Transforming Storage Controllers with Low-cost, Low-power Compute

Neil Werdmuller
Director of Storage Solutions
Agenda

• Quick overview of computational storage
• Computational storage drives and interfaces
• Programmable computational storage drives
• On-drive Linux
• Low-cost, low-power compute

Arm is a founding member of SNIA Computational Storage Technical Working Group (TWG)
Compute waits for data
- Takes time to move data across fabric
  - Processing stalled until data is available

Adds latency
- Multiple layers of interface and protocols
- Data copied many times
- Bottlenecks often exist

Consumes bandwidth/power
- Moving data is expensive
- Data copies increase system DRAM

Traditional model
1. Request data from storage
2. Move data to compute
3. Compute
4. Move results to storage
Computational Storage Drive (CSD)

Compute happens on the data
Data moved from flash to in-drive DRAM and processed

Lowest possible latency
No additional protocols – just flash to DRAM

Minimum bandwidth/power
Data remains on the drive – only results delivered

Data centric processing
Workloads specific to the computation deployed to the drive

Security
Unencrypted data does not leave the drive
CSD Types and Interfaces

Methods to control and manage a CSD
SNIA TWG developing standards enabling interoperability
SNIA TWG defining NVMe CSD control (advertising, use...)

Programmable CSD
Programmed to provide computational storage services

Fixed Purpose CSD
Performs a fixed function

Key interfaces to CSDs
PCle and NVMe: Local server offload
NVMe-oF: Offload over fabric e.g. NVMe/TCP
Ethernet to on-drive Linux - the drive is ‘just a server’
Programmable CSDs

Computational Storage workloads
- Workloads are developed, deployed to the drive and services advertised
- Workloads are initiated and results returned or stored
- Workloads are updated/enhanced and re-deployed

On-drive Linux is the simplest development and deployment
- Huge number of applications and protocols already available
- Wide range of development tools and vast open-source developer community
- Easy to deploy through existing infrastructure and operations teams

Other development systems have applications
- Bare-metal for hard real-time, FPGA development systems, ...
On-drive Linux: Two Main Approaches

1. ‘Just another networked server’: To the infrastructure, the CSD is a server
   Runs any standard Linux distribution, standard protocols and standard applications
   Workloads/containers downloaded using standard Linux systems e.g. Docker, Kubernetes…
   Standard applications, such as databases or ML, can run directly on the data in DRAM

2. NVMe/PCIe or NVMe/TCP: Enables standard drive or CSD
   Standard NVMe-oF TCPIP operates as normal – drive processes NVMe storage commands New
   NVMe CS commands received over NVMe intercepted to instigate CS functions Linux
   workloads/containers developed, deployed and actioned
Options to Add On-drive Linux

Three main options to run on-drive Linux
1. Add a separate applications processor SoC in-drive
2. Integrate into a single SoC for lower cost/latency
3. Single compute cluster for lowest cost/latency

Linux storage and DRAM requirements
e.g. Debian 9 ‘buster’ states system requirements...

Table 3.2. Recommended Minimum System Requirements

<table>
<thead>
<tr>
<th>Install Type</th>
<th>RAM (minimum)</th>
<th>RAM (recommended)</th>
<th>Hard Drive</th>
</tr>
</thead>
<tbody>
<tr>
<td>No desktop</td>
<td>128 megabytes</td>
<td>512 megabytes</td>
<td>2 gigabytes</td>
</tr>
</tbody>
</table>

Smaller Linux distributions are available
A typical 16TB SSD already has ~16GB DRAM
Low-cost, Low-power Compute

Linux requires an applications processor
Memory Management Unit (MMU) to virtualise memory
Arm Cortex-A series have 21 processors available and a strong roadmap
From a single processor to many clusters of compute
Meeting every possible performance point at the lowest possible power

Some eSSD controllers already use Cortex-A series processors
Other controllers, using real-time Cortex-R series, can easily add them

Arm Neon enables high-performance ML as standard
Neon Single Instruction Multiple Data (SIMD) greatly accelerates ML functions
ML processors, FPGAs, ISPs or dedicated hardware easily integrated
Computational storage is happening today
CSDs are available now from multiple manufacturers

SNIA CSD standards to deploy/manage workloads over NVMe/PCIe or NVMe/TCP
Linux delivers the fastest route for workload development, deployment and management

The drive as ‘just another networked server’ fully leverages Linux ecosystem
An Enterprise SSD connected via ethernet and running Linux is a low-cost, low-power server
I’ll be here all week

For more information, visit storage.arm.com
neil.werdmuller@arm.com
linkedin.com/nwerdmuller

Thank you