Using GPUs to Enable Highly Reliable Embedded Storage

Matthew Curry (curryml@cis.uab.edu)
Lee Ward (lee@sandia.gov)
Anthony Skjellum (tony@cis.uab.edu)
Ron Brightwell (rbbrigh@sandia.gov)

University of Alabama at Birmingham       Computer Science Research Institute
115A Campbell Hall                         Sandia National Laboratory
1300 University Blvd.                      PO Box 5800
Birmingham, AL 35294-1170                   Albuquerque, NM 87123-1319

High Performance Embedded Computing (HPEC) Workshop
23-25 September 2008

Approved for public release; distribution is unlimited.
The Storage Reliability Problem

- Embedded environments are subject to harsh conditions where normal failure estimates may not apply
- Since many embedded systems are purposed for data collection, data integrity is of high priority
- Embedded systems often must contain as little hardware as possible (e.g. space applications)
Current Methods of Increasing Reliability

- **RAID**
  - RAID 1: Mirroring (Two-disk configuration)
  - RAID 5: Single Parity
  - RAID 6: Dual Parity

- **Nested RAID**
  - RAID 1+0: Stripe over multiple RAID 1 sets
  - RAID 5+0: Stripe over multiple RAID 5 sets
  - RAID 6+0: Stripe over multiple RAID 6 sets
Current Methods of Increasing Reliability

• RAID MTTDL (Mean Time to Data Loss)
  – RAID 1: $\text{MTTF}^2/2$
  – RAID 5: $\text{MTTF}^2/(D \cdot (D-1))$
  – RAID 6: $\text{MTTF}^3/(D \cdot (D-1) \cdot (D-2))$

• Nested RAID MTTDL
  – RAID 1+0: $\text{MTTDL}(\text{RAID}1)/N$
  – RAID 5+0: $\text{MTTDL}(\text{RAID}5)/N$
  – RAID 6+0: $\text{MTTDL}(\text{RAID}6)/N$
RAID Reliability (1e7 hours MTTF, 24 hours MTTR)

The diagram illustrates the MTDL (Mean Time to Data Loss) for different RAID configurations as a function of the number of disks. Each line represents a specific RAID configuration, as indicated in the legend:

- **RAID N+3**
- **RAID 6+0**
- **RAID 6**
- **RAID 1+0**
- **RAID 5+0**
- **RAID 5**
- **RAID 0**

The x-axis represents the number of disks, while the y-axis shows the MTDL on a logarithmic scale. The data points show how the MTDL decreases with an increasing number of disks for each RAID configuration.
Why N+3 (Or Higher) Isn’t Done

• Hardware RAID solutions largely don’t support it
  – Known Exception: RAID-TP from Accusys uses three parity disks

• Software RAID doesn’t support it
  – Reed-Solomon coding is CPU intensive and inefficient with CPU memory organization
An Overview of Reed-Solomon Coding

• General method of generating arbitrary amounts of parity data for \( n+m \) systems
• A vector of \( n \) data elements is multiplied by an \( n \times m \) dispersal matrix, yielding \( m \) parity elements
• Finite field arithmetic
Multiplication Example

• \( \{37\} = 32 + 4 + 1 = 100101 = x_5 + x_2 + x_0 \)

• Use Linear Shift Feedback Register to multiply an element by \( \{02\} \)
Multiplication Example

• Direct arbitrary multiplication requires distributing so that only addition (XOR) and multiplication by two occur.
  – \{57\} \times \{37\}
  – \{57\} \times (\{02\}^5 + \{02\}^2 + \{02\})
  – \{57\} \times \{02\}^5 + \{57\} \times \{02\}^2 + \{57\} \times \{02\}

• Potentially dozens of elementary operations!
Optimization: Lookup Tables

- Similar to the relationship that holds for real numbers:
  \[ e^{\log(x) + \log(y)} = x \times y \]

- This relationship translates (almost) directly to finite field arithmetic, with lookup tables for the logarithm and exponentiation operators.

- Unfortunately, parallel table lookup capabilities aren’t common in commodity processors.
  - Waiting patiently for SSE5
NVIDIA GPU Architecture

- GDDR3 Global Memory
- 16-30 Multiprocessing Units
- One shared 8 KB memory region per multiprocessing unit (16 banks)
- Eight cores per multiprocessor
Integrating the GPU
29+3 Performance

Throughput (MB/s) vs. Data Size (KB)

- Throughput peaks at around 1480 MB/s for a Data Size of 174 KB.
- The throughput drops significantly before increasing again.
- The data size ranges from 58 KB to 348 KB.
Neglecting PCI Traffic: 3+3
Conclusion

• GPUs are an inexpensive way to increase the speed and reliability of software RAID
• By pipelining requests through the GPU, N+3 (and greater) are within reach
  – Requires minimal hardware investment
  – Provides greater reliability than available with current hardware solutions
  – Sustains high throughput compared to modern hard disks