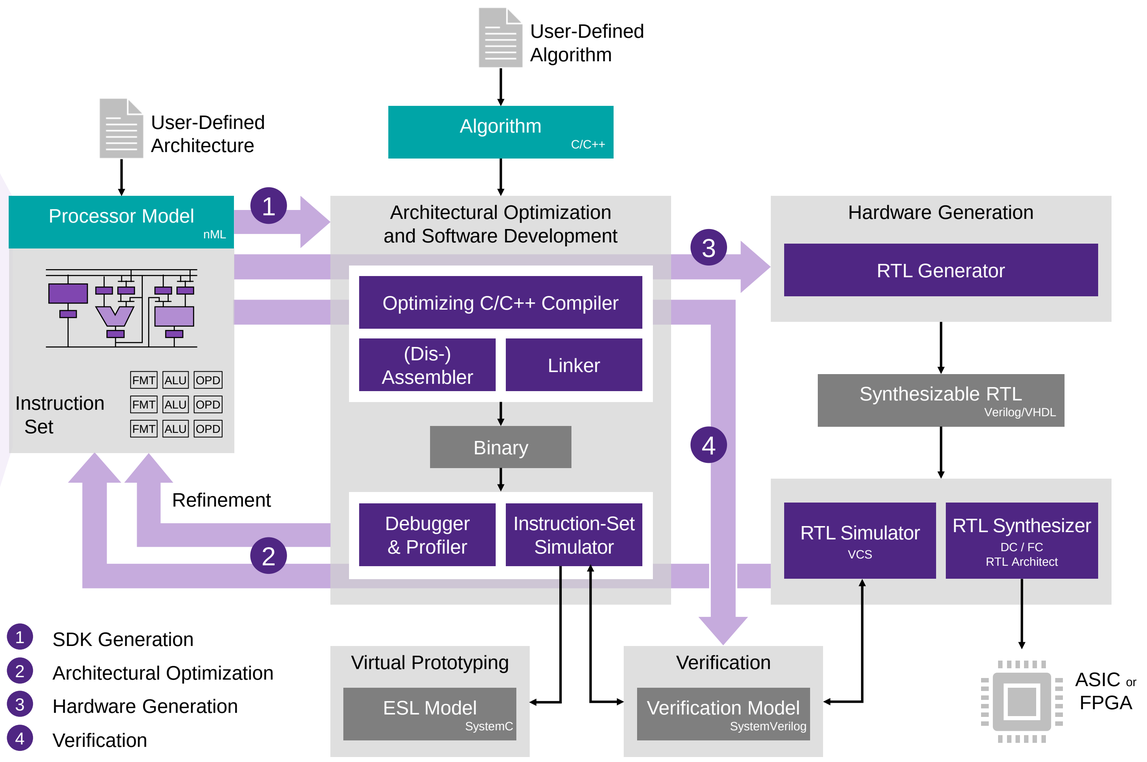
SYNOPSYS®

Our Technology, **Your Innovation**™

Creating Custom RISC-V Processors Using ASIP Design Tools: A Post-Quantum Cryptography Case Study

Gert Goossens Dominik Auras Werner Geurts

```
start trv32p5x;
opn trv32p5x(bit32_ifmt | bit16_ifmt);
opn bit32_ifmt(majOP | majOP_IMM | majLOAD | ... | majCUSTOM3);
opn majOP(alu_rrr_ar_instr | mpy_rrr_instr | div_instr);
opn alu_rrr_ar_instr(op: majOP_fn10, rd: eX, rs1: eX, rs2: eX){
  action{
  stage ID:
     pidX1 = r1 = X[rs1];
     pidX2 = r2 = X[rs2];
  stage EX:
     aluA = pidX1;
     aluB = pidX2;
     switch(op){
        case add:  aluR = add (aluA,aluB) @alu;
        case sub:  aluR = sub (aluA,aluB) @alu;
        case slt:  aluR = slt (aluA,aluB) @alu;
        case sltu: aluR = sltu(aluA,aluB) @alu;
        case xor:  aluR = bxor(aluA,aluB) @alu;
        ...
        case sra:  aluR = sra (aluA,aluB) @alu;
     }
  stage EX:
     pexX1 = texX1 = aluR;
  stage ME:
     pmeX1 = tmeX1 = pexX1;
  stage WB:
    if (rd: x0) w1_dead = w1 = pmeX1;
    else X[rd] = w1 = pmeX1;
  }
  syntax : "neg "  rd "," rs2 op<<sub>>  rs1<<x0>>
         | "snez " rd "," rs2 op<<sltu>> rs1<<x0>>
         | "sltz " rd "," rs1 op<<slt>>  rs2<<x0>>
         | "sgtz " rd "," rs2 op<<slt>>  rs1<<x0>>
         | op " "  rd "," rs1 "," PADOP2 rs2;
  image  : op[9..3]::rs2::rs1::op[2..0]::rd, class(alu_rrr);
}
...
```

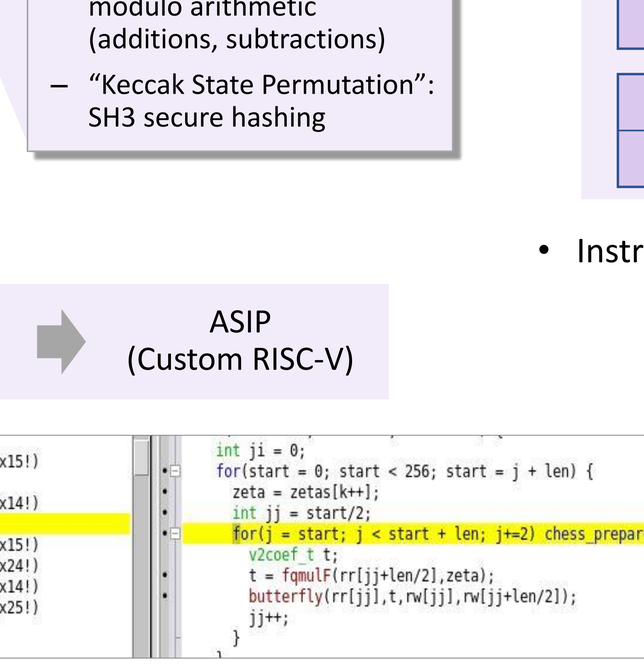


ASIP Designer™

- Industry-leading tool to design your own processor
- Language-based description of ISA and microarchitecture: nML
- Single processor model ensures that SDK and RTL are in sync
- Architectural exploration with Compiler-in-the-Loop™ & Synthesis-in-the-Loop™
- Licensed as a tool (not IP): product differentiation, no royalties
- Full interoperability with other Synopsys EDA tools

RISC-V Extensibility

- ISA customization & extensibility drive RISC-V adoption
  - Extension instructions can be encoded in RISC-V's reserved opcode space or in parallel issue-slots (VLIW)
- Result: RISC-V compatible Application-Specific Processor (ASIP)
  - Reuse SW code & interfaces designed for general-purpose RISC-V
- ASIP Designer supports the entire design process
  - nML models of Trv family are included with ASIP Designer tools
  - Designers extend these nML models as desired
  - Explore, leveraging Compiler-in-the-Loop, Synthesis-in-the-Loop
- Custom RISC-V cores complement ARC-V IP ▶



Trv (RISC-V) Models Shipped with ASIP Designer

Integer models: Trv<mm>p<n>

	32-bit datapath	64-bit datapath
3-stage pipe	Trv32p3 Trv32p3x	Trv64p3 Trv64p3x
5-stage pipe	Trv32p5 Trv32p5x	Trv64p5 Trv64p5x

- ISA: RV64IM, RV32IM
  - Integer & multiply
- Micro architecture
  - Protected pipeline, 3 or 5 stages
  - Hardware multiplier
  - Iterative divider
- Optional extensions: Trv<mm>p<n>x
  - Two-way static ILP
  - Zero overhead hardware loops
  - Load/store with post-modify addressing

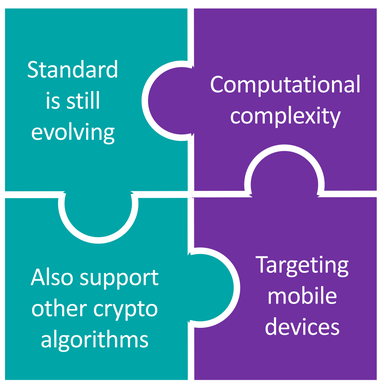
Floating-point models: Trv32p<n>f

	32-bit datapath
3-stage pipe	Trv32p3f Trv32p3fx
5-stage pipe	Trv32p5f Trv32p5fx

- ISA: RV32IMZfinx
  - Integer & multiply, single-prec. float
- Micro architecture
  - Protected pipeline, 3 or 5 stages
  - FPU based on HardFloat [Hauser]
  - Iterative divider & square-root
- Optional extensions: Trv32p<n>fx
  - Two-way static ILP
  - Zero overhead hardware loops
  - Load/store with post-modify addressing

Case study: Post-Quantum Crypto

- Acceleration of Kyber key encapsulation algorithm



- "Montgomery Reduction": modulo arithmetic (multiplications)
- "Barrett Reduction": modulo arithmetic (additions, subtractions)
- "Keccak State Permutation": SH3 secure hashing

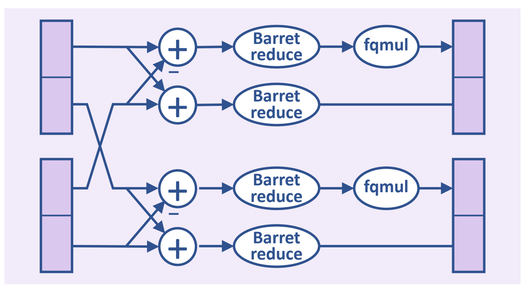
Software programmability ✕ Performance & Low power ➡ ASIP (Custom RISC-V)

```
3864   2  addi   x26, x26, -1
3868   2  addi   x11, x11, 2       | lw   x28, 4(x15!)
3876   2  addi   x12, x12, 2
3880   2  fqmf   x27, x28, x13     | lw   x17, 4(x14!)
3888   2  zlp    x26, 8,   32      | nop
3896   3  bfly   x16, x20, x17, x27 | lw  x28, 4(x15!)
3904   3  addi   x12, x12, 2       | sw   x16, 4(x24!)
3912   3  fqmf   x27, x28, x13     | lw   x17, 4(x14!)
3920   3  addi   x11, x11, 2       | sw   x20, 4(x25!)
3928   2  bfly   x16, x20, x17, x27
3932   2  sw     x16, 4(x24!)
```

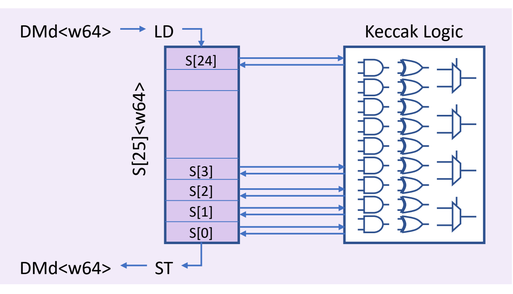
```
int ji = 0;
for(start = 0; start < 256; start = j + len) {
  zeta = zetas[k++];
  int jj = start/2;
  for(j = start; j < start + len; j+=2) chess_prepare_for_pipelining {
    v2coef_t t;
    t = fqmulF(rr[jj+len/2],zeta);
    butterfly(rr[jj],t,rw[jj],rw[jj+len/2]);
    jj++;
  }
```

▲ Fragment of compiled machine code, with corresponding C source code

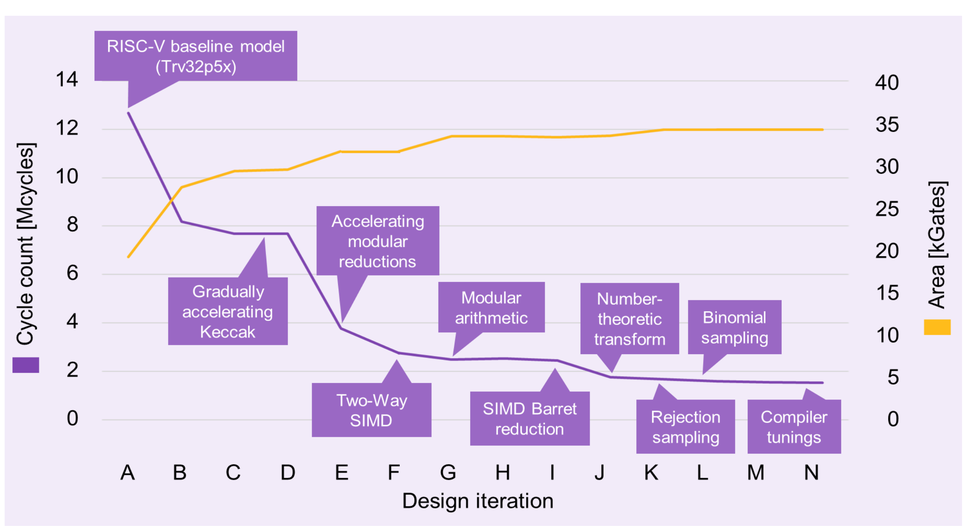
- 14 design iterations starting from Trv32p5x, executed in a CI/CD flow
- Montgomery and Barret Reduction:
  - Instruction fusion
  - Packed SIMD: 2x 13-bit → 32-bit registers



- Keccak State Permutation:
  - Instruction fusion
  - Custom multi-port register-file: parallel access to hashing state



- Instruction-level parallelism: e.g. [packed vector operation] || [load-store]



Design trajectory with evolution of cycle and gate count ▶

Acknowledgement – To Robin Geens, COSIC research group , KU Leuven, for his contributions to the design of the custom RISC-V processor for Kyber key encapsulation.

Take-Aways

- Designing custom RISC-V architectures with application-specific extensions yields product differentiation and superior PPA, while maintaining flexibility and eco-system compatibility
- ASIP Designer is the industry-leading processor design tool, taking the risk out of your RISC-V design optimization

More info

- www.synopsys.com/asip
- asip_info@synopsys.com
- Authors: <first_name.last_name>@synopsys.com