CV32E40P UPDATES

CUSTOMIZING AN OPEN-SOURCE RISC-V CORE AT INDUSTRIAL-GRADE; EXPERIENCES AND CHALLENGES

Pascal Gouédo (pascal.gouedo@dolphin.fr) - Yoann Pruvost - Xavier Aubert - Olivier Montfort / Dolphin Design

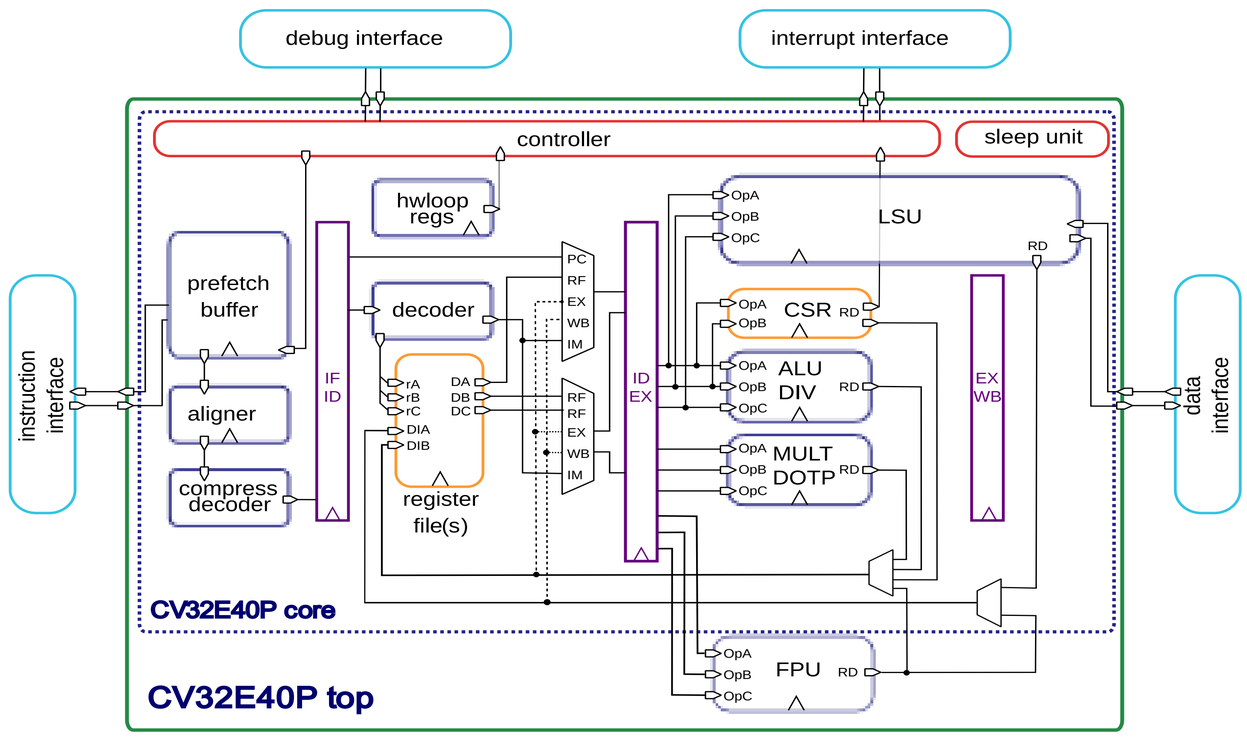
Pasquale Davide Schiavone - Mike Thompson / OpenHW Group

OpenHW group CORE-V : From CV32E40Pv1 to CV32E40Pv2

- 4-stage single-issue in-order pipeline
- OBI protocol memory interfaces
- Standard external debug and interrupt support

- **From PULP instructions to RISC-V X custom extensions**
  - **Post-incrementing load & store**
  - **ALU**
  - **Multiply Accumulate**
  - **8- & 16-b SIMD Instruction**
  - **Hardware Loops (zero-cycle branch)**
- **Configurable RTL**
  - **FPU (with parametrized latency)**
  - **ZFINX**
  - **COREV_PULP**
  - **COREV_CLUSTER**

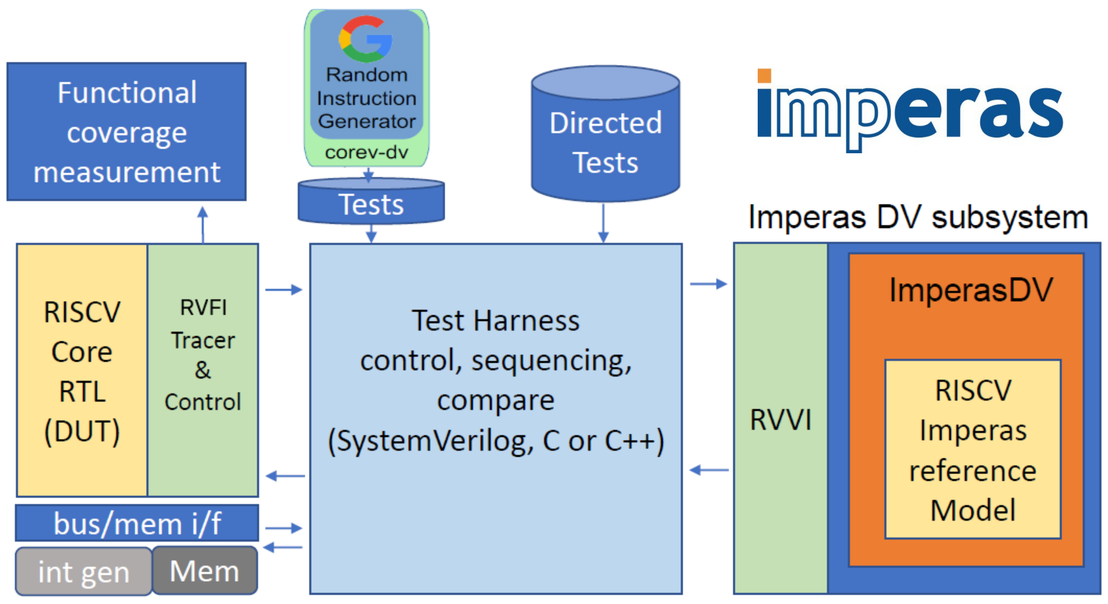
RV32IM[F]C_Zicsr_Zifencei[_Zfinx][COREV_PULP][COREV_CLUSTER]



7	Standard RISC-V Extensions	114	Standard instructions	21	CSRs	5	**RTL Parameters**
4	**New** Custom RISC-V Extensions	320	**New** Custom Instructions	8	**New** CSRs		

OpenHW group core-v-verif: Step&Compare to ImperasDV

CORE-V-VERIF



- Moved from Step&Compare to ImperasDV methodology
- COREV-DV instruction generator
  - Addition of Zfinx stream generator
  - Addition of 320 Pulp Instructions
- Used for everything not verified by Formal Verification
  - Asynchronous events (Interrupts, Debug, etc, ...)
  - Prefetcher Unit
  - Hardware Loops
  - Floating-Point Division and Square-Root

SOFTWARE TOOLCHAINS

- GCC upstream upgraded with 320 custom instructions
- Addition of ~200 builtin for C intrinsic usage
- Custom instructions automatic mapping for:
  - Post-incremented Load/Store
  - Branch with Register-Immediate comparison
  - 32-b Multiply-Accumulate
  - Hardware Loops
- LLVM toolchain upgrade on-going (due date e/o 2023)

Formal Verification

Based on SiemensEDA OneSpin 360
RISC-V Processor Verification app

From **Custom instructions description** using SAIL syntax
to **automatic assertions generation**

```
"custom_extensions": {
   "instructions":[
      {
         "name" : "CV.SDOTUP.B",
         "disassembly" : "cv.sdotup.b {rd},{rs1},{rs2}",
         "decoding" : "1001100 rs2 rs1 001 rd/rs3 1111011",
         "execution" : "X(rd) = X(rs3) + X(rs1)[7..0] * X(rs2)[7..0] +
                                 X(rs1)[15..8]   * X(rs2)[15..8] +
                                 X(rs1)[23..16] * X(rs2)[23..16] +
                                 X(rs1)[31..24] * X(rs2)[31..24]
      }
```

FORMAL VERIFICATION NUMBERS

5	Configurations	~400	Assertions per CFG
~2 h	Runtime for 70% of assertions per CFG	100%	Unbounded Proves

31 BUGS FOUND (IMPACT OF ENABLING FPU/PULP)