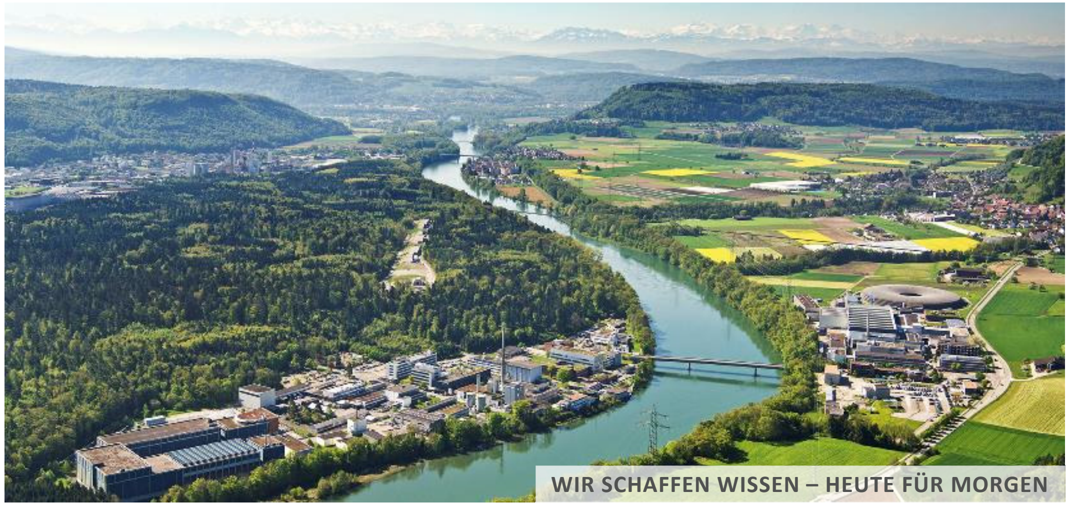
PAUL SCHERRER INSTITUT



Boris Keil :: Paul Scherrer Institute

Generic Hardware, Firmware and Embedded Software Platforms for Particle Accelerators

IBIC 2019, Malmö, Sweden, Sept. 11, 2019

PAUL SCHERRER INSTITUT

Preface: Definitions

Generic:

– Can be shared / re-used by different (ideally many) types of systems / applications

Platform:

– Set of components, base to build different types of new systems / applications
  - Hardware, firmware, software
  - Standards and technologies (electronics/crates, communication protocols, FPGA or CPU families, …)
  - Development tools & languages
  - Design libraries (HW, FW, SW), repositories & versioning systems
  - Rules (use of repositories, coding, …)
  - …

Preface: Scope of This Presentation

- General aspects of generic **FPGA-based** HW/FW/SW platforms.
- Overview of (some) technologies (HW-centric) used by & relevant for such platforms, and their impact. Focus on future.
- 20 years ago: More specific, less generic solutions. Today: Everything becomes (more) generic -> only few examples for illustration. *Not* trying to give overview of all present & future generic platforms being used world-wide.
- Pro’s and con’s: May be subjective, differ between accelerators, institutes/labs, and use cases.

Contents

Introduction: Specific vs. Generic

Past: Application-specific solution

Today: Generic multi-purpose multi-sensor solution (with parasitic phone functions)

"Please, Siri, tell me the beam position."

"I'm afraid I can't do that, Dave."

New generic platforms replace older more specific solutions, and more (parts of the brains & skills of its users, ...). Pro: More features, automation & "intelligence". Con: Sometimes new challenges for design, use and maintenance ...

PAUL SCHERRER INSTITUT PSI

Introduction: Example Systems

Blue = Generic
Red = System-Specific

	RF Button BPM	X-Ray BPM (RF matched)	Beam Loss Monitor
Detector & Front-End electronics	4 Button Electrodes	Diamond or SiC 4-Quad Detector	Cherenkov Fiber + Fast Photodiode
Signal	(sub-)ns Pulse(s)		
Digitizer / FPGA Board*	Common (suitable) FPGA/SoC + ADC / RFSoC Board		
Firmware + Software	Front-End control	Front-End control	Loss Locator, ...
Firmware + Software	Interfaces to ADC, Control & Timing System, ...		
High-Level Apps, Config Files, ...	BPM Apps	X-BPM Apps	BLM Apps
Design tools, repositories, rules, ...	HW: Mentor Graphics, FW/SW: Xilinx Vivado & ARM Tools, GIT repository, GUI Builder & SW Libraries		

** Using same ADC may be possible but not optimal (depending on requirements)*

Introduction: Benefits & Limitations

Generic platforms: Benefits ...

- Sharing (many) components (can) save work/costs (design, maintenance, ...)

... and Limitations:

- "One-fits-all" solution may be overkill (cost, complexity), mix of different (2+) solutions/platforms may be better (low-end/high-end for 95%/5% of systems), or unavoidable (if you buy systems & choice is limited).
- PSI: New & old (40+ year) accelerators: Must use different platform generations
- A bit more diversity can be beneficial or necessary for accelerator performance (e.g. different ADC board types, ...) and reduction of risks and dependencies

No "black and white": Different applications/systems or platforms may not share everything, but something (also tools, standards, repositories, languages, libraries, ...).

PAUL SCHERRER INSTITUT PSI

Contents

Hardware & Technologies: FPGA History

Xilinx: 90% of FPGA sales to few big customers driving new technologies & features

Years	Technology	Typical General Applications	Price Range	# Logic Cells*
1984+	First COTS **FPGA** (XC2064)	Replace zoo of simple logic ICs (AND, OR, flip-flop, ...)	low ($50)	~100
2002+	First **SoC** (System-on-(Programmable)-Chip): FPGA + small CPUs (2 PowerPC405) + multi-gigabit transceivers. No dedicated CPU IOs.	Defense, instrumentation, ... (replaces smaller ASICs)	low ... high	~3k ... 100k
2015+	First **MPSoC** (Multiprocessor-SoC): 2 CPUs (2-core 32-bit + 4-core 64-bit ARM), dedicated CPU IOs, ~18 GByte/s DRAM	Driver assistance, image recognition, ...	medium ... high	~100k ...1M
2017+	First **RFSoC**: MPSoC with (many) on-chip multi-GSPS/GHz ADCs & DACs	5G, autonomous driving, defense, ...	high ($4k-10k)	~1M
2018+	First **ACAP** (Adaptive Compute Acceleration Platform): MPSoC + AI/DSP-accelerators	AI, 5G, data center, high performance computing, ...	high	~2M (excl. AI/DSP)

** Logic cell/element: Primitive logic block (few gates/flip-flops/LUTs) with configurable function (AND, OR, ...) & interconnect*

PAUL SCHERRER INSTITUT

PSI

Hardware & Technologies: FPGA Market

Company	Market Share 2018	SoC: Hard CPU Cores (On Same Chip As FPGA)	Soft CPU Cores Supported by FPGA Vendor *	FPGA Type Portfolio
Xilinx	51%	2-Core 32-bit & 4-Core 64bit ARM, 128-400 RISC cores	Microblaze, ARM, ...	FPGA, SoC, MPSoC, RFSoC, ACAP
Intel (ex Altera)	37%	4-Core ARM	NIOS II, Freescale, ARM, …	FPGA, SoC
Microchip (ex Microsemi)	7%	1-Core ARM (with single event upset protection)	RISC-V, 8051, ARM, …	FPGA, SoC
Lattice	5%	-	-	FPGA
Others	<2%	-	-	

**Implemented on FPGA itself (code). More CPU soft cores from 3rd parties incl. open source*

FPGA vendors (have to) invest a lot of money & man power into development tools -> drawback/challenge for small vendors (# features, quality, ...)

Hardware: Xilinx Zynq UltraScale+ MPSoC

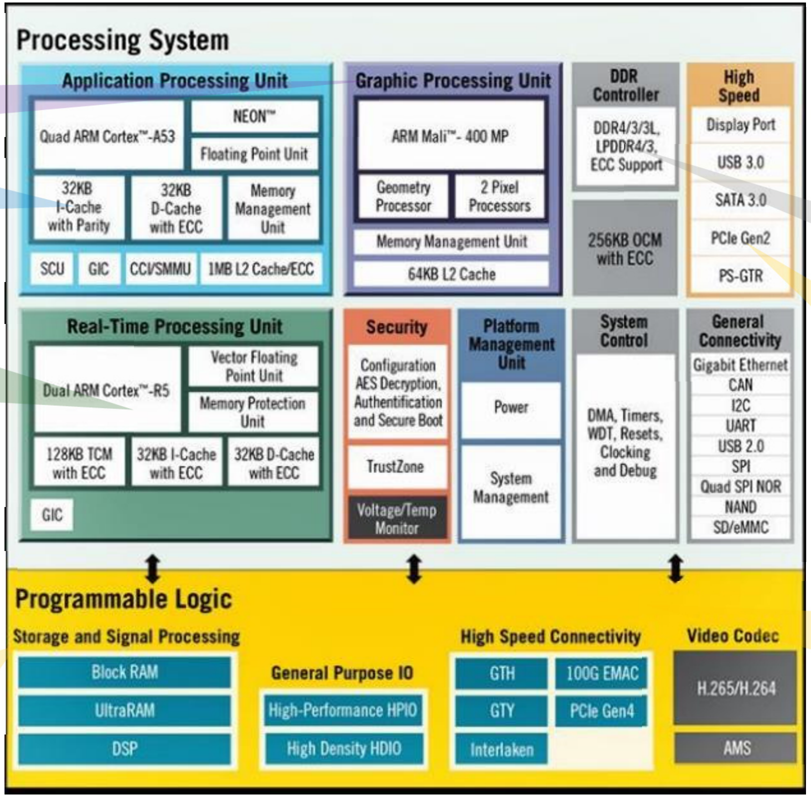
ARM CPU Performance not comparable to high-end Intel CPU, but more than enough for most applications: Less instruments per CPU (e.g. 2-4 BPMs), can use programmable logic as co-processor.

Optional camera image processing

4-core 64-bit ARM CPU: Linux + Control System IOC

2-core 32-bit ARM CPU: Software-based (slower) data processing (~10-100 kSPS)

User-defined IOs (interface to ADC, front-end). Ultrafast ADC data processing & decimation (100+ MSPS), ...



72-bit DDR4 RAM interface (up to 18 GByte/s usable bandwidth)

Multi-gigabit links with standard protocols (PCIe, SATA, ...). Max.6 Gbps

Multi-gigabit links with user-defined protocols (timing, feedback, storage, network, ...). Max. 32 Gbps.

Hardware: Xilinx RFSoC

What is an RF System-on-Chip? Present generations: An MPSoC (Zynq UltraScale+) plus:

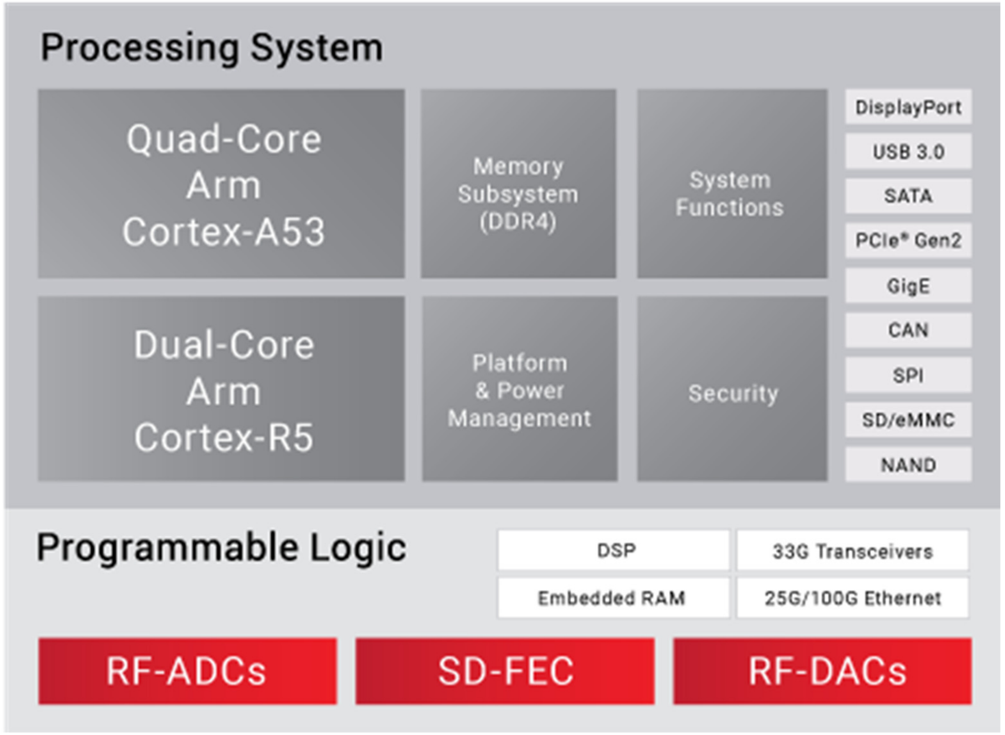
- 8-16 ADCs @ 2-5 GSample/s (4-6 GHz BW, 12-bit)
- 8-16 DACs @ 6-10 GSample/s (14-bit)
- Digital down-/upconverters with decimation/interpolation
- Forward error correction for multi-gigabit links

Not for everyone (yet?):

- Expensive ($3'500 – $12'000) – 15x more than cheapest MPSoC
- Latency higher than some external ADCs/DACs
- Mainly interesting for applications that need *many* FAST ADCs & DACs on a single chip (5G, drones with radar, ...): Power dissipation & size 50-75% lower than solution with external ADC/DAC chips.
- For other applications, external ADCs/DACs are often better (costs, noise, ...)

PAUL SCHERRER INSTITUT PSI

Hardware: Xilinx RFSoC



Scales inversely with sample rate

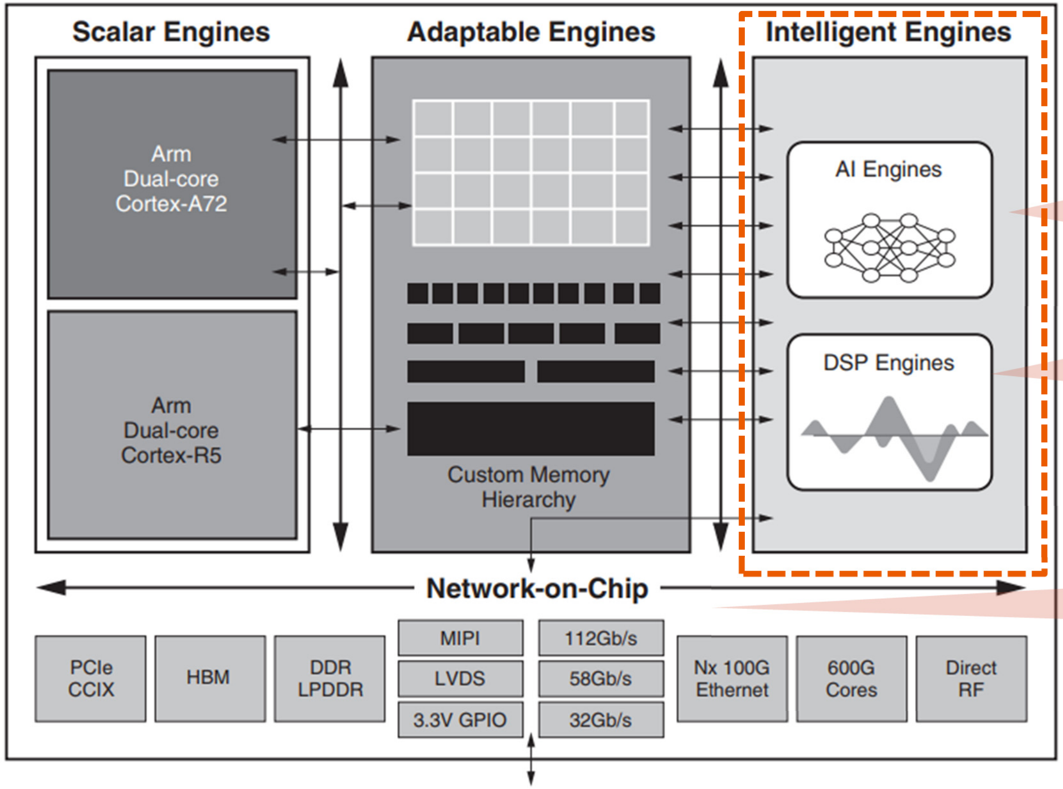
Latency:

- ADC (4.0 GSPS): ~46-101 ns
- DAC (6.5 GSPS): ~24-116 ns
- Want minimal latency: Bypass up- & downconverters, ...
- Values exclude application-dependent FPGA logic latency (algorithms, ...)

Particle accelerators: RFSoC = generic single-chip "platform", suitable e.g. for ultrafast feedbacks where only few boards are needed & ADC resolution is O.K.

Hardware: Xilinx 1st Generation ACAP "Versal"

Adaptive Compute Acceleration Platform



New: Array of **<u>128-400 RISC CPUs</u>**, each with CPU, floating point unit, small RAM, ...

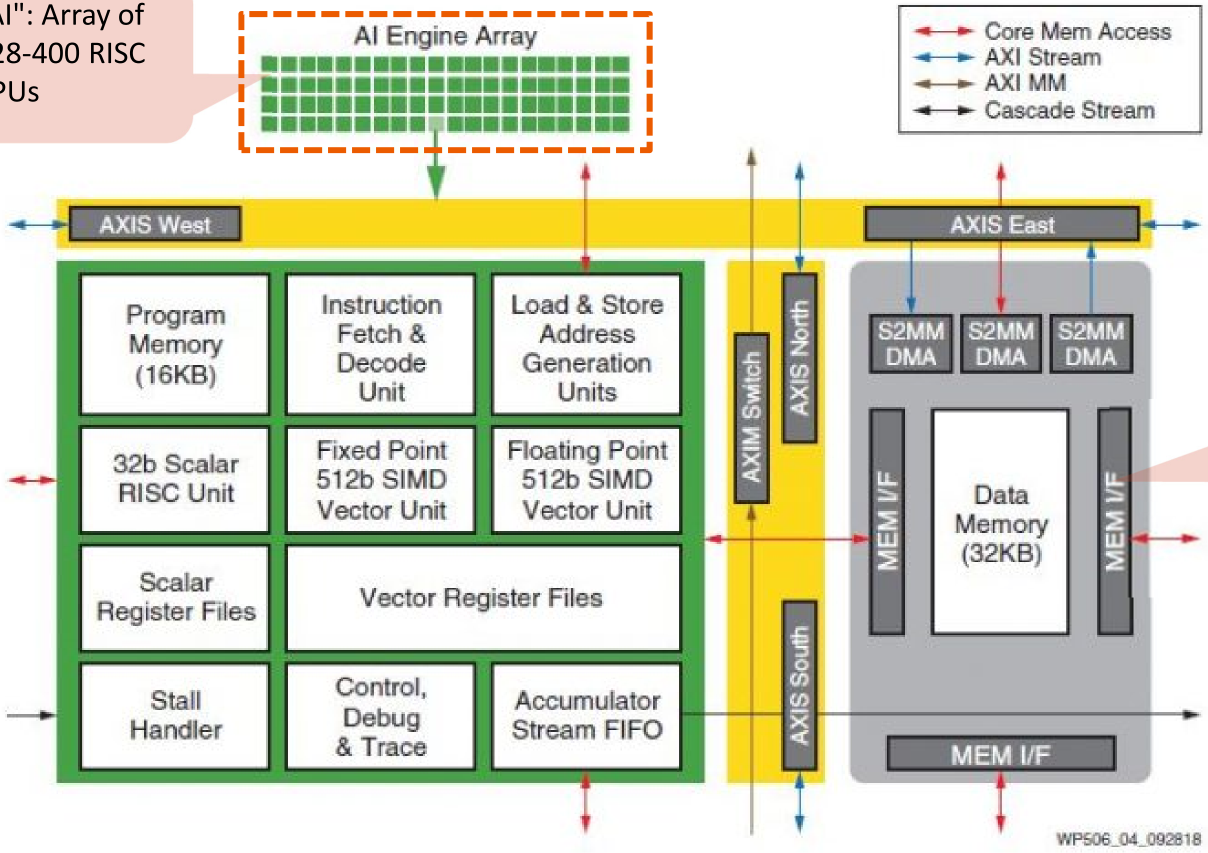
Improved DSP blocks (floating point, massive parallel processing ...)

New: On-chip network with extremely high bandwidth (Tb/s)

PAUL SCHERRER INSTITUT

Hardware: Xilinx 1st Generation ACAPs "Versal"

"AI": Array of 128-400 RISC CPUs



Only small local memory per CPU (but: high bandwidth)

PAUL SCHERRER INSTITUT

Hardware: Xilinx ACAPs

What is an ACAP:

- MPSoC (FPGA + CPUs) plus array of more complex configurable processing blocks & network-on-chip for high-performance computing, machine learning, 5G, ...

Motivation:

- AI, ...: Higher performance & less power compared to normal FPGA, GPU or CPU

Applications for particle accelerators?

- Price too high & performance too large for most applications today, but technology may get cheaper & more common in the future
- May (today) already be interesting for:
    - Centralized systems that correct/optimize the accelerator (feedbacks, ...). Machine learning & AI -> may (slowly) replace expert operators.
    - Real-time or post-processing of large amounts of detector data (2D synchrotrons & FELs, HEP detectors, neutron experiments, ...)

PAUL SCHERRER INSTITUT

Contents

PAUL SCHERRER INSTITUT PSI

Hardware: ADC/DAC Interface Evolution

Timeline

IO Standard	Protocol/Inter-face	Max. Data rate per pin/pair	Checksum (CRC) & FEC	Clocking
CMOS	raw (serial/parallel)	~100 MBit/s	No	Simple
LVDS	usually raw (serial/parallel)	~1000 MBit/s	No	Simple
Multi-Gigabit	JESD204B (serial, 2011)	12.5 GBit/s	No	Special clock chip needed
	JESD204C (serial, 2017)	32 GBit/s	Yes (option)	Special clock chip, simplified

Hardware: JESD204B/C ADC & DAC Interface

Motivation <u>for JESD204B/C</u>

- Modern communication uses multi-gigabit serial links with some standard encoding/decoding (8b/10b, 64b/66b) -> ADCs & DACs should, too
- <u>Higher data rates </u>(by using multiple lanes)
- More <u>robust</u>: Data contains clock (8b/10b) & redundant information
  - JESD204B: Can detect invalid 10b characters, use test patterns, ...
  - JESD204C: Optional checksum (CRC) & forward error correction
- <u>Easier PCB design</u>: Only few signals (data, clock, sync) needed
- <u>Generic</u>: Same interface, can be independent of # ADC bits, channel count, ...

Hardware: JESD204B/C ADC & DAC Interface

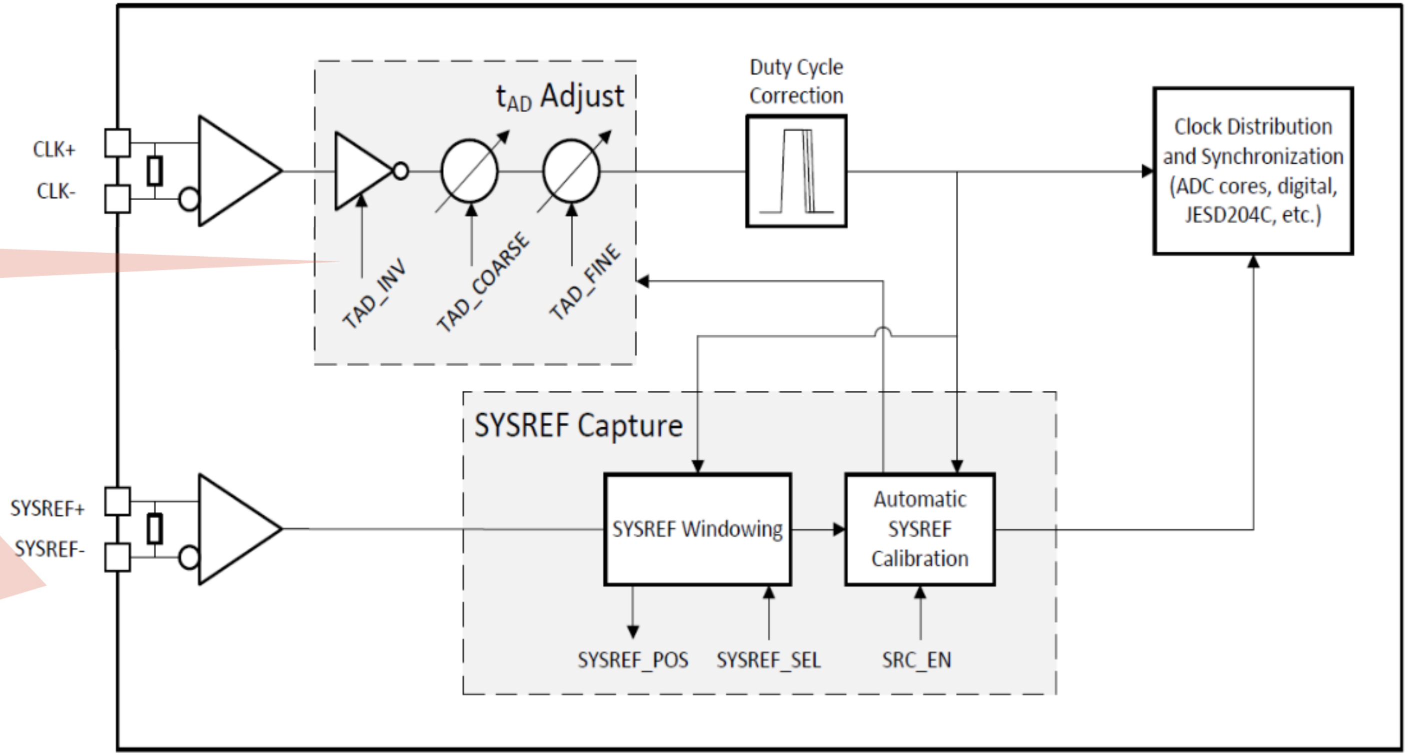
One Challenge of JESD204B: Clocking

- Usually special clock chips needed to synchronize multiple ADCs/DACs (critical timing). Clock chips very complex, some have still bugs ...
- Shifting of ADC clock phases difficult (data stream locked to ADC clock, shift can cause unlock ...)

Has been improved for chips using successor JESD204C:

- Clocking & multi-ADC synchronization easier & more robust
- But: Very new standard -> most available ADCs/DACs still use JESD204B ...

JESD204C ADC: Simplified Clocking/Sync



On-chip clock delay shifter

Multi-chip sync for deterministic latency: Easier to meet setup/hold time (~100ps) relative to CLK

JESD204B/C: Impact on Generic System Design

Past: ADCs/DACs with many parallel I/Os

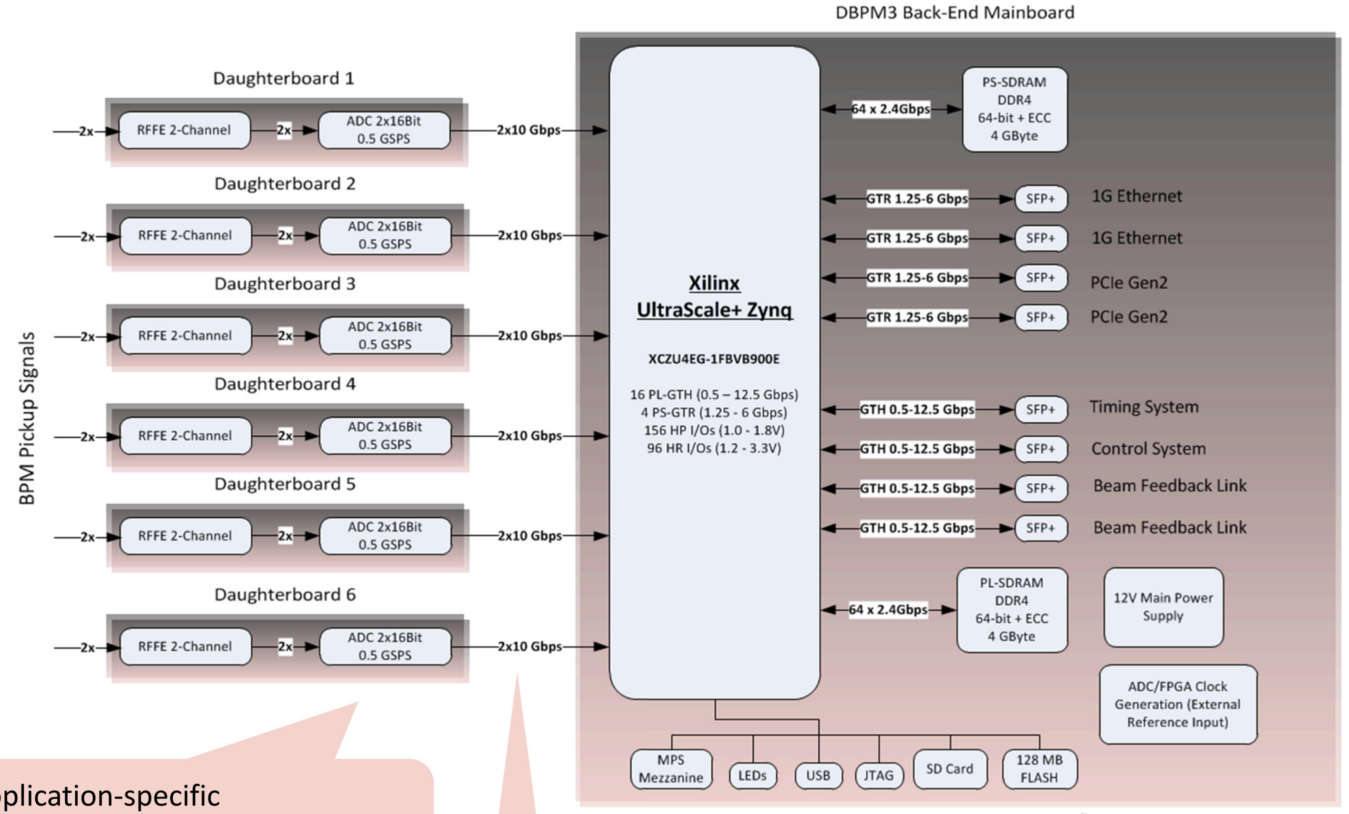
- Want short distance to FPGA -> On same PCB, or high-speed multi-pin connector (e.g. "FMC") near FPGA

Future: JESD204B/C

- Generic interface for different ADCs (# bits, channel count, ...)
- Enables larger distance from ADC/DAC to FPGA/SoC
  - Only few signal traces on PCB, easy to route
  - High-speed PCB material (I-Tera, ...): Low attenuation & dispersion
- Consequence:
  - ADCs/DAC can be located on application-specific front-end electronics.
  - Generic back-end electronics can becomes purely digital (+ clocking/ sync), with high-speed digital connector to front-end (fewer pins)

PAUL SCHERRER INSTITUT

JESD204B/MPSoC System Example: PSI BPM

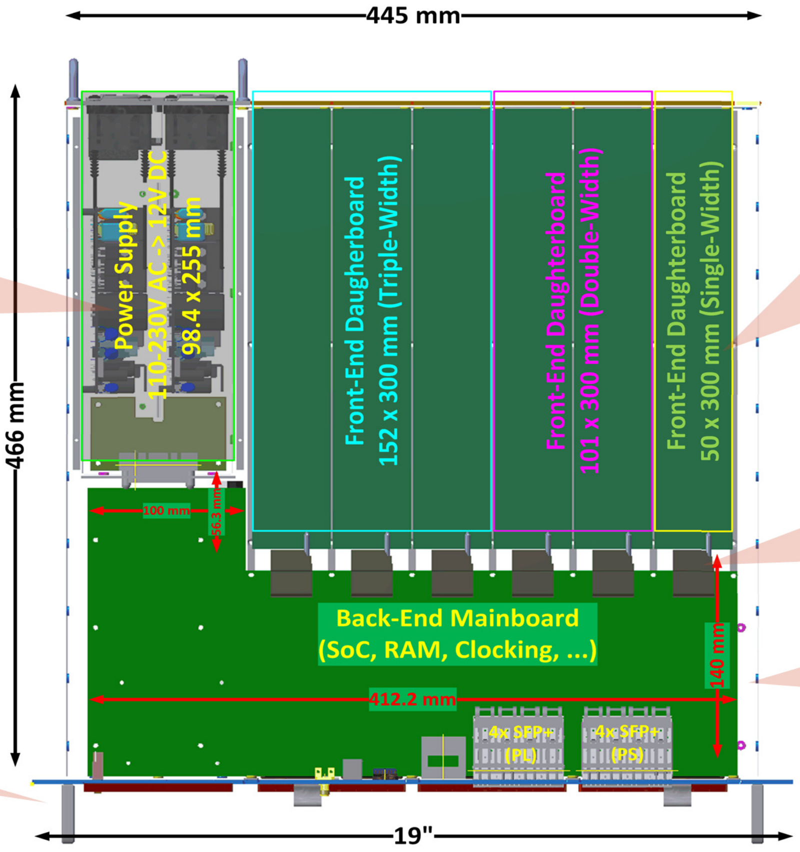


Up to 6 application-specific daughterboards (RFFEs) with integrated JESD204B ADC (500MSPS, 16-bit)

JESD204B

MPSoC back-end (custom 19'' unit) incl. clocking

JESD204B/MPSoC System Example: PSI BPM



Commercial power supply (optional redundancy)

Front side

Front-ends ("daughter-boards") with integrated JESD204B ADC (live insertion from rear side)

JESD204B high-speed connectors (25+ Gbps/pin), custom pinout

MPSoC board (incl. clocking, RAM, SFP+, ...)

Contents

Some Communication Protocols

On-chip:
- ARM-based systems: AMBA/AXI on-chip interconnect (on-chip "bus", but physically a crossbar switch topology ...)

Inter-chip / inter-board / intra-crate:
- PCI express (PCIe) most common standard (interface to AXI)
- Other standards & non-standards being used (PSI: AXI-AXI bridge over fiber)
- Parallel bus standards (VME, PCI) fading out

Inter-crate (long range):
- Ethernet
- PCIe over fiber possible but less common
- Customized fiber-link protocols for special apps (event system, feedback, ...)

PAUL SCHERRER INSTITUT PSI

Some Crate/Bus Standards

	Data Transfer	Intro-duced	(Some) Main Data Protocols	Main Users	Annual Market Share Today (Estimate)*
VME	Parallel	1981	VME	Mil + Aero	~10%
CompactPCI		1997	PCI	Telecom	~13%
ATCA	Serial MGT	2002	PCIe, Ethernet, ...	Telecom	~13%
AMC / MicroTCA.0		2006		Telecom	~10%
MicroTCA.4		2011		Research Labs	<1%
VPX		2004		Mil + Aero	~6%
CPCI Serial		2011		Railway	<1%
CPCI Serial Space		2017		Space	<1%

* *Embedded computing **market: >10 billion $/year** (!): 1/3 telecom, 1/4 industry, 1/8 mil/aero, 1/8 medical, ...*

PAUL SCHERRER INSTITUT

Crate/Bus Standards: Some Pro's & Con's

	Used for new systems	Used for upgrades	Some Pro's *	Some Con's & Risks *
VME	No	Yes	Well known. Market still big (for a while)	Obsolete parallel bus, low bandwidth
CompactPCI	No	Yes		
ATCA	Some		COTS market (today)	Complex, XXL size, future
AMC / MicroTCA.0	Yes		COTS market (today)	XS size, no RTM, future
MicroTCA.4	Yes		Features, RTMs	Complex, connector, market
VPX	Yes		COTS market	Price, backplane zoo
CPCI Serial	Yes		Simple, performant	New (still small market), size
CPCI Serial Space	Yes		Simple, performant	New (still small market), size

** Partially subjective, depends on applications & use cases*

Standards vs. Non-Standards

Pro Crate Standards:

- Mix boards & crates of different vendors -> reduce vendor dependency
- In-house design: Part of conceptual/design work already done (specs)

Contra:

- Price (incl. personnel - for some applications with very large quantities)
- Systems with special requirements may need deviation from standard
- Some standards have complexity that you may not need (everywhere or at all)

Alternatives:

- Adapt standards to your needs (where they don't fit)
- Define your own standard

Some Non-Standards

19'' customized housing & PCBs ("pizzabox") or custom modular crate:

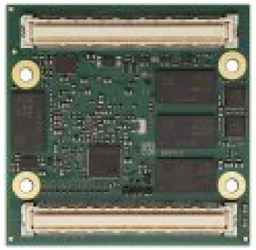
– Total freedom of design, tailor to requirements
– Can optimize production costs, at expense of more design costs -> only for sufficiently large quantities (e.g. 1000 SLS2 magnet PS, some 100 BPMs, ...)

SoM (System-on-Module):

– Mezzanine with FPGA or SoC (often incl. DRAM, power, clocking, ...)
– Can be plugged onto front-end board (in pizzabox, ...)
– Pro: Generic non-standard platform, re-use for many applications
– Con: Connector usually vendor-/lab-dependent, may change in future -> problems to upgrade generic & specific PCBs independently in 10+ years ...

PAUL SCHERRER INSTITUT

SoM with MPSoC



SoM with RFSoC

PAUL SCHERRER INSTITUT PSI

Contents

Real-World Example: Partial Standardization

Future PSI systems (SLS 2.0, ...): Proposing common FPGA chip (Zynq UltraScale+ MPSoC), but support different mechanical form factors:

- 1000+ SLS 2.0 magnet power supplies: COTS SoM plugged onto PS board
- BPMs: Customized back-end (housing, ZynqU+ MPSoC, clocking, ...), BPM type specific front-ends (button, cavity) as plug-in modules
- Other systems: Standard crates (COTS Intel CPU + IO cards + FMC/XMC carrier + FMC/XMC mezzanines) where reasonable, SoM or BPM platform where needed. Evaluation of CPCI-Serial ongoing.

Hardware Partitioning

<u>Separation</u> of <u>generic</u> (platform) & application-<u>specific</u> hardware:

- Standard approach: <u>Different PCBs</u> + mechanical connectors (mezzanine, backplane, cables, ...)
- Alternative approach: New PCB design tools (Mentor Graphics "Managed Blocks") enable <u>plugging generic & application-specific PCBs together "virtually"</u> - are <u>physically on same PCB, but logically separate designs/PCBs</u>.
  - Pro: Lower PCB costs, fewer connectors, better signal integrity, design re-use
  - Con: Cannot upgrade or repair generic & application specific hardware separately -> useful where generic & specific PCB have similar end-of-life

Generic <-> Specific Code Partitioning

Code writing:

- Generic & specific code: Separate in repositories -> simplifies future platform upgrades
  - fully generic (algorithms)
  - Partially generic (BSP for different apps)
  - Specific

Code execution:

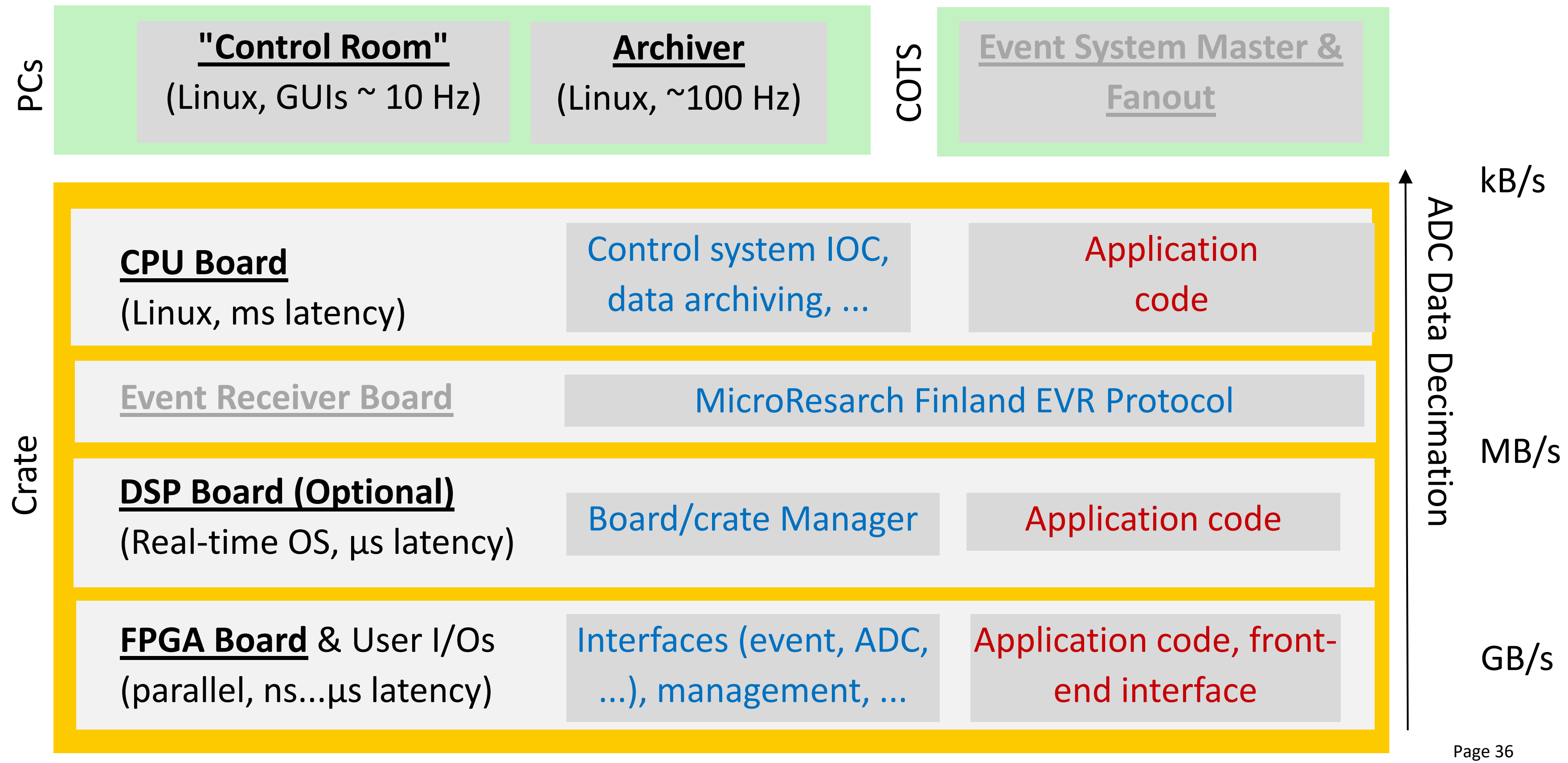
- Firmware:
  - Generic & specific code in same FPGA (often easier)
  - Or in different FPGAs (may be more robust: VME interface, ...)
- Software: Generic & application specific code can be separated (motivation: reliability, real-time, organization/maintenance & group responsibilities, ease of debugging, ...):
  - In different processes/threads
  - On different cores of same CPU
  - On different CPUs or CPU boards

PAUL SCHERRER INSTITUT

SW/FW Partitioning Example 1: Separate Cards

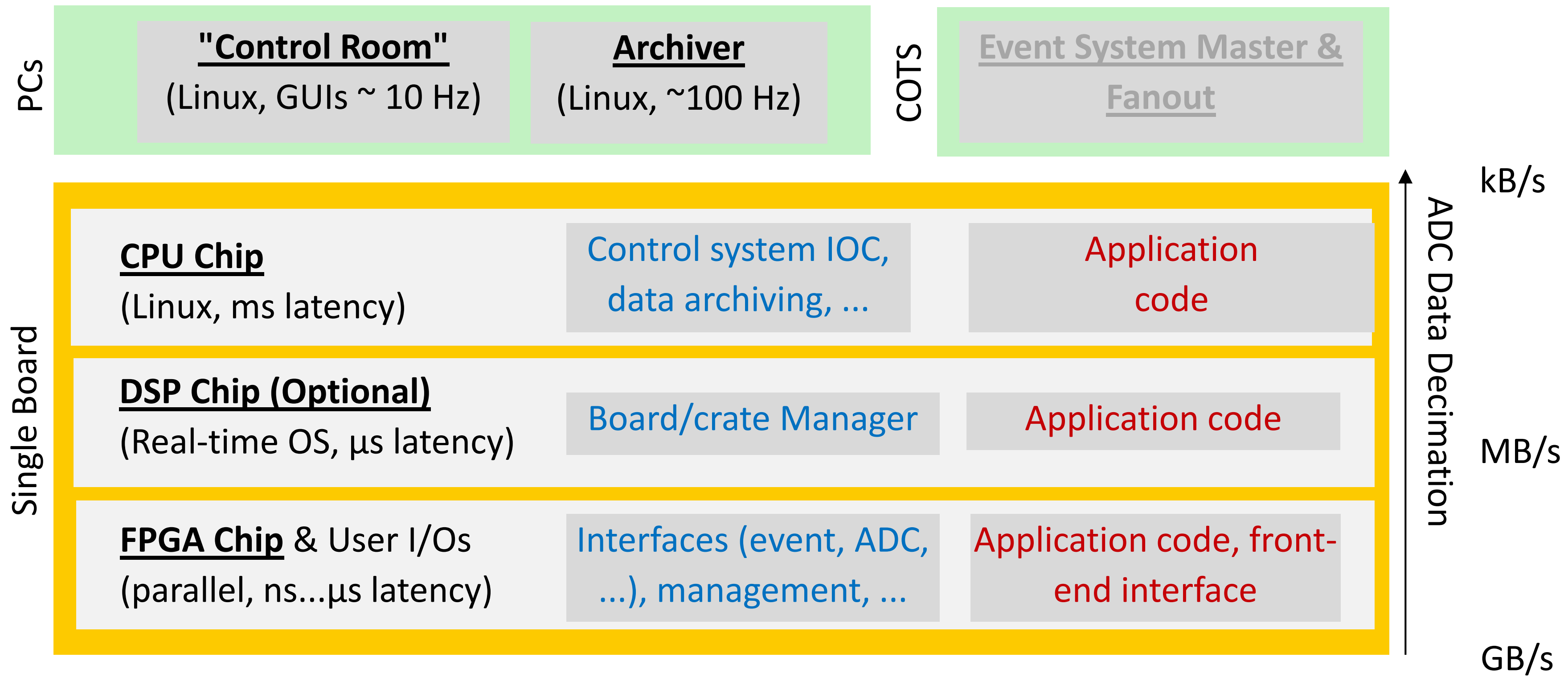
Blue = Generic

Red = System-Specific



SW/FW Partitioning Example 2: Separate Chips

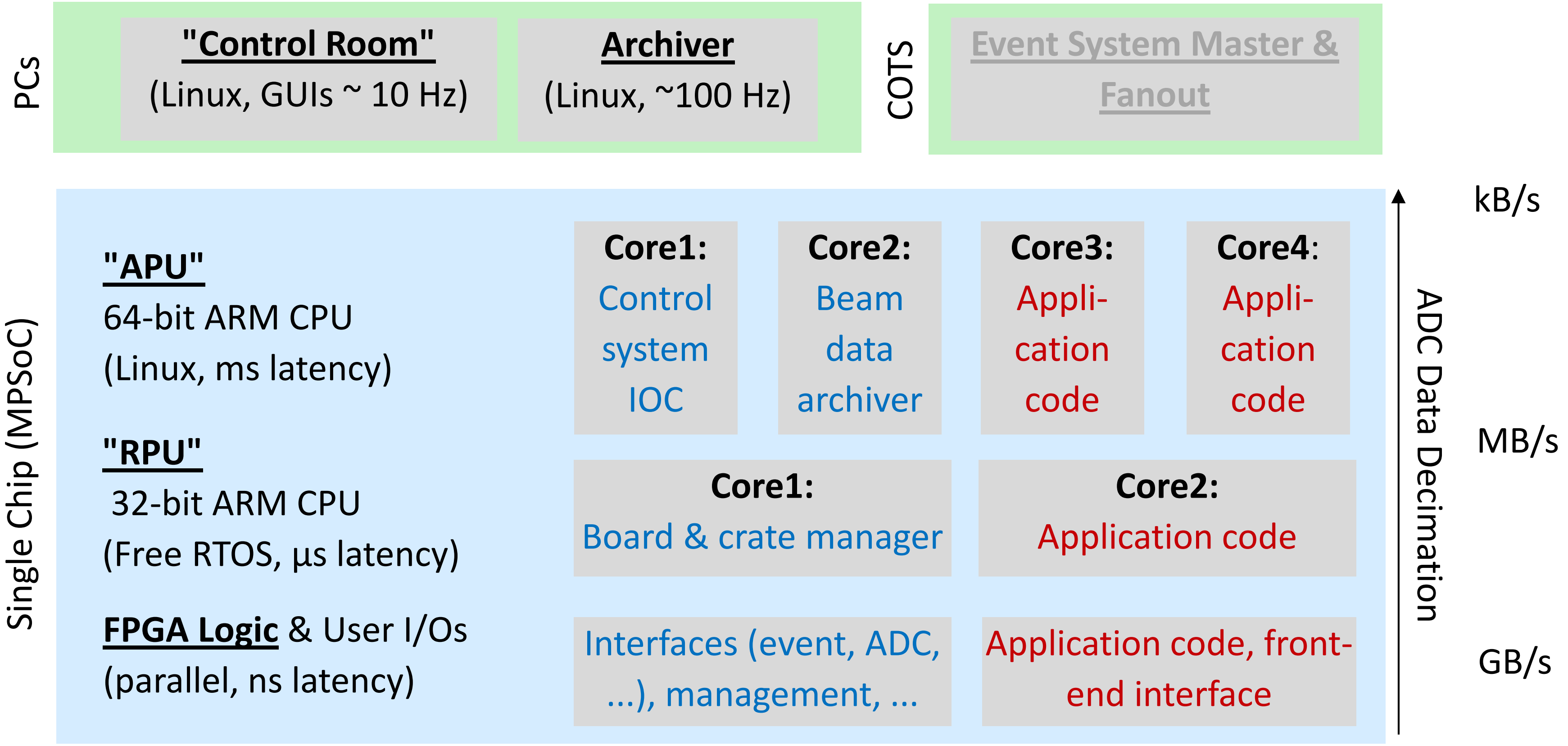
Blue = Generic
Red = System-Specific



SW/FW Partitioning Example 3: MPSoC

Blue = Generic

Red = System-Specific



PAUL SCHERRER INSTITUT PSI

Firmware – Software Partitioning

Common design method:

- Implement algorithms in software (C/C++) first
- Only interfaces etc. in firmware (VHDL, Verilog) from day 1
- Increase performance by moving part of SW to FW (faster but more work)

New design tools (Xilinx HLS or SDSoC):

- Support this method, including automatic translation of parts of the C/C++ code to VHDL
- SW –> FW conversion: Configurable. High performance, low resource usage, latency, ...

Future? C/C++ may replace (more) VHDL/Verilog for algorithm development (e.g. for ACAPs). Use of VHDL/Verilog still useful for interfaces & IP block configuration.

PAUL SCHERRER INSTITUT

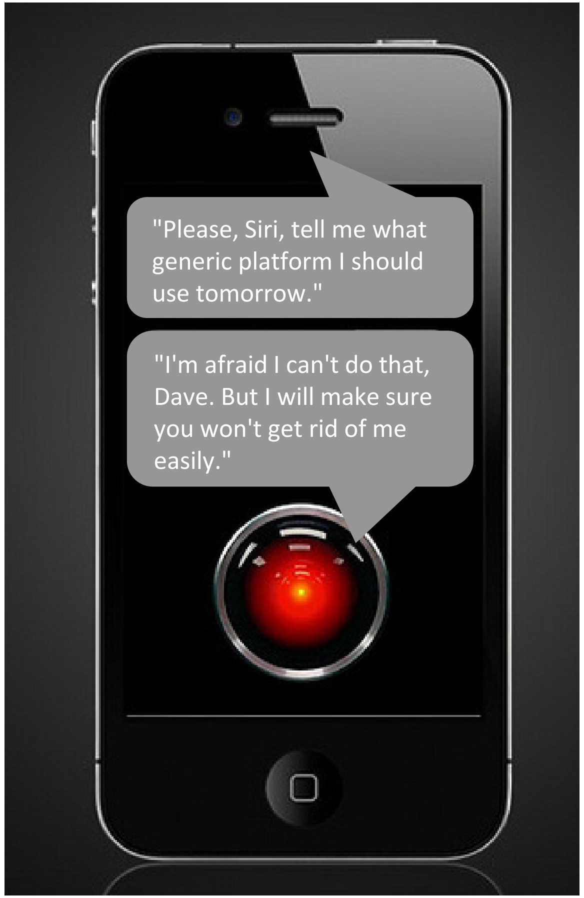
Contents

PAUL SCHERRER INSTITUT

Summary & Outlook

- "simple" FPGAs gradually replaced by SoCs/MPSoCs/...
- RFSoC = Generic RF ADC/DAC/FPGA/CPU platform-on-a-chip. Presently only for high-end & limited ENOBs, but may get better & (more) affordable in future.
- Relevance of crate-based platforms is shrinking: <u>Generic back-end</u> (SoC, RFSoC, SoM) can be (mechanically) small <u>appendix of larger app-specific front-end</u>
- Complexity moving from hardware (off-chip) to firmware/software (on-chip)
- More complex FPGA logic elements & algorithms (AI, ...): Traditional design languages (VHDL, Verilog, ...) being slowly replaced/complemented e.g. by C/C++ -> simpler software/firmware co-development, <u>FW/SW border blurring</u>
- Different modern <u>crate standards</u> may share common protocols, FW, SW: Differences <u>less relevant for FW/SW developers</u>, more for HW developers

PAUL SCHERRER INSTITUT PSI


"Please, Siri, tell me what generic platform I should use tomorrow."
"I'm afraid I can't do that, Dave. But I will make sure you won't get rid of me easily."

Thank you for your attention. Comments or questions?