Rad-hard RISC-V SoC and ASIP Ecosystems Studies for High-Energy Physics Applications

Marco Andorno, on behalf of A. Caratelli, D. Ceresa, J. Dhaliwal, K. Kloukinas, R. Pejasinovic

2022/09/21

Outline

1. **Introduction**
   - Motivations and benefits
   - R&D context and timeline
2. **System-on-Chip platform**
   - Open-source cores comparison
   - SoC architecture
   - Radiation hardening techniques
3. **ASIP (Application Specific Instruction set Processor) cores**
   - Introduction to ASIP Designer
   - Optimization

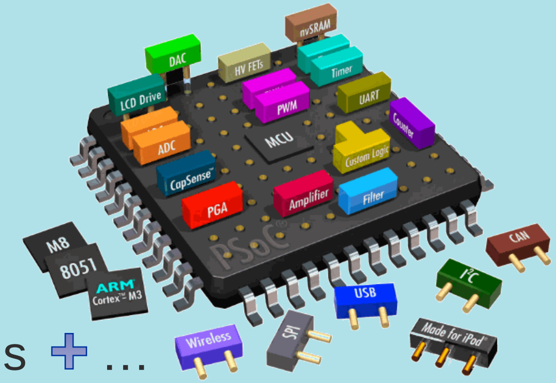
Introduction

Few definitions

System-on-Chip

Full system on a single IC

Processor core + Memories + Peripherals + …



RISC-V

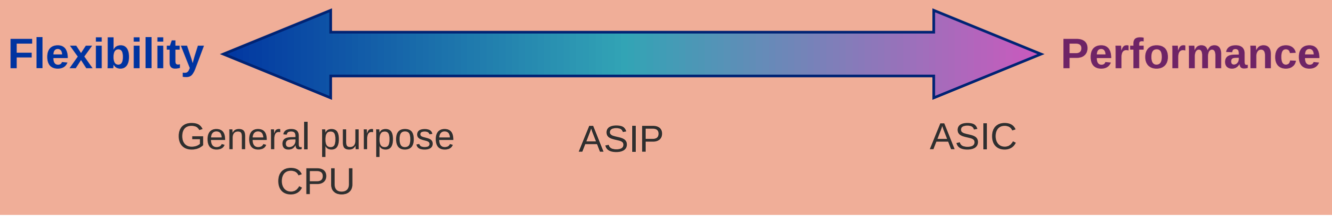
Open-standard instruction set
→ No license fees

Platinum Members

Few simple base instructions + Extensions (floating point, vector, …)

ASIP (Application-Specific Instruction set Processor)

Custom instruction set tailored to the application



Motivations and benefits

The increase in **design complexity** calls for the need of a more **abstract design methodology**

Programmability

- Allows retargeting an ASIC for a different application
- Can change the algorithm within the same application

Standard interconnect bus

- Promotes collaborative work
- Ensures compatibility of blocks designed by different teams

IP blocks library

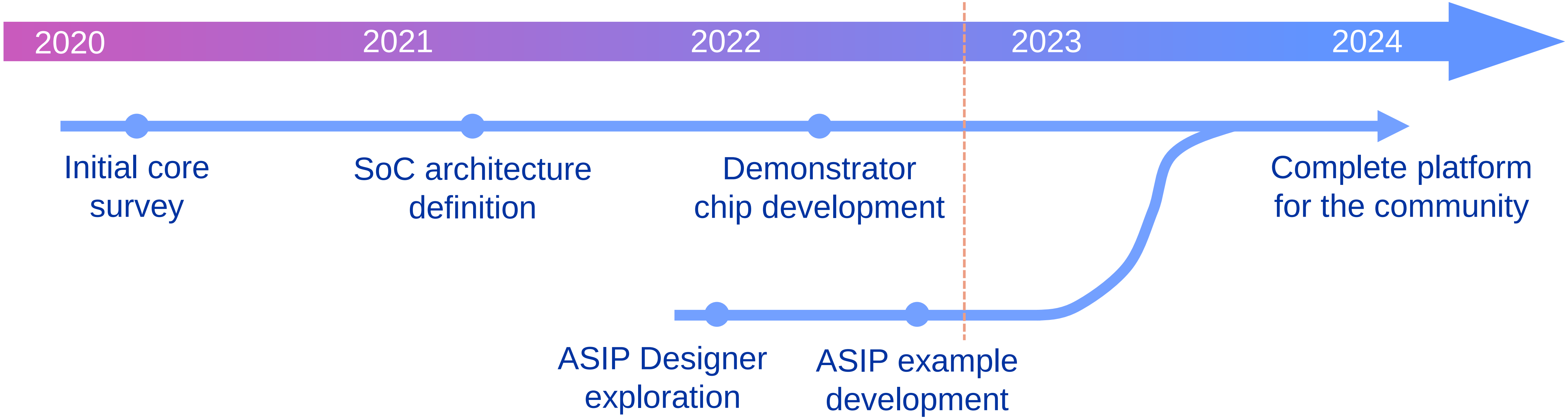
- Enables modularity with self-contained building blocks
- Helps design reusability

An SoC platform provides all this, leading to

- ⏩ Shift from design for an application to **design for resources**
- **Faster turnaround time**, both for design and verification

EP R&D context and timeline

Activities within the framework of the **EP R&D WP5 IC Technologies** (5-year initial plan)



2020
2021
2022
2023
2024
Initial core survey
SoC architecture definition
Demonstrator chip development
Complete platform for the community
ASIP Designer exploration
ASIP example development

Rad-tol SoC ecosystems for HEP applications

Two complementary R&D works ongoing

Control and monitoring

Data processing

- Standard open-source RISC-V ISA
- SoC interconnect and IP blocks for the community
- Fully radiation tolerant

- ASIP (Application Specific Instruction set Processor) based
- ASIP Designer by Synopsys
- Custom optimized ISA and microarchitecture

System-on-Chip Platform

Initial RISC-V cores and SoC platforms survey

RISC-V Cores	Related SoC	
EH1, EL2, EH2 *(Western Digital)*	SweRVolf	
VexRiscv *(SpinalHDL)*	Briey SoC	
Rocket Core *(UC Berkeley)*	Rocket Chip	*Promising chip-building framework in Chisel*
PicoRV32	PicoSoC	*Simple, written in pure Verilog*
Ibex *(ETH Zurich)*	Pulpissimo	*Direct collaboration with ETH*
XuanTie CPUs *(Alibaba)*		
SiFive CPUs *(RISC-V designers)*		

Selected for more thorough comparison

Notable, but commercial license

Open-source cores comparison

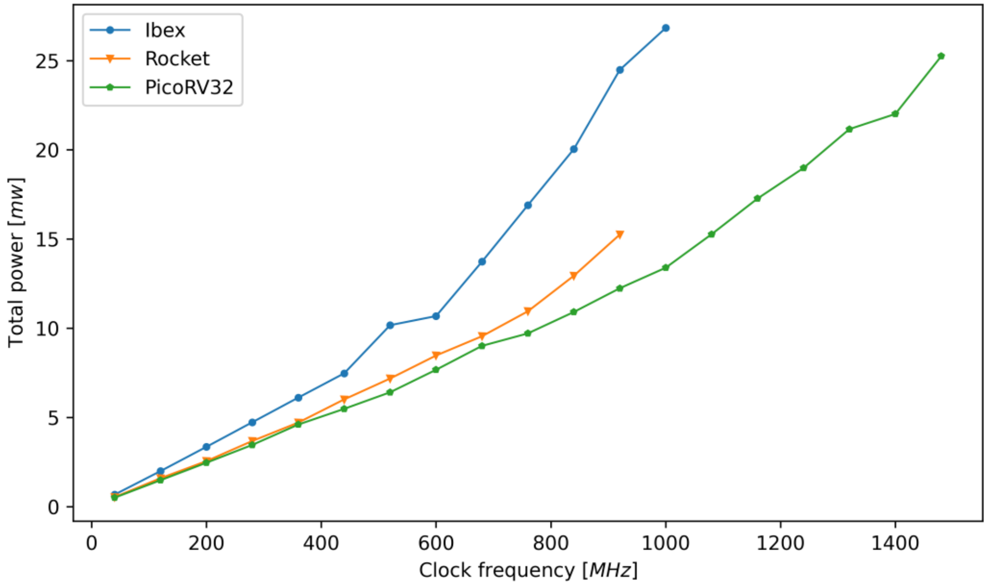
Three RISC-V open-source cores were evaluated initially:

	Ibex	Rocket Core	PicoRV32
Developers	lowRISC (prev. ETH Zurich)	CHIPS Alliance (prev. UC Berkeley)	Independent
HDL	SystemVerilog	Chisel (generates Verilog)	Verilog
License	Apache License 2.0	BSD 3-clause	ISC
ISA	RV32I[E,M,C,B]	RV[32,64]I[M,A,F,D]	RV32I[E,M,C]
Pipeline	2-stage in order	5-stage in order	None
Memory interface	req/gnt (separate for instr. and data)	TileLink (adapts to AXI/APB)	valid/ready (shared for instr. and data)
CPI	≥ 1	≥ 1	~ 4
Power*	6.11 mW	4.71 mW	3.97 mW
Area*	8543.3 µm^2	9427.6 µm^2	6141.9 µm^2

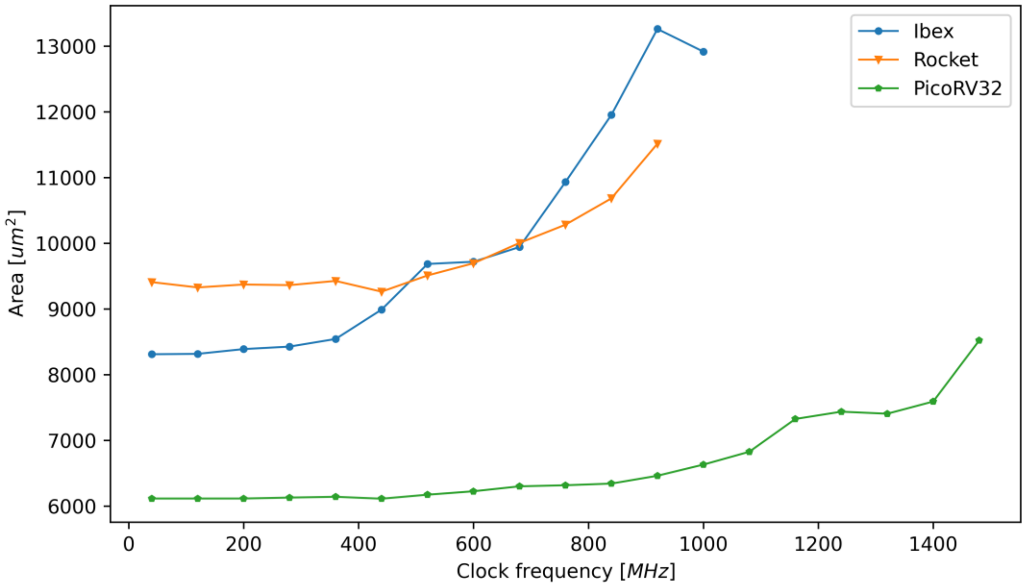
**Post synthesis results on TSMC 28 nm technology @ 320 MHz, using low, standard and ultra-high Vt transistors.*

Open-source cores comparison

Power



Area



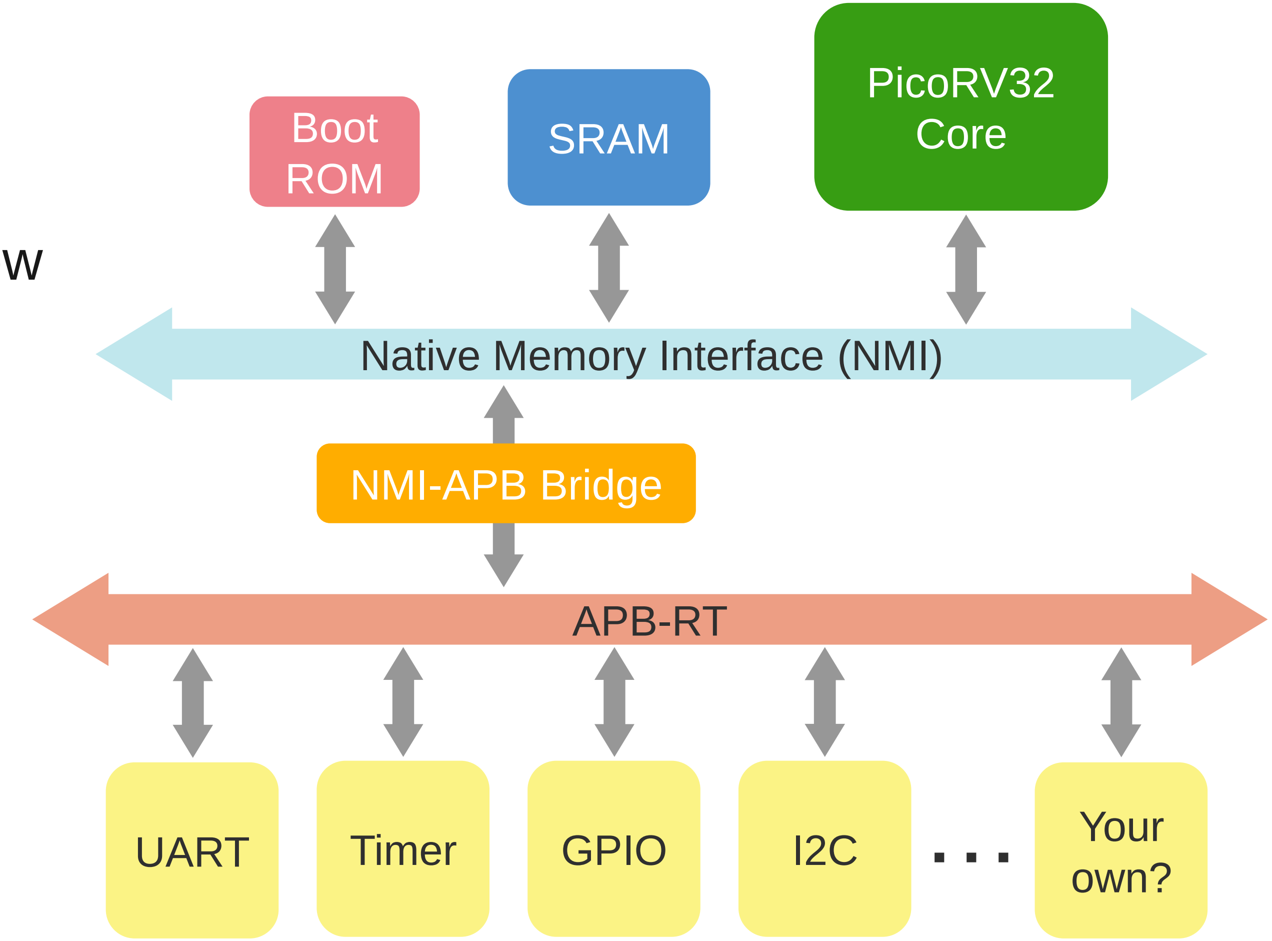
For control and monitoring applications speed is not a requirement
→ selection based on **power consumption** and **area**

Post-synthesis results show a clear winner: **PicoRV32** was selected as the SoC core

SoC architecture

Specifications:

- Microcontroller-style (small, low power, low performance)
- RV32I instruction set (no multiplier/FPU)
- Radiation-tolerant
- APB-RT bus (from ARM AMBA) with a set of peripherals
- Simple programming interface



Radiation hardening techniques

- **Triple Module Redundancy (TMR)** where possible

- Full TMR on the APB interconnect would be too much overhead, so we use **Hamming Encoding** for error correction/detection → **APB-RT**
  - Control signals are still fully triplicated (`pwrite`, `psel`, `pstrb`, `pready`, `pslverr`)
  - 32-bit data and address buses are encoded by byte with Hamming(13,8), to permit byte access operations
    - 4 parity bits for single error correction
    - 1 extra parity bit for additional double error detection

    SECDED
  - Placement constraints on buffers to avoid multi-bit upsets

SoC platform recap

Some possible applications:

- Slow control and monitoring on detector chips
- Standalone radiation-tolerant microcontroller
- Embedded microcontroller on a pixel chip to run routines like pixel-matrix calibration

What we’re building now is a **demonstrator chip** to showcase this design paradigm

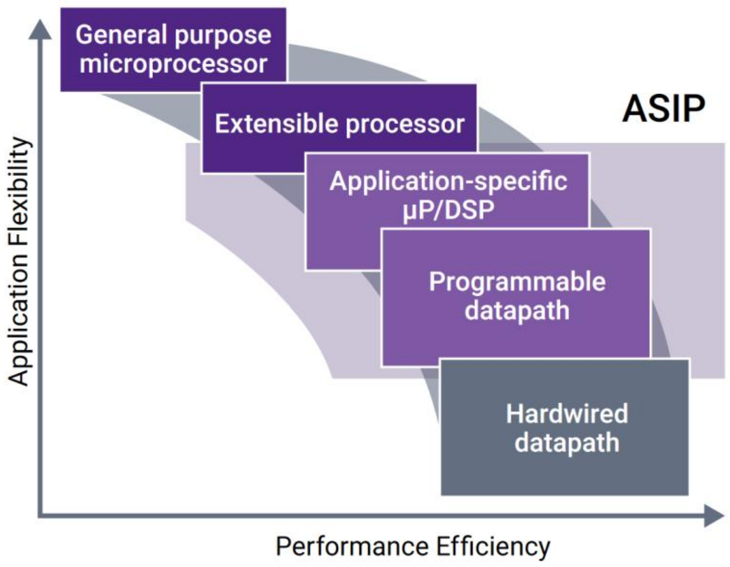
↓

The goal for the future is to provide this as a ***flexible, standardized and open platform*** *for designers in the community to* ***build and implement their own SoC****, with their own peripherals and custom hardware blocks.*

ASIP Cores

(Application Specific Instruction set Processor)

Data processing requirements



Beyond a general-purpose microprocessor:

- Maximize Performance/Power ratio
- Application specific solution

➡ Specialized **hardware accelerators** help offload the CPU.

Two possibilities for them:

Custom logic

👍 Can be easier to design

👎 No flexibility or programmability

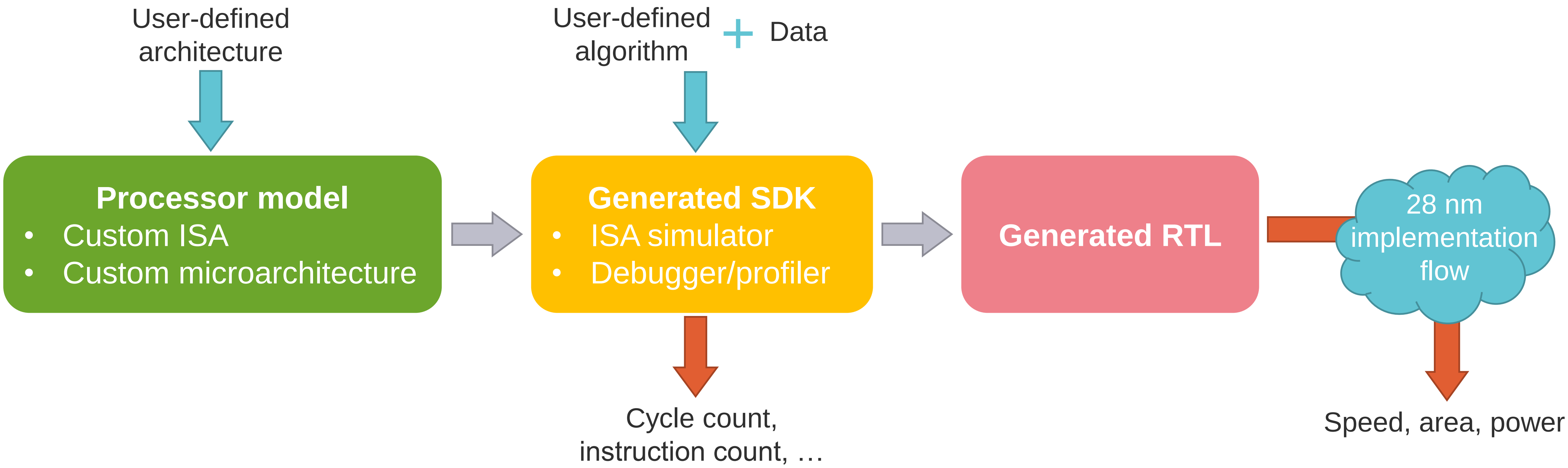
ASIP

👍 Software programmable

👎 High development effort (hardware + SDK)

Introduction to ASIP Designer

Tools like **Synopsys ASIP Designer** help with the hardware/software codesign of ASIPs:



User-defined architecture

User-defined algorithm + Data

Processor model
- Custom ISA
- Custom microarchitecture

Generated SDK
- ISA simulator
- Debugger/profiler

Generated RTL

28 nm implementation flow

Cycle count, instruction count, …

Speed, area, power

Core comparison - Implementation

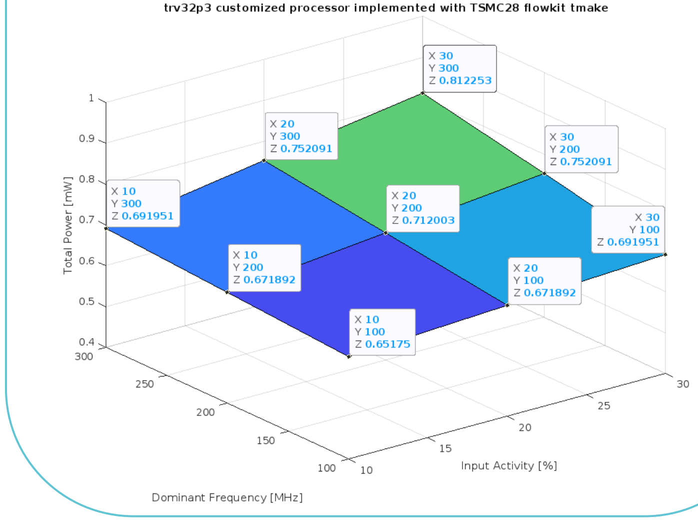
To showcase the workflow of ASIP Designer, two processor cores were compared using a clustering and filtering algorithm for pixel hit events:

TRV32P3

- 32-bit RV32IM
- 3-stage pipeline
- 32 registers

- Area: 11655 um^2
- Power: 0.81 mW

trv32p3 customized processor implemented with TSMC28 flowkit tmake



Implemented on TSMC 28 nm

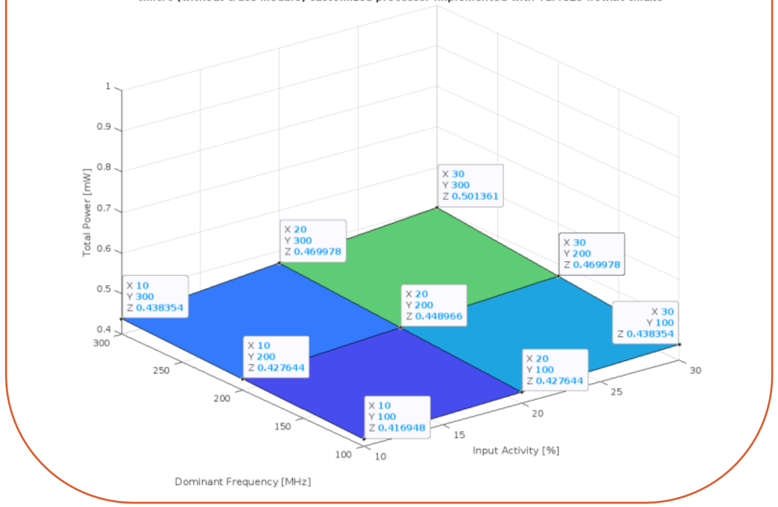
- *standard Vt*
- *333 MHz clock*
- *30% activity*

TMICRO

- 16-bit custom instructions
- 3-stage pipeline
- 8 registers

- Area: 6147 um^2
- Power: 0.50 mW

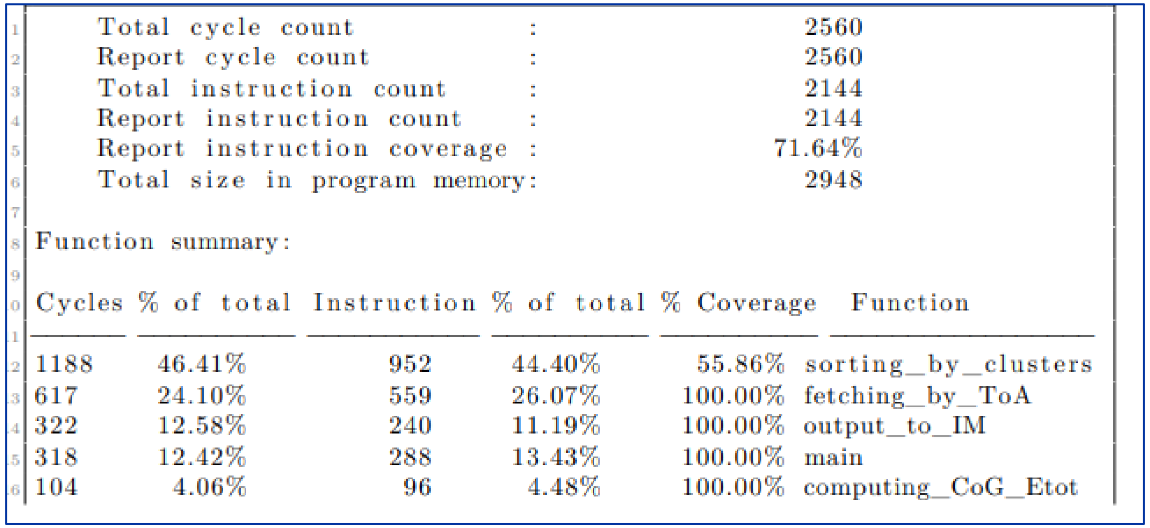
tmicro (without trace module) customized processor implemented with TSMC28 flowkit tmake



Core comparison - Profiling

ASIP Designer offers a number of profiling metrics:

- Instructions
- Functional units
- Model coverage
- Hazards
- Memory accesses



```
    Total cycle count           :               2560
    Report cycle count          :               2560
    Total instruction count     :               2144
    Report instruction count    :               2144
    Report instruction coverage :             71.64%
    Total size in program memory:               2948

Function summary:

Cycles % of total  Instruction % of total % Coverage   Function
------ ----------  ----------- ---------- ----------  ----------------
1188      46.41%           952     44.40%     55.86%  sorting_by_clusters
617       24.10%           559     26.07%    100.00%  fetching_by_ToA
322       12.58%           240     11.19%    100.00%  output_to_IM
318       12.42%           288     13.43%    100.00%  main
104        4.06%            96      4.48%    100.00%  computing_CoG_Etot
```

These can be used to direct the optimization

Core comparison - Optimization

From profiling data, several algorithm-specific optimizations were implemented:

Area and power reduction

- *Resource optimization*: removed unused divider block
- *Instruction pruning*: profiling shows around 30% instructions are never used, so can be removed in the ISA

Performance enhancement

- *Custom instructions*: memory accesses are always of 2 or 3 words at a time, so defining new instructions to read multiple words at once improves latency
- *Dedicated accelerators*: a MAC unit was introduced to speed up repetitive operations

ASIP recap

- ASIPs offer a good tradeoff between the flexibility of a general purpose processor and the performance of custom logic.
- Tools can help deal with the complexity of designing the hardware and software together and optimizing the architecture.
- ASIPs allow to add **on-chip data processing** capabilities to pixel chips, detector electronics and SoCs in general.

Conclusions

Summary and conclusions

We have demonstrated a workflow for both control and data processing applications:

General-purpose RISC-V processor

Standard SoC interconnect bus

Radiation tolerance techniques

Application-specific ASIP

Application-optimized ISA and architecture

Example of on-chip data processing solution

Main takeaway

Designing for resources and programmability are keys to **cost-effective**, **collaborative** and **fast** silicon design.

Thank you!

CERN

home.cern