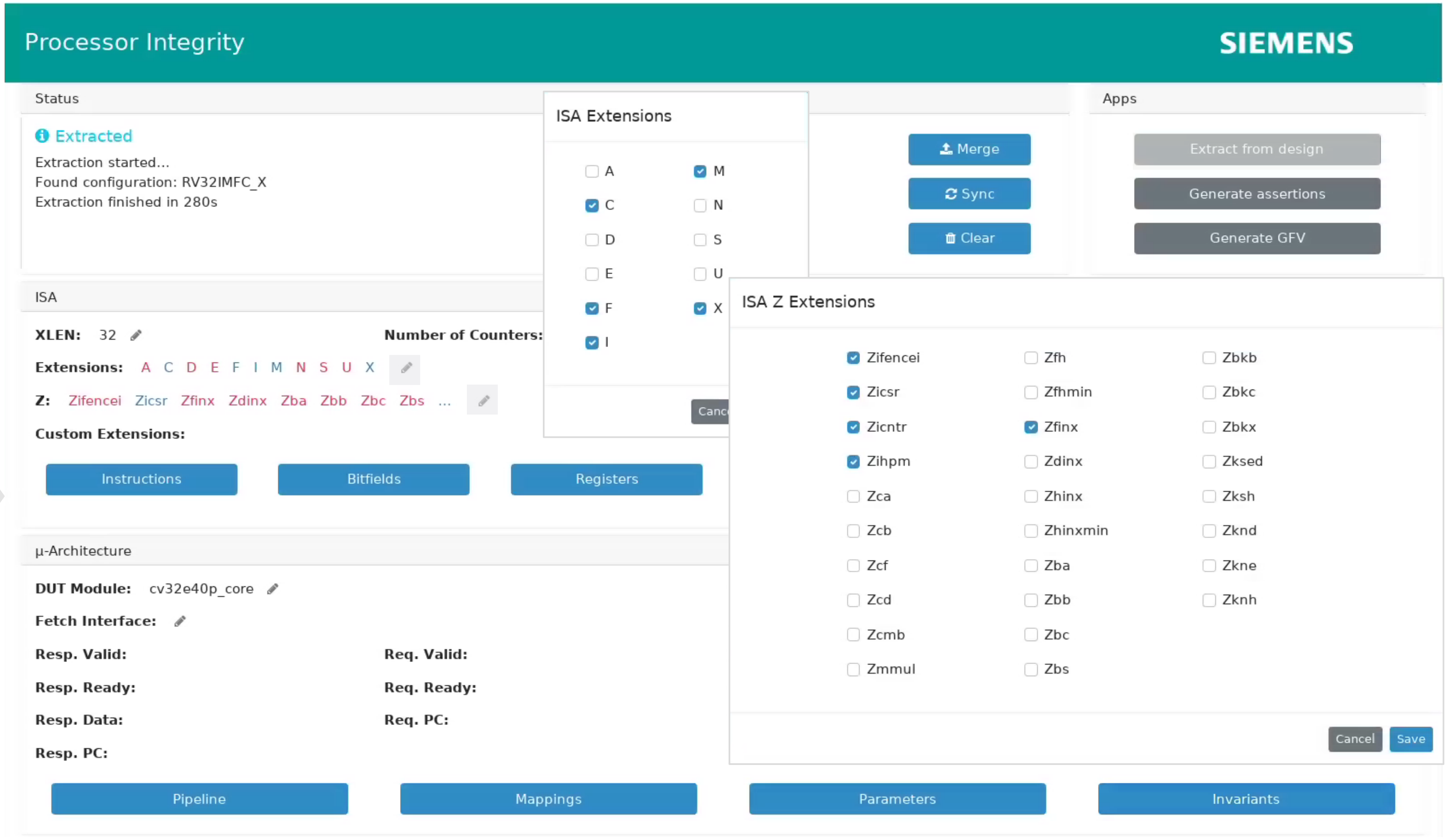
Design ISA information

Design μ-Architecture information

# Automated design analysis

**Design Analysis**

**Extract ISA**



Processor Integrity — SIEMENS

**Status**

Extracted

```
Extraction started...
Found configuration: RV32IMFC_X
Extraction finished in 280s
```

**ISA**

**XLEN:** 32

**Extensions:** A C D E F I M N S U X

**Z:** Zifencei Zicsr Zfinx Zdinx Zba Zbb Zbc Zbs ...

**Custom Extensions:**

**µ-Architecture**

**DUT Module:** cv32e40p_core

**Fetch Interface:**

| | |
|---|---|
| **Resp. Valid:** | **Req. Valid:** |
| **Resp. Ready:** | **Req. Ready:** |
| **Resp. Data:** | **Req. PC:** |
| **Resp. PC:** | |

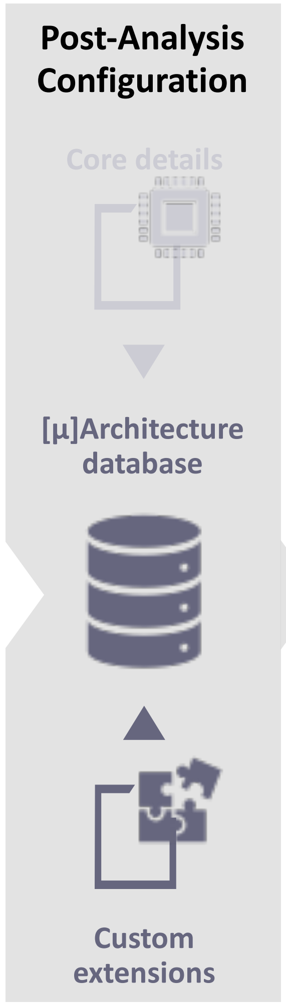
**ISA Extensions**

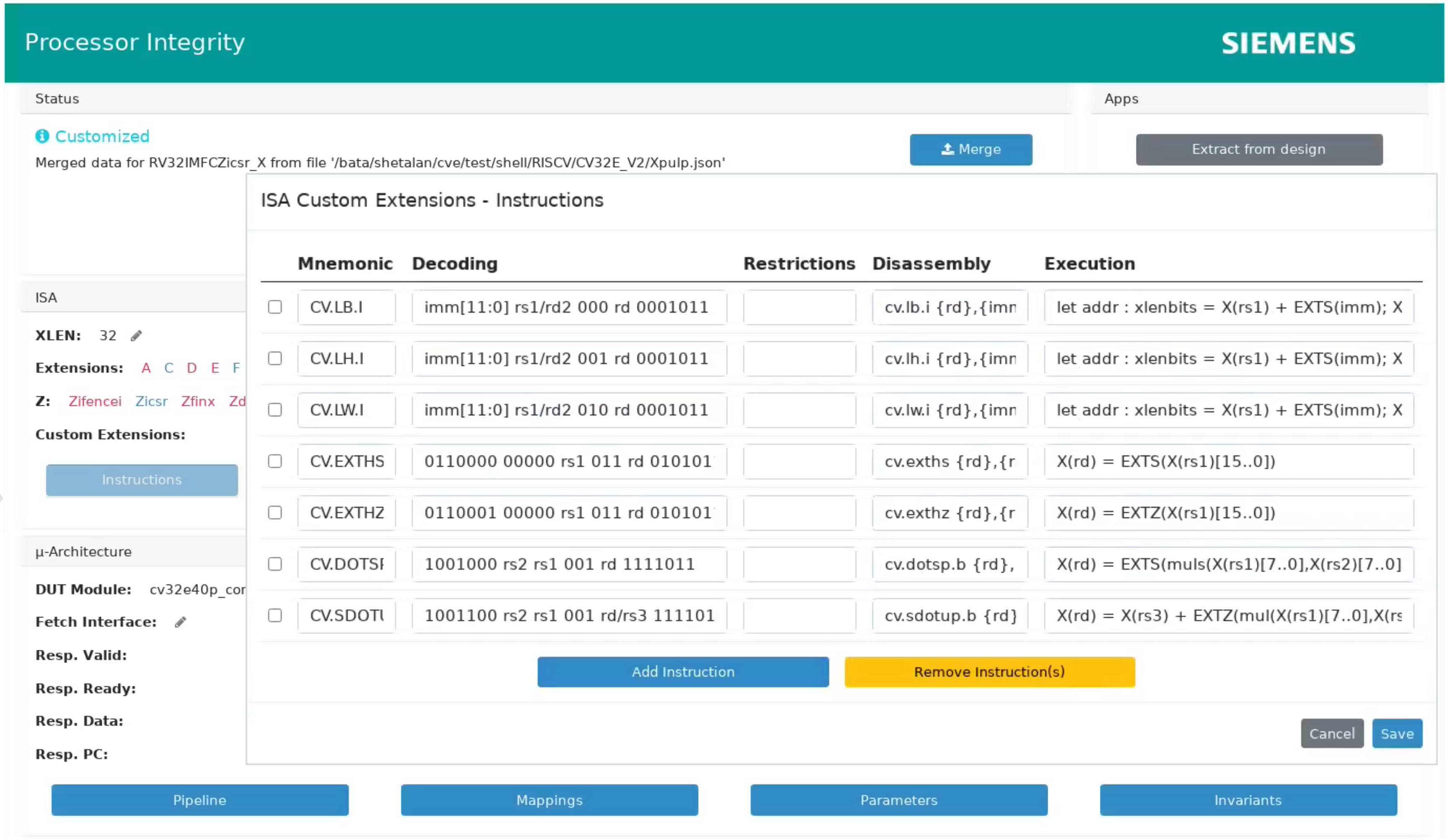
| | |
|---|---|
| [ ] A | [✓ M] |
| [✓ C] | [ ] N |
| [ ] D | [ ] S |
| [ ] E | [ ] U |
| [✓ F] | [✓ X] |
| [✓ I] | |

**ISA Z Extensions**

| | | |
|---|---|---|
| [✓ Zifencei] | [ ] Zfh | [ ] Zbkb |
| [✓ Zicsr] | [ ] Zfhmin | [ ] Zbkc |
| [✓ Zicntr] | [✓ Zfinx] | [ ] Zbkx |
| [✓ Zihpm] | [ ] Zdinx | [ ] Zksed |
| [ ] Zca | [ ] Zhinx | [ ] Zksh |
| [ ] Zcb | [ ] Zhinxmin | [ ] Zknd |
| [ ] Zcf | [ ] Zba | [ ] Zkne |
| [ ] Zcd | [ ] Zbb | [ ] Zknh |
| [ ] Zcmb | [ ] Zbc | |
| [ ] Zmmul | [ ] Zbs | |

# Post-analysis configuration





Status

Customized

Merged data for RV32IMFCZicsr_X from file '/bata/shetalan/cve/test/shell/RISCV/CV32E_V2/Xpulp.json'

Merge

Apps

Extract from design

ISA

XLEN: 32

Extensions: A C D E F

Z: Zifencei Zicsr Zfinx Zd

Custom Extensions:

Instructions

μ-Architecture

DUT Module: cv32e40p_cor

Fetch Interface:

Resp. Valid:

Resp. Ready:

Resp. Data:

Resp. PC:

ISA Custom Extensions - Instructions

| | Mnemonic | Decoding | Restrictions | Disassembly | Execution |
|---|---|---|---|---|---|
| ☐ | CV.LB.I | imm[11:0] rs1/rd2 000 rd 0001011 | | cv.lb.i {rd},{imn | let addr : xlenbits = X(rs1) + EXTS(imm); X |
| ☐ | CV.LH.I | imm[11:0] rs1/rd2 001 rd 0001011 | | cv.lh.i {rd},{imn | let addr : xlenbits = X(rs1) + EXTS(imm); X |
| ☐ | CV.LW.I | imm[11:0] rs1/rd2 010 rd 0001011 | | cv.lw.i {rd},{imn | let addr : xlenbits = X(rs1) + EXTS(imm); X |
| ☐ | CV.EXTHS | 0110000 00000 rs1 011 rd 010101 | | cv.exths {rd},{r | X(rd) = EXTS(X(rs1)[15..0]) |
| ☐ | CV.EXTHZ | 0110001 00000 rs1 011 rd 010101 | | cv.exthz {rd},{r | X(rd) = EXTZ(X(rs1)[15..0]) |
| ☐ | CV.DOTSI | 1001000 rs2 rs1 001 rd 1111011 | | cv.dotsp.b {rd}, | X(rd) = EXTS(muls(X(rs1)[7..0],X(rs2)[7..0] |
| ☐ | CV.SDOTI | 1001100 rs2 rs1 001 rd/rs3 111101 | | cv.sdotup.b {rd} | X(rd) = X(rs3) + EXTZ(mul(X(rs1)[7..0],X(rs |

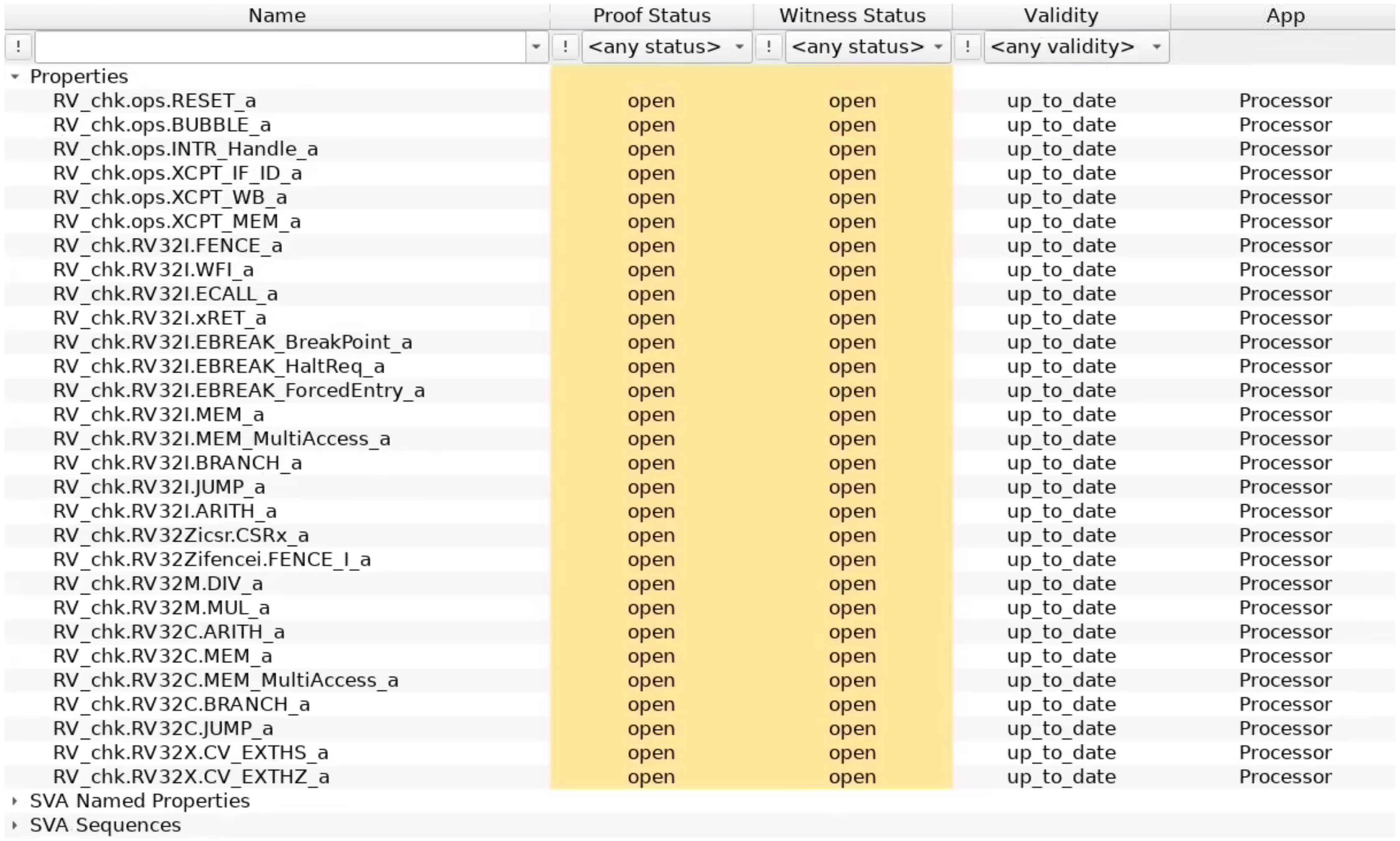
Add Instruction | Remove Instruction(s)

Cancel | Save

Pipeline | Mappings | Parameters | Invariants

# App assertions

Assertion Running
Sign off
Run assertions
Analyze trace



| Name | Proof Status | Witness Status | Validity | App |
|---|---|---|---|---|
| ! | ! <any status> | ! <any status> | ! <any validity> | |
| Properties | | | | |
| RV_chk.ops.RESET_a | open | open | up_to_date | Processor |
| RV_chk.ops.BUBBLE_a | open | open | up_to_date | Processor |
| RV_chk.ops.INTR_Handle_a | open | open | up_to_date | Processor |
| RV_chk.ops.XCPT_IF_ID_a | open | open | up_to_date | Processor |
| RV_chk.ops.XCPT_WB_a | open | open | up_to_date | Processor |
| RV_chk.ops.XCPT_MEM_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.FENCE_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.WFI_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.ECALL_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.xRET_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.EBREAK_BreakPoint_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.EBREAK_HaltReq_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.EBREAK_ForcedEntry_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.MEM_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.MEM_MultiAccess_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.BRANCH_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.JUMP_a | open | open | up_to_date | Processor |
| RV_chk.RV32I.ARITH_a | open | open | up_to_date | Processor |
| RV_chk.RV32Zicsr.CSRx_a | open | open | up_to_date | Processor |
| RV_chk.RV32Zifencei.FENCE_I_a | open | open | up_to_date | Processor |
| RV_chk.RV32M.DIV_a | open | open | up_to_date | Processor |
| RV_chk.RV32M.MUL_a | open | open | up_to_date | Processor |
| RV_chk.RV32C.ARITH_a | open | open | up_to_date | Processor |
| RV_chk.RV32C.MEM_a | open | open | up_to_date | Processor |
| RV_chk.RV32C.MEM_MultiAccess_a | open | open | up_to_date | Processor |
| RV_chk.RV32C.BRANCH_a | open | open | up_to_date | Processor |
| RV_chk.RV32C.JUMP_a | open | open | up_to_date | Processor |
| RV_chk.RV32X.CV_EXTHS_a | open | open | up_to_date | Processor |
| RV_chk.RV32X.CV_EXTHZ_a | open | open | up_to_date | Processor |
| SVA Named Properties | | | | |
| SVA Sequences | | | | |

RV32IMC
_Zicsr
_Zifencei

27

Assertions

# Application example

## CV32E40Pv2

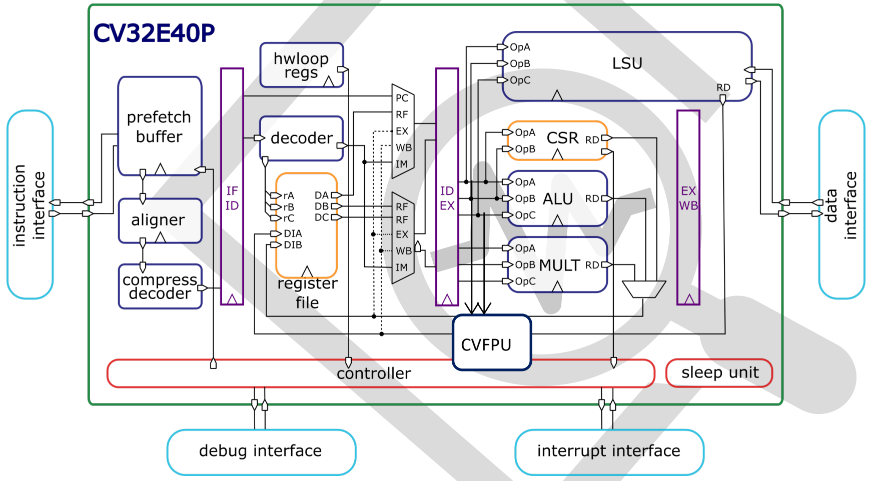
**Design specification**

- **+ Floating point & X custom instruction set extensions**
  - Post-incrementing load & store
  - ALU & Multiply accumulate
  - Single instruction multiple data (SIMD)
  - Hardware loops (zero-cycle branch)

**Selection of issues reported**

- #722 Wrong instruction fetch caused by multicycle F instructions
- #723 Misaligned memory instructions set wrong memory access
- #725 No illegal instruction exception raised for non-Zfinx instructions
- #729 FMUL.S sets underflow flag of fflags wrongly
- #731 Custom Xpulp memory instructions set extra memory access
- #742 Simultaneous register file update by custom instructions

**RV32**IMFC_Zicsr_Zifencei**_Zfinx_Xpulp_Xcluster**



32-bit CV32E40Pv2

**31** bugs

| | | | |
|---|---|---|---|
| **+2** | Standard extensions | **+8** | Custom CSRs |
| **+2** | Custom extensions | **+320** | Custom instructions |

# Summary

**Over 10x improvement**

On verification setup and runtime

**High degree of automation**

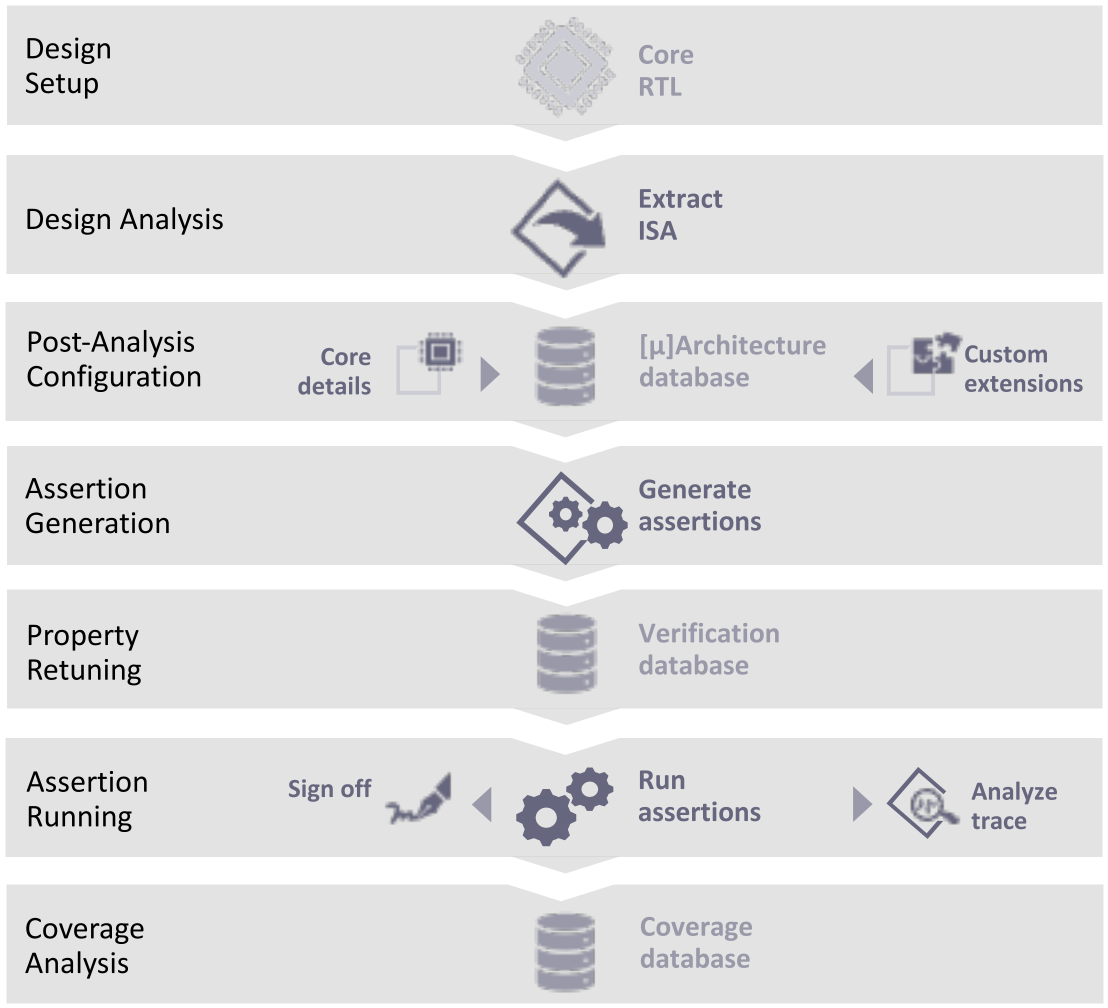
Designed to easily verify custom RISC-V cores

**Exhaustive & complete verification**

Leaves no bugs and exposes vulnerabilities

**Superior & unique**

Unrivalled expertise & leverage unique solutions



# Enabling high-quality processors

## Siemens EDA supports the RISC-V community

**Technology**

**Industry involvement**

**User community**

**QOS Processor verification**

- Core verification
- Integration verification

**RISC-V International**

**OpenHW Group**

**Scale4Edge project**

**Commercial solution adoption**

## Disclaimer

# Contact

Published by Siemens EDA

**Seiya Nakagawa**

Application Engineer

20F Gotenyama Trust Tower

7-35, Kita-Shinagawa 4-chome, Shinagawa-ku,

Tokyo 140-0001, Japan

**E-mail seiya.nakagawa@siemens.com**