



Hewlett Packard
Enterprise

HPE Cray EX255a Telemetry

Improved Configurability and Performance



Sean Byland
May 09, 2024



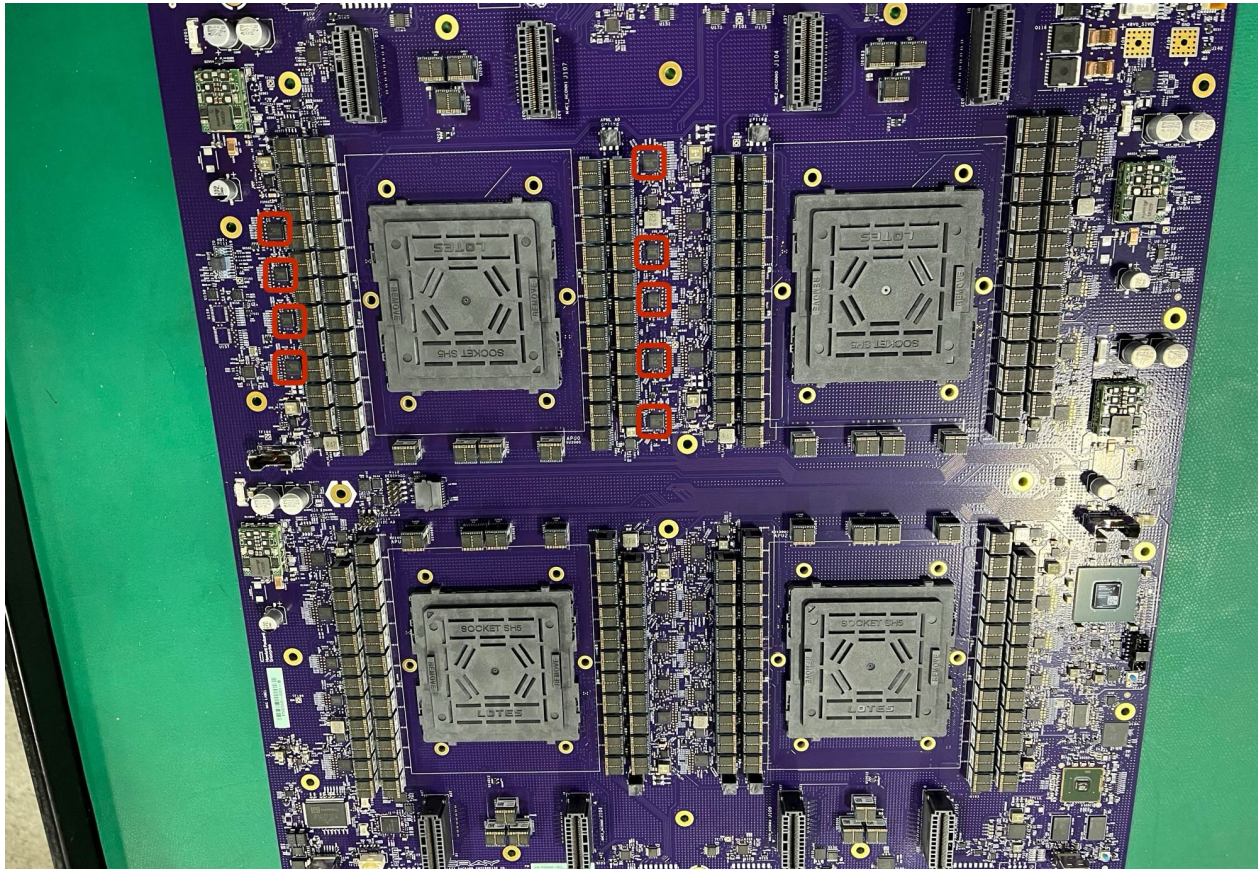
PERTH AUSTRALIA

CUG 2024 - © COPYRIGHT 2024 HEWLETT PACKARD ENTERPRISE

Agenda

- A short review, centered on the fundamentals of new telemetry software for HPE Cray EX255A blades
 - This is not a list of features, currently EX255A functionality mirrors other HPE Cray EX blades
 - “Telemetry” in this context primarily refers to environmental metrics (e.g. power, temperature)
- Why should you care
 - Improves observability and testability, for better quality
 - Decreases turnaround time
 - Enables future features
- Note: This architecture makes all available telemetry controls mutable in the back-end, but further API engineering work is required to change any specific values. At any point during the presentation, if you think a control should be made available as part of an API, please let me know. I’m here to learn what you all want and see what we can do.

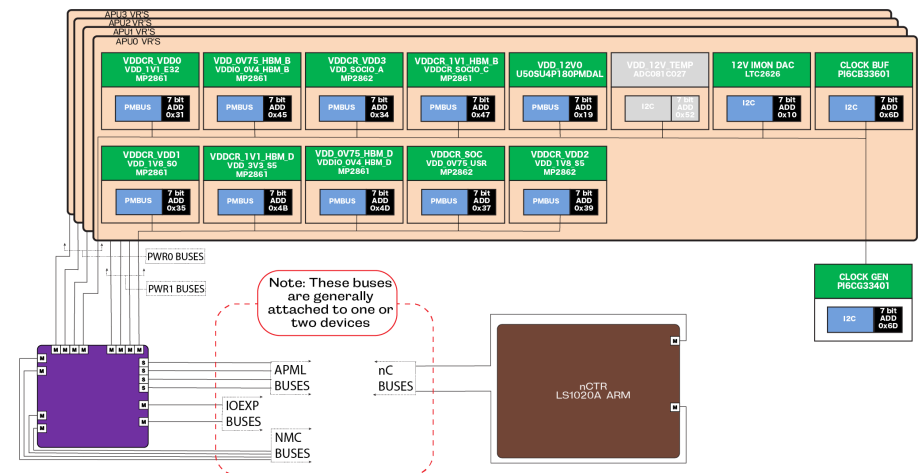
EX255a: More Power, More Voltage Regulation, More Telemetry



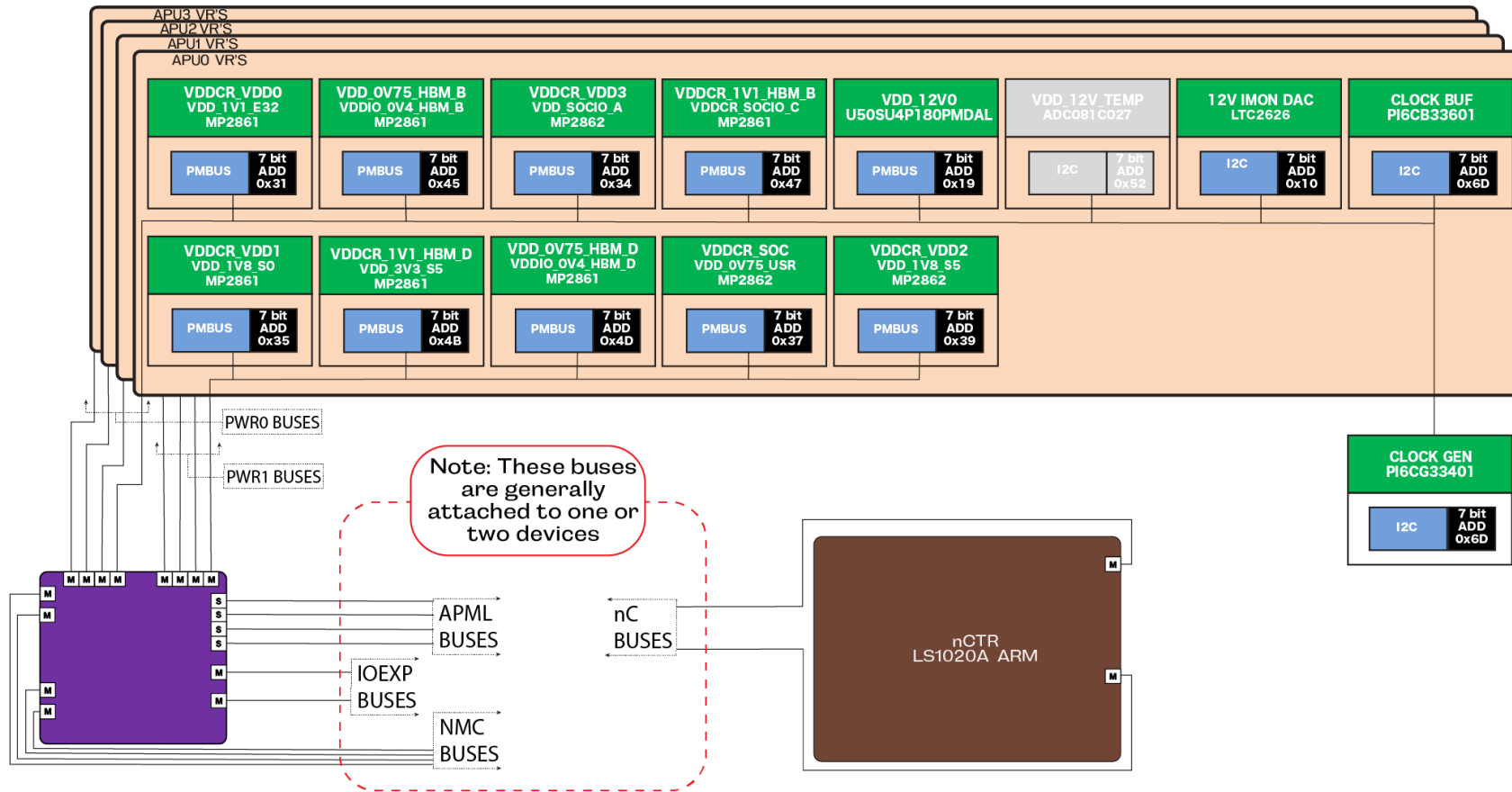
- 2 Node Cards / Blade
- 4 MI300a APUs / Node Card
- 10+ VRs / APU
 - 2 Power Rails / VR
- Up to 334 Commands / VR
- For a total of 900+ numeric sensors
 - HPE's EX235A (MI250) Blade has 222 sensors

nC Communication Block Diagram

- The Challenge:
 - MANY devices on each power bus
 - Requires refactoring and new code
- The Good:
 - Access methods are uniform
 - We prototyped i2c batching performance increases
 - Used on the 4-node CPU-based EX4252
 - Our FPGA supports “offloading”
 - Experience taught us what was needed and what wasn’t

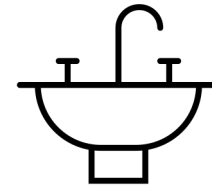


nC Communication Block Diagram



A New Architecture: The Goals

- Scan and publish data without lowering frequency
- Decrease CPU and memory utilization
- Enable fine-grained data collection and controls for:
 - Filtering for relevant data
 - Setting user-defined data rates
 - Easier site-specific configurations
- Nice-to-have's:
 - Simplify software flow
 - Reduce learning, debugging and hardware-bringup time
 - Improve cross-team collaboration
 - Improve testability and reliability
 - Enable (future) features
 - Provide Apple Pie



A New Architecture: The Plan

- Pick a good name
- Standardize hardware-facing data structures
- Place access, control, and descriptive data into a unified location
 - Embed defaults at compile-time
 - Enable run-time parsing from non-volatile memory
- Develop a device abstraction API
- Enhance and generalize rate controls



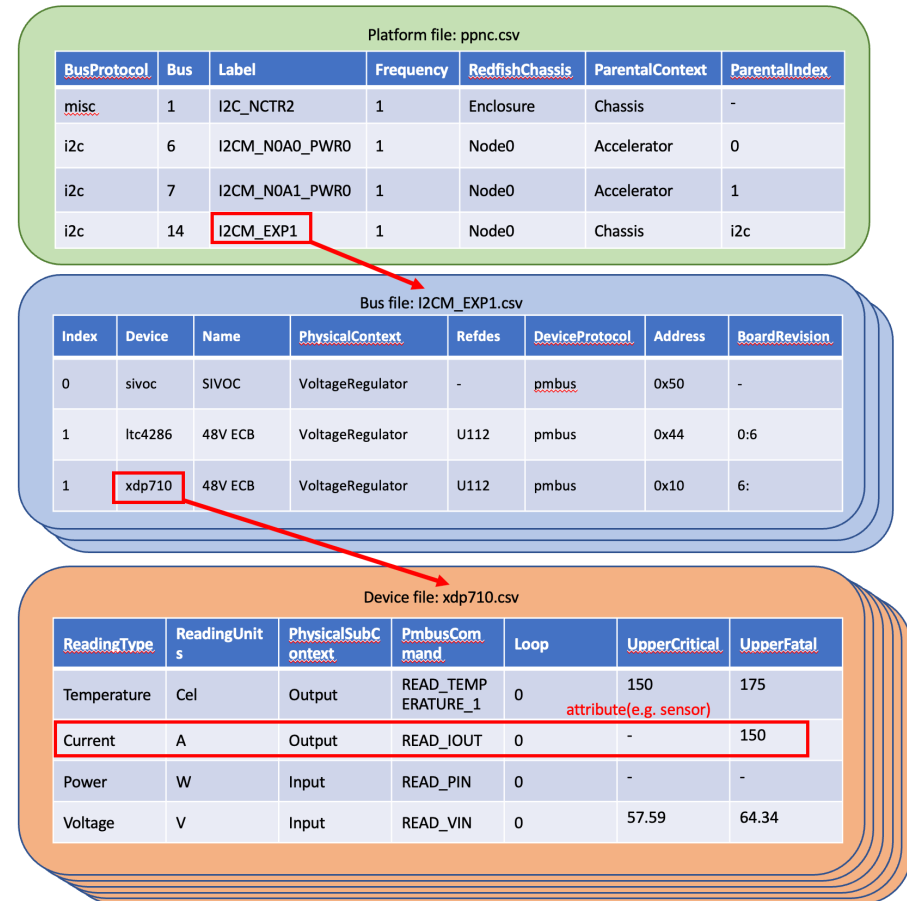
The Library: Device Interface Library (DIL)

- DIL is a static library:
 - Used in multiple applications
 - Includes general-use functionality and avoids dependencies
 - Enables single binary deployment and testing
 - Both test and production binaries are easily hosted at run-time
- We went with this name because:
 - It's descriptive
 - It's not overloaded
 - Works as a function prefix
 - When there are bugs it can be called a “dil pickle”



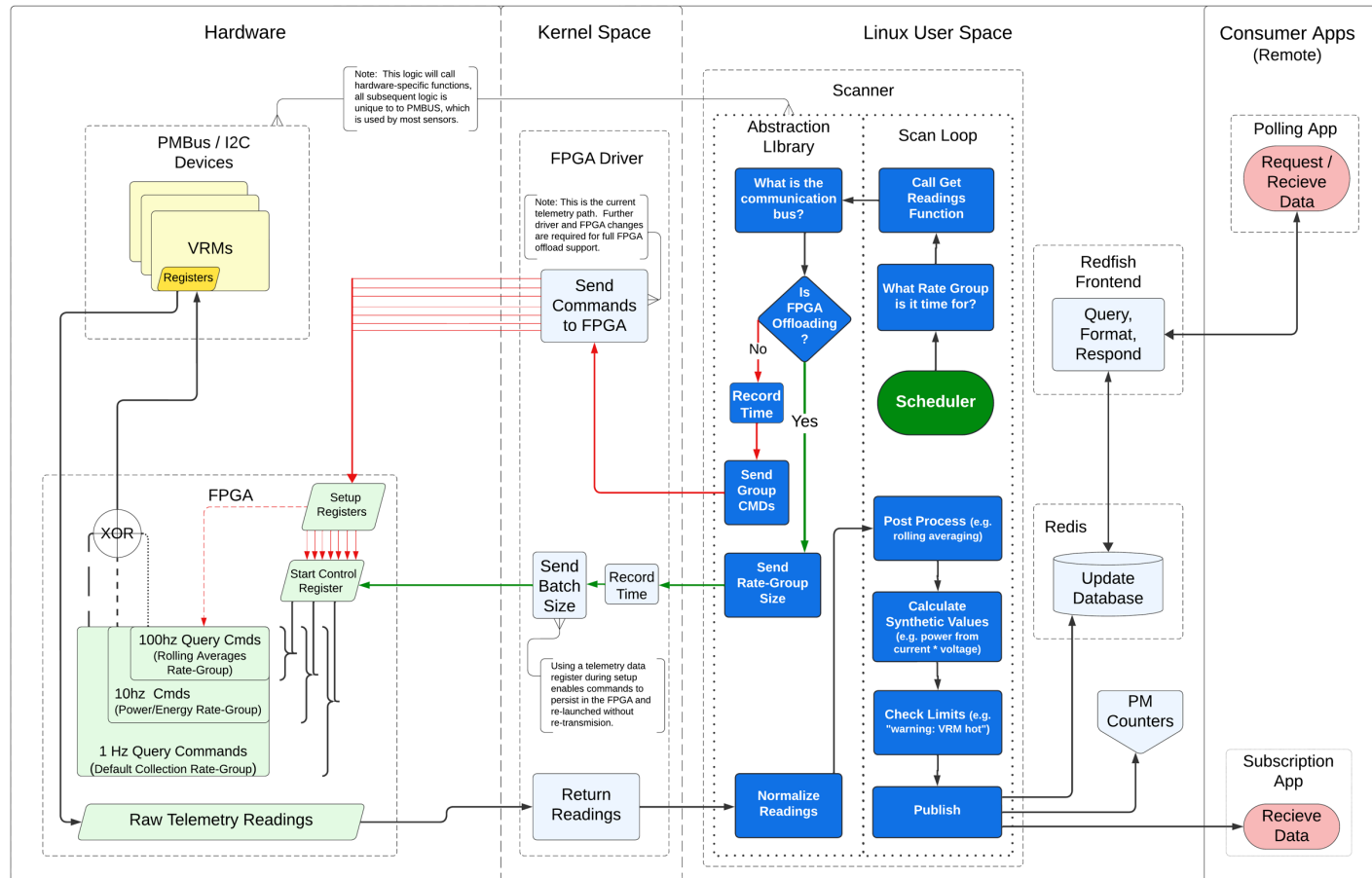
Device Interface Library (DIL) – File Format

- Uses CSV:
 - Encapsulates arrays well
 - Spreadsheets were already used for thresholds
 - Low-effort to parse via C or shell
- Uses a file bundle:
 - Enables device-specific fields and columns
 - Enables isolated updates



DIL – Runtime Architecture

Device Interface Library (DIL) Run-time Architecture



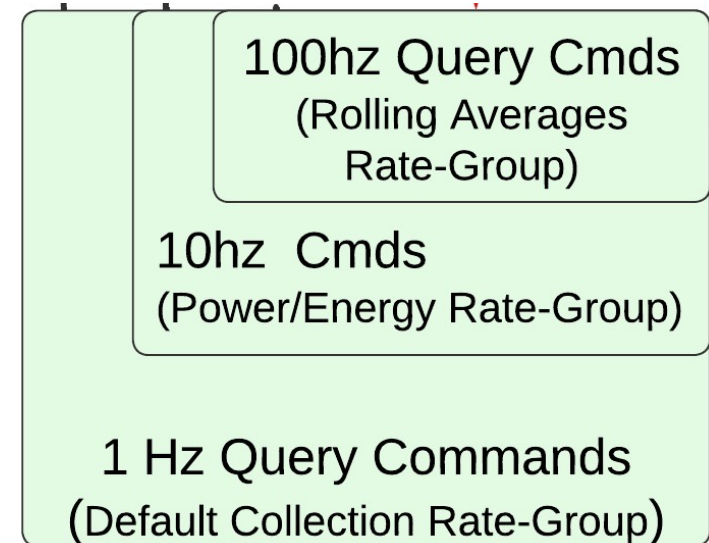
DIL – Rate Control

- “Frequency” is set at the system-level
- Bus frequency has configurable:
 - Scan rates: Set as a “ratio” of the system frequency
 - Rate groups: Set as “ratio” of bus rate
 - Publishing rates: Set as a “ratio” of the group rates



DIL – Rate Control

- Rate group rules:
 - Slow groups include fast groups
 - Groups are sorted by descending frequency
- Which enables
 - Performant partial scans
 - A single value per tick, per bus, to dictate what need scanning
 - Is pre-calculated to avoid expensive run-time computations
 - Is sent to bus-specific threads via a scheduler
 - Full FPGA offloading (of all rate groups)




Conclusion

- EX255a is streaming at the desired rates
- File-auditing by experts enabled early detection of issues
- Publishing control needs to be completed
- Batch data is sent every query. Data is organized for FPGA offloading, but further work is required
- All attributes, with front-end development, could be changed and persisted
 - Current interfaces match other HPE blades
 - Enables, doesn't add new features
 - New functionality requires front-end and system management software support
 - User feedback should drive new functionality



Thank you – Questions?



User feedback would be appreciated

sean.byland@hpe.com



Abstract

The new HPE Cray EX255a blade (two nodes, each with 4 AMD MI300a sockets) has power demands and additional sensors that require a more robust power subsystem and data collection capability.

This necessitated careful evaluation and changes to how we manage, collect, and publish sensor data. We factored all sensor access parameters and operational characteristics for the EX225a node cards into a standardized file format. These files define default values for compilation. The files can be edited and marshalled into runtime accessible structures, enabling testing, tuning, and experimentation with alternative settings.

Starting at the hardware access layer and working up the stack we optimized the code paths to enable collection of more sensors in our fixed time budget. This work is the foundation for future work that could enable the ability for higher-level management and monitoring software to customize data collection on behalf of users.

