CXL Memory
Disaggregation and Tiering

Lessons Learned from Storage

Presented by Andy Mills, SMART Modular Technologies
Summary

- Memory tiering can learn a lot from storage
  - Scaling
  - High availability and redundancy
  - Recovery objectives (RTO/RPO)
  - Stateful vs. Stateless virtualized environments

- Topics covered
  - Quick Refresh on SANs, tiering, CXL memory expansion/sharing
  - Case Study: SSD Transparent Page Tiering – a Real World Product
  - Lessons learned that are useful for memory tiering
Background …

- **1980s** Multi-parallel processor system design (Transputer), real time embedded signal processing, parallel processor programming
- **1990s** Early involvement in 100Mbps optical fiber networks (FDDI), Ethernet networks, SDH/SONET and network redundancy/failover
- **2000s** Shared networked storage HA and virtualized storage appliances supporting tiering, remote replication, snapshot
- **2010s** Software defined storage, workload driven dynamic storage allocation and real time server/OS based storage tiering development and deployment
  - Developed production transparent page tiering solution for SSDs for Linux/Windows
- **2020s** PCIe Gen 4/5 Advanced SSDs and CXL memory disaggregation architecture and development
  - Now exploring pooled memory add-in components and appliances
Disaggregation and Composability

*Disaggregate {verb}: to separate into component parts*

*Composability: ability of system to configure disaggregated components*

- **Disaggregation**
  - 1993-4 – Storage Area Networks (FCSI formed in 1992) – first industry standards based storage “disaggregation”
  - 2013 – Intel and Facebook at Open Compute Summit first use the term disaggregation
  - 2017 – Intel White Paper: Disaggregated servers drive data center efficiency and innovation - Decouple CPU/DRAM and NIC/Drives from other server components

- **Composability**
  - 2020 – GPUs start to become disaggregated
  - 2022 – First demo of memory disaggregation

CXL Memory Expansion Types

- **CXL 3.0 Fabric/Expansion Cable Connection**
  - Standard DIMMs
  - CXL 2.0 Memory Expansion Add-in-cards
  - CXL 3.0 Fabric Switch

- **CXL 3.0 Memory Expansion Units**
  - JBOM Chassis
  - CXL 1.1/2.0 Memory Expansion

- **CXL 2.0 E3.S/L CMM and NV-CMMs**

- **CXL/DDR5 Capable Server Motherboard**
  - 6-12 DDR5 DIMMs
  - MC

- **Custom Modules**
Latency and Bandwidth, NUMA and CXL

Latencies approximate and vary by controller/memory types.
Caching Refresher

- Capacity visible to application in all cases is only that of the primary storage “tier”
- Caches rarely are larger than a few % of the primary storage
- Diminishing point of return due to cache flushing to make room for new data
• Capacity visible to application in all cases is **the sum of** the Fast and Slow tiers, unless a reservation scheme is used
• Reads and writes and direct through to the media via a simple page translation table
• Data/files are split across the fast and slow tiers (not copied)
Disaggregated Storage Refresher - SANs

**Networked Servers (no SAN)**

- Floppy net
- LAN/networked drive copy or backup

**SAN Based Environment**

- Compute Nodes
- SAN Appliance
- JBOD Expansion

**Networked Backup/Tiering**
- Early – manual copy
- Automated copying of files

As file sizes increased
- Batch file based tiering
- File extent tiering (partial files)
- Block based tiering
- File stubbing

**SAN Backup/Tiering**
- Automated copying of files
- File extent tiering

As SAN appliances migrated to virtualized
- Transparent block based activity tiering using SSD and HDD combinations
- Remote replication

**Newer forms: NVMe over fabrics that bridge gap between DAS and SAN**
SAN High Availability

- Removing single points of failure by ensuring multiple paths between compute nodes and storage
- Active-Active – both paths used for parallelism
- Active-Passive – earlier dual controllers only used one for active, the other for standby but needs to be in sync
- Active-Failover – one controller has failed until replaced
- Ability to replace controllers, switches and other key components (e.g. PSUs) without taking the system offline
Case Study: Transparent Page Block Tiering

- Auto Discovery and Classification
  - Profiling the tiers using live probe/performance data
- Page Virtualization
  - Efficient, low tax method for virtualizing and aggregating physical storage components to an application, optimized for PCIe Gen 4/5 NVMe
  - Memory cache to RAM or NVDIMM as “third” tier
- Hot Page Tracking and Ranking
  - High frequency sampling of storage IO access patterns to determine high use areas
  - Preference is to leverage/utilize hardware counters or fast memory if possible
- Cold Page Tracking and Ranking
  - Less intensive, background task to determine which areas of storage are not being heavily used
- Background Migration
  - Migrating performance data from cold to hot tiers and visa versa
  - Policy based
- APIs
  - Promote and policy setup
  - Page pinning
  - Manual or directed promote/demote controls

Key Tiering Components

- Discovery and APIs
- Virtual SSD
- Page Virtualization
- Hot/Cold Page Tracking
- Background Migration
- Fast Media
- Slower Media
- Virtual Pages (visible to application/OS)
- Virtual Page Remapping Layer
- Raw Block Storage Devices
Storage Tiering Stack

OS/Hypervisor/File System/Applications

Local Machine Utilities/Tools

Class Library

Virtual Block Devices

Block Virtualization Layer

Fast Block Level Remapping Layer (< 1us)

MicroTiering™ Block Data Migration

Policy Driven Stats & Decision Engine

Device Striping, Mirror, Redundant Copy

Block Device Layer Interface

UEFI Virtual Boot

NVMe

AHCI

SCSI/RAID/SAN

Storage Devices

File System

Logical Volume Managers

Kernel Block I/O

Device Drivers

EFI BIOS Drivers

System Management and Logging

System Management and Logging

UEFI Virtual Boot

NVMe

AHCI

SCSI/RAID/SAN

Storage Devices
Tiering Engine High Level Functions

Goal of the engine is to continuously modify the virtual mapping until optimal performance is achieved using a policy driven model.
Virtual Page States

- 'N' Virtual Volume Pages
  - Highly Active page stays on fast tier
  - Mostly Inactive page stays on slow tier
  - Page mapped to slow tier heating up

- Foreground Page Translation
  - Example Page Size: 128K-4M

- Fast Tier Pages
  - Page on fast tier cooled down
  - Page mapped to slow tier heating up

- Slow Tier Pages
  - Page of 'B' Blocks
  - Background Data Movement
  - Pages Migrated in Background

- Host I/O

Statistics Table
Page Promote/Demote Queues
OS Kernel Transparent Memory Tiering

**Kernel tiering form of software defined tiering**

**Process Allocation**
- Application (user)
- O/S or Container (user/kernel)
- Hypervisor (kernel)

Mix of Static and Dynamic Allocation

**Background tasks and processor/memory/PCIe affinity dependent**
- Tiering Virtualization (kernel)
- Tiering telemetry tables (kernel)
- Tiering Data movement (kernel)
- Core CXL Driver (kernel)
# Page Statistics Table

## Mega Region Counters
- RD IOs
- WR IOs
- RD Blocks
- WR Blocks
- Promotes Pending/Threshold
- Total Promotes

## Virtual Pages

## Virtual Page Counters
- RD IOs
- WR IOs
- RD Blocks
- WR Blocks
- Cur Policy Count
- Time Last Accessed
- Rigidity Control/Lock/Pcount
- Host Access Count
Tiering Policy Engine

- Allows users/administrators/system architects to tune policies
- Case study example used page activity counters
- Policy settings
  - Promote on read IO threshold
  - Promote on read and write IO threshold
  - Promote on write IO threshold
  - Same as above for MB/size (i.e. amount) of data changed per page
  - Rigidity settings – how fluid should a range of pages be
  - Page locking – pre and post promote actions
  - Numerous rates, time driven policies about when and how aggressively to move data
Analytics Data Collection and Reporting

Time based Activity View
- Workload Bursts
- Tier Promote Activity
- Time of Day

Volume based Activity View
- Zoom in on Activity
- Activity Over Capacity Range

- Restful/JSON or Redfish type interface
- Interface to logging and alert systems e.g. Splunk, ElasticStack, Logstash, Kibana…
Future Memory Tiering HA Appliance

Compute Node

HA CXL Memory Appliance

HA CXL Hybrid Memory Appliance
Summary and Lessons Learned

- Kernel based VMAP metadata maintenance
  - Took many iterations to get this right and solid
  - Lived through a “vmap repair” nightmare as we hardened for power loss and removable drives

- Processor Affinity and I/O Handler Process Placement
  - Tiering engine processes were allocated dynamically and not always on a CPU nearest the I/O path handlers
  - Moving data can significantly impact the application – need policies to deal with this
  - OS maintenance (e.g. indexing, virus scans) messes with your algorithms – need policies to deal with this

- Translation of I/O (Load/Store) Access
  - Using system memory for storing the tables is fast, however careful attention needs to be paid to CPU association of table vs. IO to prevent large context switches or wait times
  - Significantly more challenging for software based memory tiering

- Low level device media conflict management
  - SSD housekeeping and block migration often conflicts with tiering migration
  - Important for hybrid/persistent CXL storage and intelligent CXL appliances

- No one size fits all
  - Mission critical vs. non-critical, tiering in scale up vs. scale out, hyper-converged vs. tenant based
OCP Composable Memory Systems

- OCP Composable Memory System (CMS) is a sub-project within the Server Project
  - Led by Manoj Wadekar (Meta) and Reddy Chagam (Intel)
  - Members include device vendors, CPU vendors, CSP, ISV

- Charter
  - Focus on key applications driving CMS adoption
  - Establish CMS architecture and nomenclature
  - Identify gaps in specifications across full stack
  - Offer benchmarks enabling innovations in new and emerging use cases

- Currently working on draft specification for memory tiering

- More at: https://www.opencompute.org/projects/composable-memory-system
SMART at SDC23

Demo'ing E3.S CXL 2.0 Memory Module at SDC23 Hackathon

Wednesday, September 20 starting at 10:35 am in Salon 8
Please take a moment to rate this session.

Your feedback is important to us.