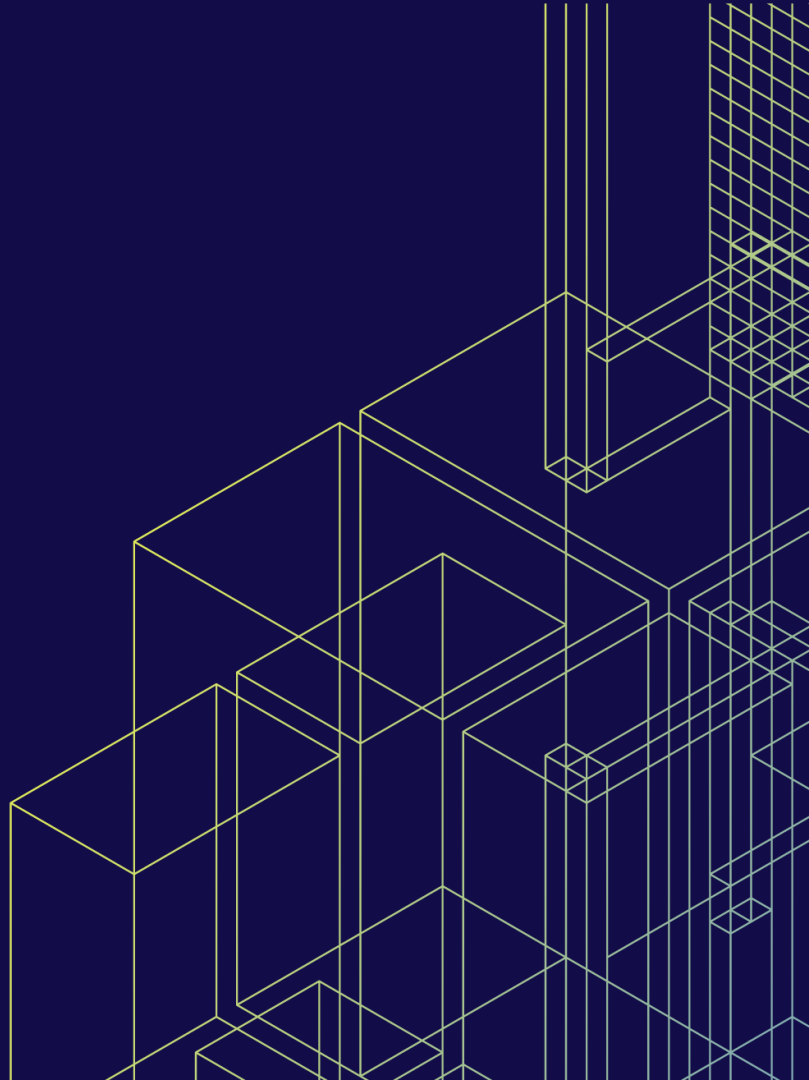
SDC 19

September 23-26, 2019
Santa Clara, CA

A Simple Approach to Implementing Computational Storage

Jason Molgaard
Storage Solutions Architect
Arm



Agenda

- Overview of Computational Storage
- Fixed and Programmable CSD Hardware
- Dataflow Comparison
- Communicating with a CSD
- Interfacing to a CSD
- Programmable CSD Software
- Current Status

What is Computational Storage?

The SNIA Technical Working Group (TWG) is currently defining – Get Involved!

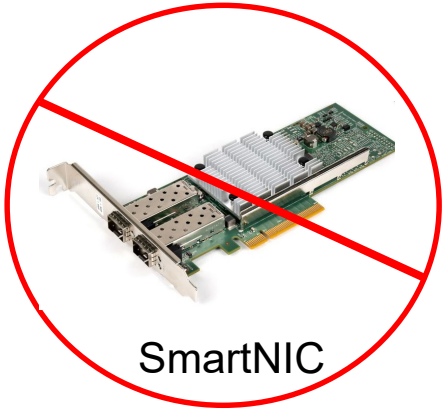
Some definitions:

- Computational storage: Computational storage services coupled to storage, offloading host processing or reducing data movement.
- Computational Storage Services (CSS): Services that perform computation on data where the service and data are associated with a storage device.

Compute coupled to storage to offload the host or reduce movement.

SNIA COMPUTATIONAL STORAGE

What is NOT Computational Storage?



- SmartNICs are coupled to the network; not the storage
  - Computational Storage has compute coupled to the storage
- No latency reduction with SmartNICs
  - Still need to package/unpackage data in fabric protocol

Additional Definitions

Additional Key Definitions:

- Fixed Purpose (FCSS): Fixed function that may be configured and used
- Programmable: (PCSS): CSS that can be programmed
- Computational Storage Device (CSx): Drive, Processor, or Array
- Computational Storage Drive (CSD): A drive with storage
- Computational Storage Processor (CSP): Compute for storage
- Computational Storage Array (CSA): Array of devices

SNIA COMPUTATIONAL STORAGE

arm

Why Computational Storage?

The volume of data we generate is expected to grow by 27% per year*

Drive capacities continue to increase – 16TB SSDs (and HDDs) available

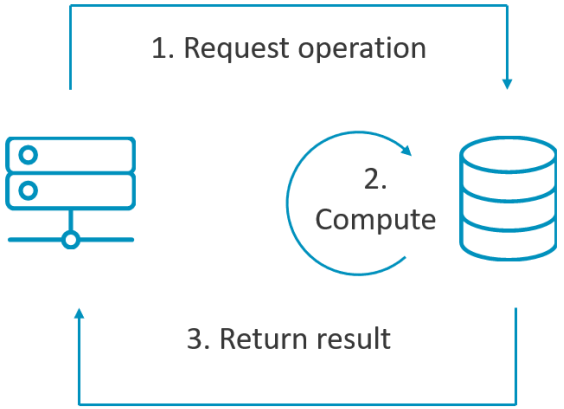
We need to find new ways to manage these volumes of data

*https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf

How Does Computational Storage Help?

- Compute happens on the drive
  - No moving data to the host
- Lower latency
  - Data not packaged in fabric protocol
  - No data movement across the interface
- Minimum bandwidth and power
  - Only results delivered to the host
- Data centric processing
  - Workloads specific to the data are deployed to the drive
- Security
  - Unencrypted data doesn't leave the drive

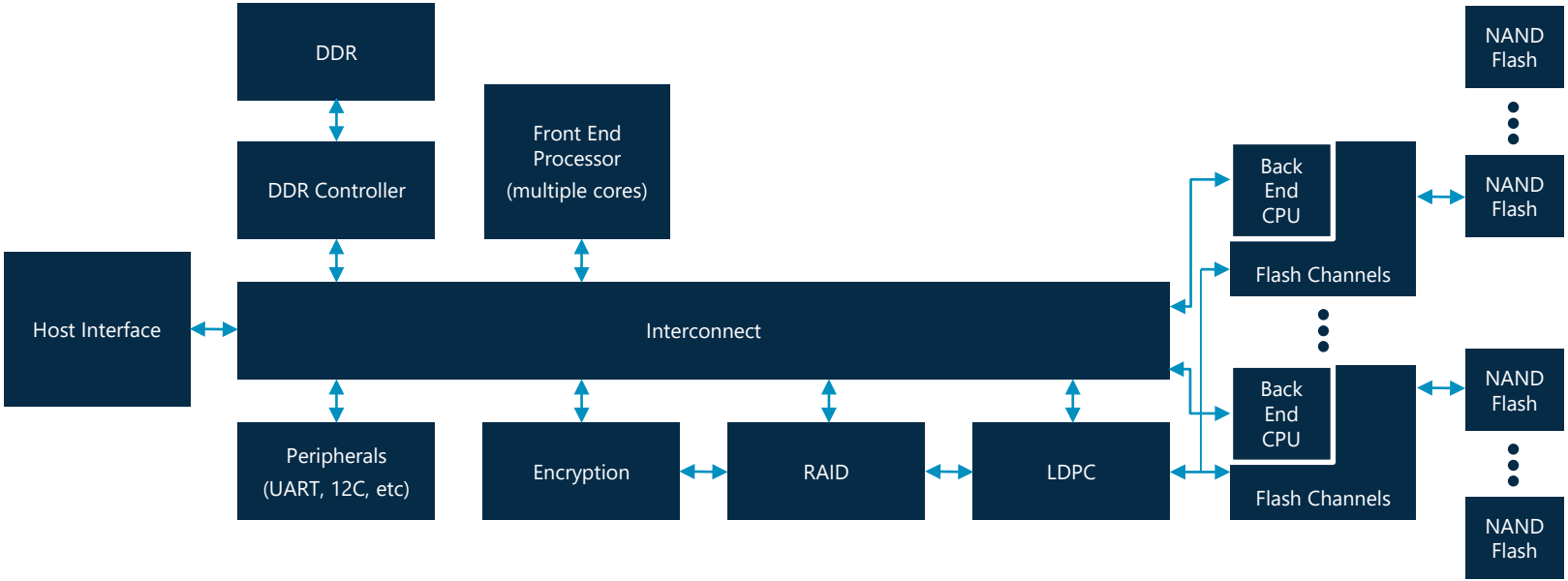
Simplified model with computational storage



SSD Controller Hardware

CSD maintains existing SSD controller architecture for data movement and control

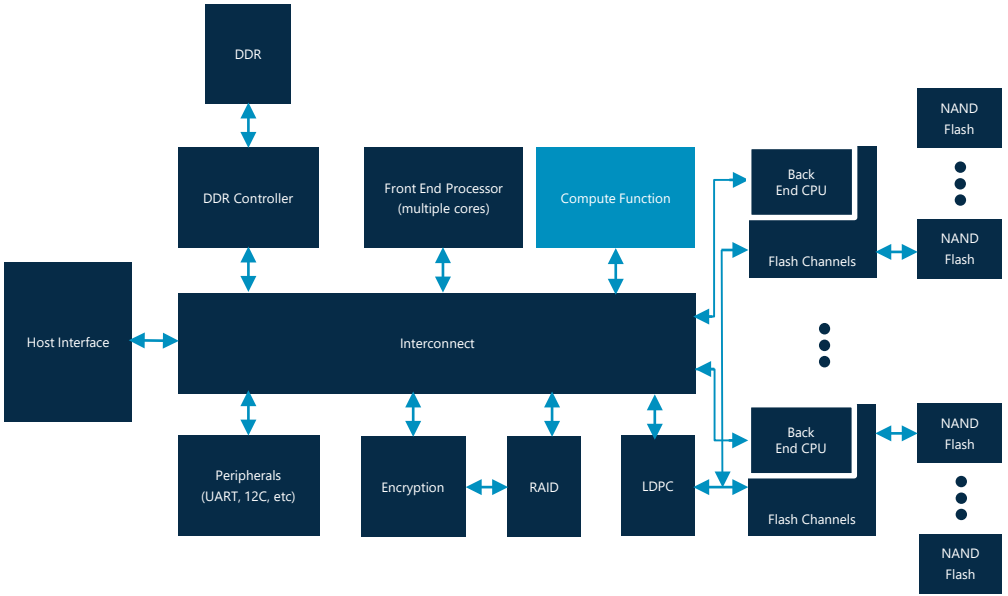
- Compute already present to manage FTL and control path



Fixed-Purpose CSD

Any function possible

- Fixed in the controller or after reset
- Could be implemented in an FPGA next to the controller or as part of the controller
  - Allows a new compute function to be downloaded
- Can be configurable with different parameters

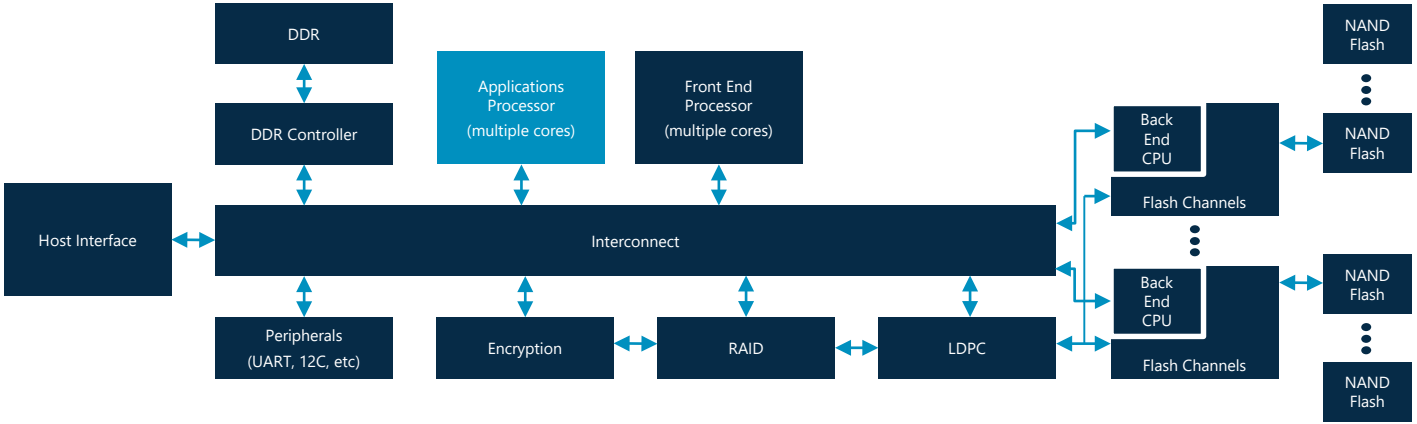


Programmable CSD

- Any function possible – liked fixed-purpose
  - Use your favorite programming language
- Can be downloaded dynamically
  - Download as needs arise
- Can be changed dynamically
  - Modify existing or create new workload on-the-fly

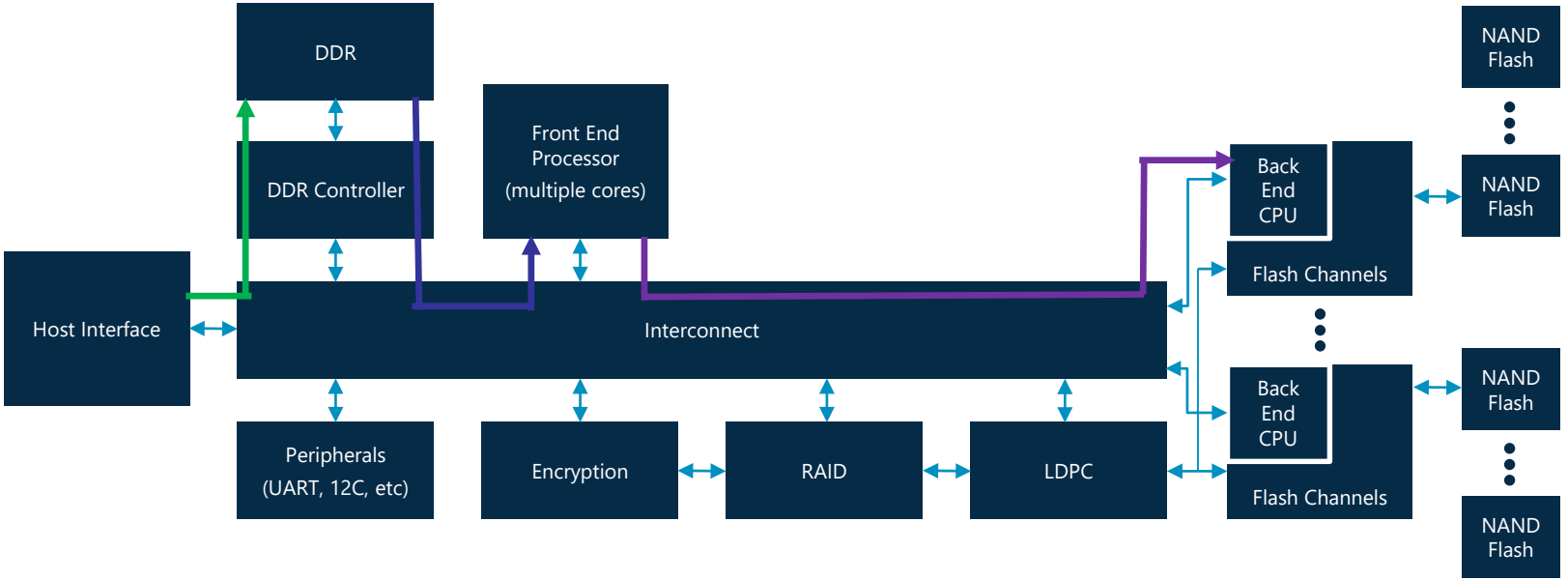
Programmable CSD Controller HW

- May need to add more low-cost compute
  - Generally easy to add
  - Processor cores optimized for running Linux and containers
  - Optimize the quantity and performance based upon workloads to be handled on the drive
- Modify the datapath to allow processors to access data



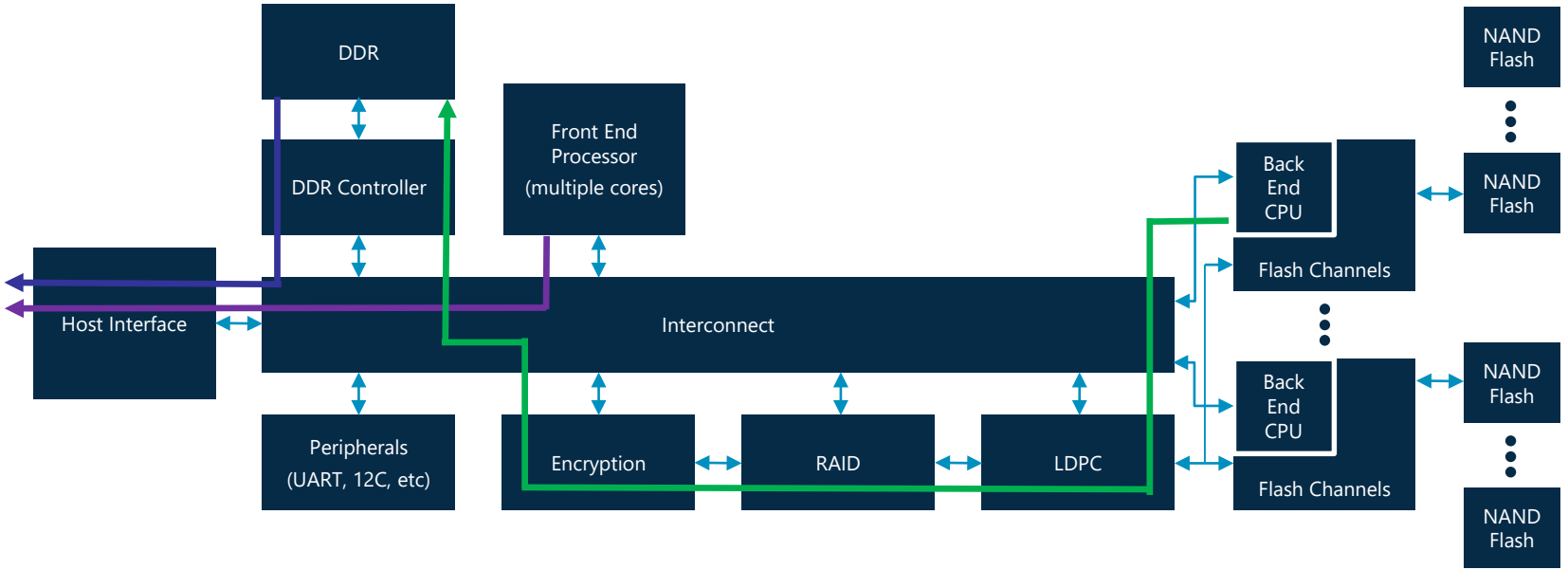
SSD Controller Dataflow (Read Request)

1) Read request arrives from host
2) Front-End processor analyzes request and builds transfer descriptor
3) Descriptor dispatched to appropriate flash channel



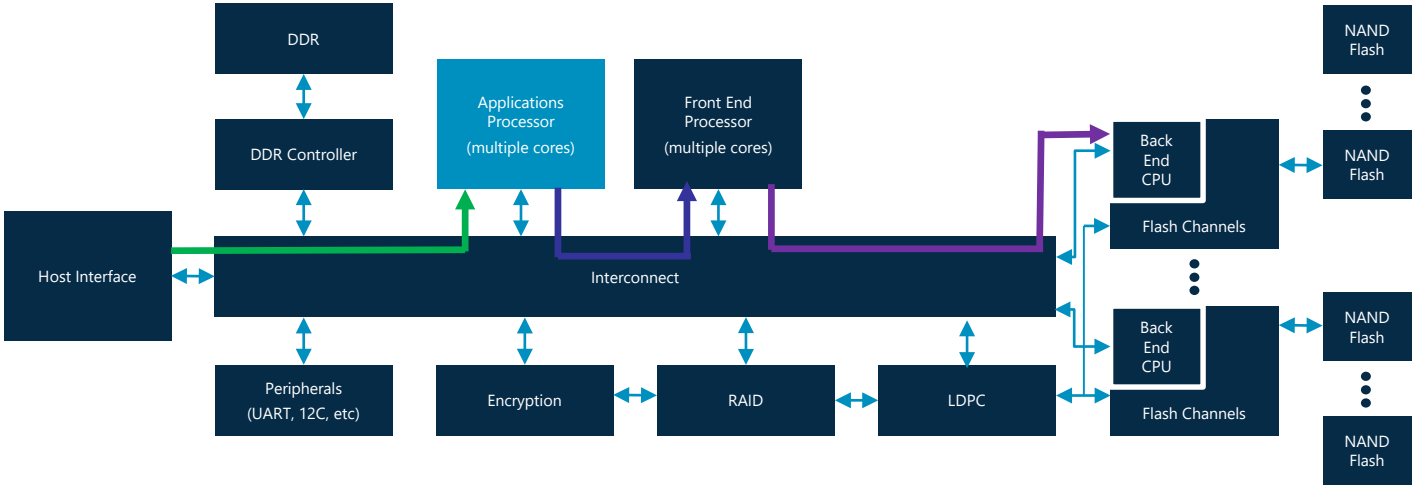
SSD Controller Dataflow (Read Data)

4) Read data returns from channel and passes through data path (ECC/Decryption) to DDR
5) Read data from DDR packaged in fabric protocol, and DMA'ed to host memory
6) Completion sent to host from front-end processor



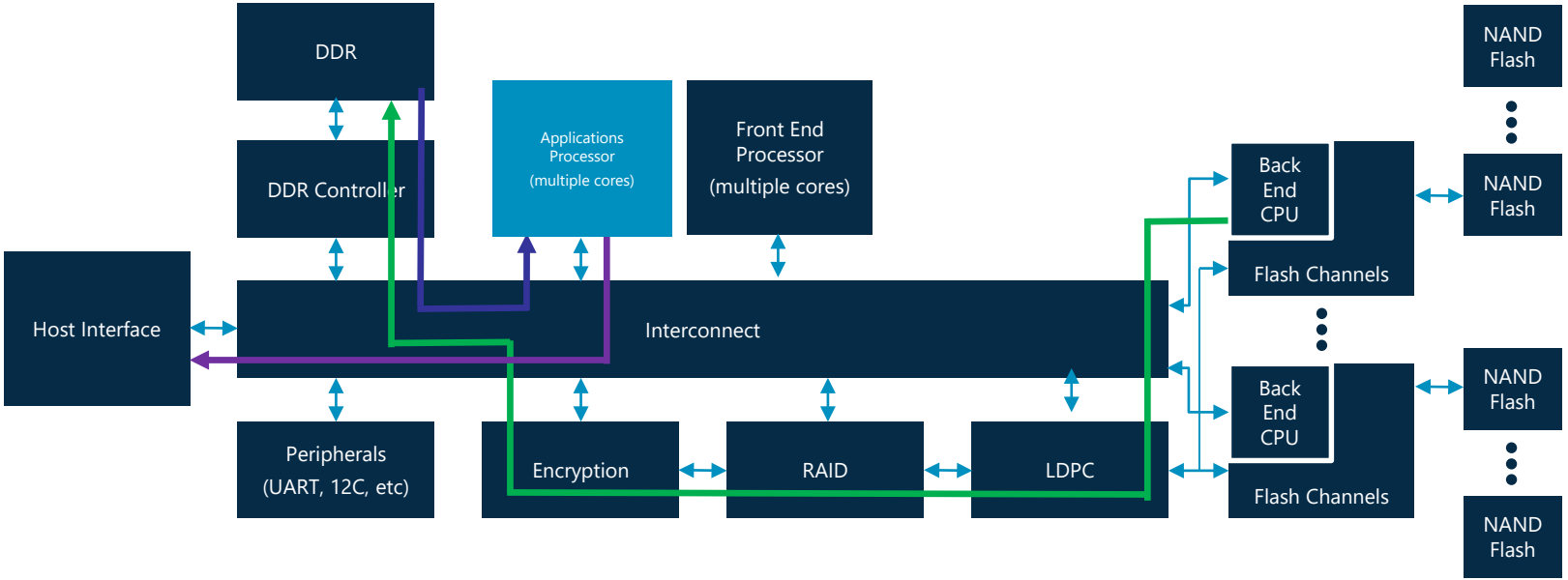
CSD Controller Dataflow (Request)

1) Compute request arrives from host
2) Applications processor notifies front-end processor to build transfer descriptor
3) Descriptor dispatched to appropriate flash channel



CSD Controller Dataflow (Compute)

4) Read data returns from channel and passes through data path (ECC/Decryption) to DDR
5) Compute performed on read data from DDR
6) Repeat steps 2-5 for all data
7) Result returned to host when compute finished



Communicating with a CSD

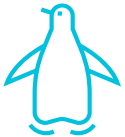
- Discovery – Discover the capabilities available on a drive
  - May involve changes to the fabric protocol – CS TWG to recommend
  - What is needed:
    - Identifiers, types (fixed/programmable), sub-types, state (ready, available, etc.), configuration descriptor to describe current configuration plus all valid options
- Configuration – Configure the compute services
  - May also involve new fabric protocol commands (CS TWG to recommend)
  - What is needed:
    - Method to write configuration data structure
    - Read/Check actual configuration accepted by the device (could differ from what was written)
- Usage – Use the compute service(s)
  - Could be transparent or direct
    - Transparent would simply happen during normal data transfers
    - Direct would require commands from the host to initiate and retrieve results
  - What is needed:
    - Unified method of interacting with CSSes

Interfacing to the CSD

- Data movement using standard NVMe, NVMe-oF, SAS, etc
  - The drive operates like normal
- Compute workloads:
  - May need new fabric commands (likely to-be-defined)
    - Includes discovery, configuration, and usage
  - Also need driver changes

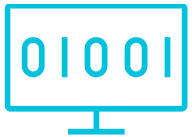
Programmable CSD Controller Software

What to run on the applications processor?



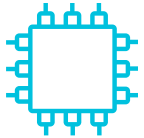
Linux

Could develop bare-metal code, but why not leverage the Linux ecosystem?



Container software (Docker, Kubernetes)

Develop on one machine and compile for another



Compute function(s)

The desired operation for the data on the drive

Linux

- Any distribution, including full distributions
  - e.g. Ubuntu, Debian
- Would want to strip down and make small

Linaro
www.linaro.org

Multi-Architecture Containers

- Allows software to be created on your development computer and executed on the processor in your CSD
  - Abstracts the underlying OS and hardware
  - Applications written and tested on the development computer
    - Can be tested on a cloud server in the native architecture
      - If there is an Arm core on the CSD, the application can be tested on a cloud Arm server
    - Shorter development time; Faster time-to-market

Application (Compute Function)

- Develop (cross-compile) and test with Container software
- Can be any function
  - Examples: Search, Thumbnail Generation

Current Status

- Computational Storage TWG in SNIA defining the standard
  - Still early in the definition
  - Developing theory of operations and examples
  - Much work ahead, particularly interfacing to the compute and downloading workloads

Join the effort!

SNIA COMPUTATIONAL STORAGE

Computational Storage White Paper

Download our free guide to computational storage on Arm at storage.arm.com

- The benefits computational storage brings to architects, developers and organizations
- How to move from a traditional storage solution to a more intelligent device
- Real-world examples from Arm's partners

Thank You!

Learn more and download the white paper:
storage.arm.com

Contact: jason.molgaard@arm.com

Visit: snia.org/computational

arm