SMART Modular Technologies
A PENGUIN SOLUTIONS® BRAND

USE CASE STUDY

Fault-Tolerant Distributed Machine Learning Training with SMART Modular Technologies' 32GB CXL® NV-CMM

Background

This document presents a hypothetical use case example showcasing how SMART Modular Technologies' 32GB CXL NV-CMM modules can be implemented to address fault-tolerance challenges in distributed machine learning training.

A leading AI research institute is developing large-scale language models using a distributed computing system comprising over 100 GPU nodes. The training process for these models is computationally intensive, often running for weeks or months. Hardware failures, network issues, and software crashes are common in such extended runs, leading to significant setbacks and wasted resources.

Challenge

The institute faced several critical issues in their distributed training pipeline:

Frequent Node Failures: On average, 2-3 nodes would fail each week, causing training interruptions.

Data Loss: Node failures often resulted in the loss of hours of computation, as the last valid checkpoint was typically hours old.

Checkpoint Overhead: Traditional disk-based checkpointing added significant overhead:

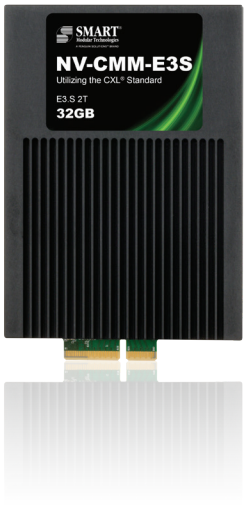
- 8-12ms latency per snapshot
- Up to 30 minutes to complete a full system checkpoint

Bandwidth Limitations: The existing memory subsystem (24GB/s bandwidth) created bottlenecks during state saves.

Recovery Time: Restarting training from the last checkpoint took 8-12 minutes, further reducing overall efficiency.

Energy Consumption: Frequent checkpointing to SSDs increased power usage and heat generation.

These issues collectively resulted in a 15-20% loss of compute time weekly and significantly extended the overall training duration for large models.



Solution: Implementing SMART Modular Technologies' 32GB CXL NV-CMM

The institute decided to integrate SMART Modular's 32GB CXL NV-CMM-E3S modules into their server infrastructure. This solution leverages several key technologies:

1. **CXL 2.0 Interface**
   - Provides 32GB/s throughput via PCIe Gen5 x8 connection
   - Achieves 200ns round-trip latency for near-instant state captures
   - Enables direct CPU-to-memory communication, reducing system complexity
2. **Persistent Memory Architecture**
   - Utilizes an onboard Energy Source Module (ESM) to back up 32GB DRAM to NAND during unexpected power loss
   - Implements AES-256 encryption to protect sensitive model parameters in memory
   - Ensures data persistence without relying on external power sources
3. **Global Persistent Flush (GPF)**
   - Leverages CXL's GPF feature for atomic writes, ensuring crash consistency
   - Allows for instant, system-wide persistence of checkpoints
4. **CXL.mem Protocol**
   - Enables memory expansion and sharing across the CXL interconnect
   - Facilitates seamless integration with existing memory subsystems

Implementation Details

The institute's engineering team developed a custom checkpointing system that leverages the NV-CMM modules:

```
python
import cxl
import distributed_ml_framework as dmlf

class CXLCheckpointer:
    def __init__(self, cluster, nv_cmm_devices):
        self.cluster = cluster
        self.nv_cmm_devices = nv_cmm_devices
        self.cxl_controller = cxl.Controller()

    def distributed_checkpoint(self):
        # 1. Coordinate checkpoint across all nodes
        self.cluster.broadcast("PREPARE_CHECKPOINT")

        # 2. Parallel write to NV-CMM via CXL.mem
        checkpoint_tasks = []
        for node, nv_cmm in zip(self.cluster.nodes, self.nv_cmm_devices):
            task = node.async_save_state(nv_cmm.address_space)
            checkpoint_tasks.append(task)
```

SMART® Modular Technologies
A PENGUIN SOLUTIONS® BRAND

```
        # Wait for all nodes to complete their checkpoints
        dmlf.wait_all(checkpoint_tasks)

        # 3. Trigger Global Persistent Flush
        self.cxl_controller.global_persistent_flush()

        # 4. Verify backup integrity
        if all(nv_cmm.verify_backup() for nv_cmm in self.nv_cmm_devices):
            self.cluster.commit_checkpoint()
            return True
        else:
            self.cluster.abort_checkpoint()
            return False

    def restore_from_checkpoint(self):
        # Load the latest checkpoint from NV-CMM devices
        restore_tasks = []
        for node, nv_cmm in zip(self.cluster.nodes, self.nv_cmm_devices):
            task = node.async_load_state(nv_cmm.address_space)
            restore_tasks.append(task)

        dmlf.wait_all(restore_tasks)
        self.cluster.resume_training()

        # Usage in the main training loop
        checkpointer = CXLCheckpointer(ml_cluster, nv_cmm_devices)

while training:
    try:
        train_epoch()
        if epoch % CHECKPOINT_INTERVAL == 0:
            checkpointer.distributed_checkpoint()
    except NodeFailureError:
            checkpointer.restore_from_checkpoint()
```

This implementation allows for:

- Frequent, low-overhead checkpoints (every 5 minutes)
- Near-instantaneous state capture across all nodes
- Rapid recovery from node failures

Results and Benefits

After implementing the SMART's 32GB CXL NV-CMM solution, the institute observed significant improvements:

Metric	Before NV-CMM	After NV-CMM	Improvement
Checkpoint Frequency	Every 30 mins	Every 5 mins	6x increase
Checkpoint Duration	15-20 minutes	<1 second	>900x faster
Recovery Time	8-12 minutes	~2 minutes	>4x faster
Training Efficiency	78%	94%	20.5% increase
Power Consumption for Checkpointing	450W	35W	92% reduction
Data Loss per Failure	2-3 hours	<5 minutes	>24x reduction

Key Benefits:

- **Enhanced Fault Resilience:** The system now survives various failure scenarios, including power outages, thanks to the ESM-backed persistence.
- **Improved Performance:** The 32GB/s throughput matches DDR4 bandwidth, eliminating checkpoint-related performance bottlenecks.
- **Cost-Effective Scaling**: Replaces expensive TSV-DIMM solutions (>$200K) with more affordable and flexible CXL-based scaling.
- **Reduced Energy Consumption:** Lower power usage for checkpointing contributes to better overall energy efficiency.
- **Increased Productivity:** The dramatic reduction in wasted compute time allows researchers to iterate on models faster and tackle more ambitious projects.

Conclusion

The integration of SMART Modular Technologies' 32GB CXL NV-CMM into the distributed computing infrastructure has transformed the institute's approach to fault-tolerant AI model training. By enabling frequent, low-overhead checkpointing and rapid recovery from failures, the solution has significantly improved overall system efficiency and reliability. This case study demonstrates the transformative potential of CXL-attached non-volatile memory in addressing the challenges of large-scale distributed computing applications.

SMART® Modular Technologies
A PENGUIN SOLUTIONS® BRAND

For more information, please visit: **www.smartm.com**

**Product images are for promotional purposes only. Labels may not be representative of the actual product.*

Headquarters/North America:
T: (+1) 800-956-7627 • T: (+1) 510-623-1231
F: (+1) 510-623-1434 • E: info@smartm.com

Latin America:
T: (+55) 11 4417-7200 • E: sales.br@smartm.com

Asia/Pacific:
T: (+65) 6678-7670 • E: sales.asia@smartm.com

EMEA:
T: (+44) 0 7826-064-745 • E: sales.euro@smartm.com

Customer Service:
T: (+1) 510-623-1231 • E: customers@smartm.com

© 2025 SMART Modular Technologies, a Penguin Solutions brand. All rights reserved. The stylized "S" in conjunction with "SMART", as well as "SMART Modular Technologies" are registered trademarks of SMART Modular Technologies. All other trademarks are the property of the respective owners. These materials are provided by SMART Modular Technologies as a service to its customers and may only be used for informational purposes.
04.07.25/NVCMMUseCaseStudy/Rev.1