Storage Area Networks in Embedded Processing

Jason Harnish
Northrop Grumman Corporation
Linthicum, MD, U.S.A.
Jason.Harnish@ngc.com

John Holland
Northrop Grumman Corporation
Linthicum, MD, U.S.A.
John.Holland@ngc.com

Jeremy Horner
Northrop Grumman Corporation
Linthicum, MD, U.S.A.
Jeremy.Horner@ngc.com

Timothy Linden
Northrop Grumman Corporation
Linthicum, MD, U.S.A.
Timothy.Linden@ngc.com

Steve Mattson
Northrop Grumman Corporation
Linthicum, MD, U.S.A
s.mattson@ngc.com

***Abstract*— This paper explores the application of Storage Area Networks (SAN) to size, weight, and power (SWAP) constrained systems. The goal is providing accessible, distributed, redundant storage with a high level of reliability and availability for multifunction sensors at the edge of the mission space. This paper describes the advantage of applying SAN technology to these missions and describes the use of high performance embedded processing to provide the processing and storage for decision-making processing and data sharing at a SWAP appropriate for challenging form factors. Specifically, this paper addresses how recent device-level integration advancements that bring Information Technology and Operational Technology onto the same silicon represent a paradigm shift which enables SAN in embedded processing platforms.**

Keywords—Storage Area Network, Information Technology, Operation Technology, Data Technology, SWAP-constrained

I. Introduction

The convergence of Information Technology (IT) and Operation Technology (OT) along with their definitions are addressed at a high-level in [1] and [2]. This paper discusses IT/OT convergence from the perspective that OT is the foundation upon which IT is built. Specifically, the layering of the data technology (DT) subset of IT on OT will be addressed, in an embedded computing context, as the basis for offering higher-level capabilities in a Service Oriented Architecture.

In the world of Information Technology (IT), Storage Area Networks (SAN) are a fundamental backbone to data centers and resource-rich traditional processing paradigms. SANs are usually not applied to the realm of Operation Technology, including embedded processing, with challenging size, weight and power (SWAP) constraints.

The current perception that SANs only can live in IT data center solutions requiring specialized hardware, deep protocol stacks, and complex multi-host coordination may become a myth due to advances in OT. In the past, traditional embedded processing systems utilized multiprocessor architectures which limited any consideration of applying developments in SAN technology. This is all changing.

Embedded processing systems are expanding into multi-processor requirements, and the use of SANs can facilitate fault tolerance, system functional hyper-convergence, and reduce embedded application software development timelines. Through the careful and selective application of multi-processor architectures and data movement methods used in data centers, limited-capability SANs can be achievable in SWAP-constrained OT embedded systems without significant trade of application-specific processing resources.

Section II of this paper describes the layering of Data Technology on Operation Technology. Section III describes device-level integration advances and the impact on embedded SANs. Section IV describes Storage Area Networks. Section V describes multiprocessor requirements for embedded processing systems. Section VI lists additional benefits of embedded SAN architectures as well as an example instantiation for the architecture.

II. Data Technology Layered on Operation Technology to Provide Higher Level Capabilities and IT Services

OT is generally associated with the control and/or automation of physical devices or processes such as you would find in a factory. OT implementations that are engineered for use in challenging environments and/or form-factors use appropriate packaged components, physical interfaces, protocols, design-principles and techniques as outlined in [3]. Although the focus of this OT is toward the physical realm, interfaces and access methods are often provided to enable access to and the coordination of OT devices through IT. Through these access methods, OT devices and systems can be viewed as data producers and/or consumers that interact with the physical world, and this provides a foundation upon which DT and IT can be built.

At a high level, DT enables the interaction between data producers and data consumers. This infrastructure provides the means to collect, store and retrieve data. It provides the means to access, analyze, and distribute data and information derived from that data both where and when it is needed. It is designed and architected to be scalable, resilient, and affordable.

Standardization and publication activities within the IT community allow for broad participation in the development and use of DT and IT infrastructure, and it allows for the independent and parallel development of compatible hardware and software. This hardware and software then can serve as building blocks to realize solutions to large and complex problems.

III. Device-level Integration Advancements Enable Embedded Storage Area Networks

Even when OT implementations provide DT/IT compatible interfaces and access methods, the potential value an OT implementation offers to a layered DT/IT service can be severely limited as both physical separation and latency increases and the effective interface bandwidth decreases. These limitations play a significant role in defining where the IT/OT integration boundary is established. The location of this boundary has additional implications because it influences the trade-space of implementation choices that take place when implementing OT. As the perceived barrier between OT and IT becomes more pronounced, there can be a tendency to make design choices in isolation that lead to highly custom solutions. In general, custom solutions rarely enjoy the advantages, outlined above, that frequently characterize the IT community.

Recent device technology advances bring hardware suitable for OT implementation (e.g. Field Programmable Gate Array (FPGA)) and processing hardware suitable for DT/IT implementation (e.g. ARM application-class processor) physically together onto the same piece of silicon where they now share a common, low-latency, high-bandwidth interface. These MPSoC (Multi-Processor-System-on-Chip) devices represent a paradigm shift as it moves the IT/OT integration boundary into the same physical space where OT once lived in isolation.

The ARM processors, running Linux with the real-time pre-emptive patch set, are able to handle basic IT functions in conjunction with the real-time controls of the hardware. The MPSoC's Programmable Logic fabric (i.e. FPGA) provides the opportunity for reconfigurable hardware acceleration and processor offload as needed to meet performance requirements.

Although there are still limitations, many of the core IT services can now be offered with these devices with remarkable performance capabilities.

Using industry standard testing tools, Flexible IO (FIO) Non-Volatile Memory Express (NVMe) performance was shown to be capable of achieving high throughput data rates. This test was run on a ZCU106 development board [5] with a ZU7EV FPGA and a Samsung EVO 970 Pro NVMe drive. Ensuring that direct I/O was enabled with a block size of 1 Megabyte, the results demonstrated that 3.2 Gigabytes per second for read and 2.1 Gigabytes per second for writes was achievable. Standard Linux drivers were used with an XFS file system [4]. Achieving these rates did not require any novel inventions or technologies. This testing proves that it is possible to implement a distributed storage solution in the embedded space. Novel technologies are needed to enable the network interfaces to leverage the capability demonstrated by this testing.

Currently the standard Linux network drivers and network stack typically limit the network interfaces connected to the ARM A53 cores in the Xilinx MPSoC FPGAs to approximately 1 gpbs. This demonstrated that a custom network driver and a custom DMA engine would be necessary to achieve higher data rates for the network interface attached to these ARM cores. As of this writing a custom network driver and custom DMA engine have been demonstrated to provide a 5 gpbs network interface to the ARM A53 cores in this FPGA. Profiling the network stack and network driver shows that the primary limitation is context switching. This shows that it is possible to achieve a full 10 gbps network interface in Linux running on the ARM A53 cores in the MPSoC.

The demonstrated NVMe performance of 25.6 Gigabits per second for read operations and 16.8 Gigabits per second for write operations indicate that an embedded SAN solution can be created where a single NVMe device is capable of saturating a 10 Gbps network interface. The aggregate performance of the SAN solution can then be scaled up to meet processing requirements by increasing the network capacity and the number of NVMe devices in the embedded system.

IV. Storage Area Networks

A Storage Area Network (SAN) is a specialized network designed to provide block-level access to storage devices at the network level. Components of this network include switches, storage devices, and the host system connecting the storage device to the network. Leveraging these components to create a SAN allows a designer to address three aspects of the system, availability, performance and utilization. Availability is addressed through the redundancy of the network links in the system. Performance is addressed through parallelism by distributing the physical storage medium to a set of hosts connected to the network. Utilization is addressed through the logical consolidation of the storage devices without the physical consolidation of the storage.

A. Availability

With respect to a SAN in the edge processing domain, availability of the storage can become a critical component. In this environment it may be more valuable to have a highly available database at the expense of consistency among the nodes. Artificial Intelligence (AI) and Machine Learning (ML) functions may require access to the data even in the case where there may be consistency issues in the underlying database. This is known as the CAP theorem (Consistency versus Availability in the presence of a Partition) [6]. In the case of a SAN structure it is possible to improve the availability of the data to ensure that AI/ML control loops continue to function.

B. Performance

High Performance Embedded systems may require high performance storage in support of sensing and processing functions. Bandwidth and latency are the two metrics that are generally the most interesting in this environment. This is not to disregard other metrics such as power efficiency. Bandwidth and latency tend to be given the greatest weight when making a trade. Not all sensing functions require low latency nor do all processing functions require high bandwidth; however, when these function coexist these two metrics need to be considered simultaneously. Using SAN-based technology performance,

concerns can be addressed by leveraging the parallelism supported in a switched network environment.

C. Utilization

Efficient utilization of the available storage in the embedded edge processing space is an important component that a SAN solution enables. Due to the reduced space and the reduced availability of power additional storage is not easily added to the system. Rather the efficient utilization of the aggregate available storage is necessary to provide sufficient storage for the data necessary for the system to perform the task(s) it is intended to perform. Treating each sensor as an independent island would necessitate that each sensor bring its own storage with it; however, if storage can be aggregated then the unused storage on one node in the system can become available to other nodes, thereby limiting the amount of discrete storage devices required for the system.

V. Multi-processor Requirements for Embedded Processing Systems

Requirements development for the application of SANs to OT platforms with multiprocessor architectures must carefully consider real-time data movement into the SAN, data access for task execution, and access to the network for internal and external users of pre-processed data and processed products. Requirements also must consider scalability for the high throughput, low latency, and SWAP constraints necessary for OT systems in the edge domain.

A. Distributed Processing

Traditional OT platforms include distributed, real-time, and embedded (DRE) systems with high performance embedded computing (HPEC). In the past, these OT processors often employed custom operating system solutions, optimized for the application or mission. For a SAN, open architectures and standard hardware and software interfaces are critical in order to provide internal and external access to data produced by the sensor and processed products produced by the network.

Critical requirements for the SWAP-constrained SAN include:

- Network of distributed processors
- Local storage (both volatile and non-volatile memory) for each processor for code storage and data storage for processing tasks
- Processing nodes that can serve as data and signal processing nodes, interface nodes, or maintenance nodes
- Data bandwidth into the network to support real-time sensor data
- External data access as well the ability to stream data to external devices
- Standard interfaces (example: Ethernet) to assure compatibility with sensors in a system of systems
- Standard SW/FW interfaces for the allocation of processing tasks to SW or FW depending on latency, throughput, or SWAP constraints

B. Fault Tolerance

Fault tolerance is critical for domains with long mission life or high availability requirements. The SAN must have features that support these requirements. Required features for the SAN and its distributed processing network include:

- Distributed processing and memory
- Self-monitoring of processing nodes
- Scalable, resilient storage (example: redundant array of independent memory (RAID), physical or virtual)
- Multiple paths for connections between processors
- Minimized hops between processors for the control and data planes
- Adaptive networking
- Dynamic task allocation
- Containerized software technology (example: Kubernates, Docker, Docker Swarms)
- Redundant sensor and host interfaces

C. System Functional Hyper-Convergance

The software and hardware infrastructure for the SAN should be flexible and adaptable with the SAN implemented virtually in software rather than in hardware. This infrastructure provides the scalability necessary to support variable network sizes from micro data centers in SWAP-constrained systems to larger systems.

VI. Additional Benefits of Embedded SAN Architectures

Embedded SAN architectures allow the physical distribution of processing functions, sensing functions and storage functions. Allowing these functions to be physically separated allows a designer to have additional flexibility on distributing power requirements and space requirements. While enterprise-class problems leverage this capability for cost and scale-out and scale-up problems, the embedded space can leverage this capability to address power and SWAP issues. Processing capabilities of all nodes can be aggregated due to the storage consolidation allowing unused processing resources, CPU time and/or FPGA resources throughout the system to become available in a consolidated environment.

A. Accelerated Embedded Application Software Development Timelines

Enabling SANs in the embedded space enables the development of new systems to be emulated on standard commercially available hardware. This allows a separation of concerns such that problems can be solved without requiring a complete working set of target hardware. Developers can have access to a virtualized environment which provides identical interfaces which will be available on the target embedded platform. Enabling this capability allows software development to begin immediately before any hardware is available.

Additionally, this provides the ability for the algorithms to be modeled accurately and performance metrics to be obtained quickly to determine the best methods for accelerating applications. Section III provides an example of this testing. Gathering these metrics can expose false assumptions about what is necessary to accelerate in hardware thereby reducing the time-line and reducing the risk of the project.

B. Common Architecture Across Product Line

A key element of scalability for the SAN is a common processing architecture that can be expanded or contracted to meet requirements across missions. Figure 1 provides an example of a common processing/memory node for a SAN architecture. The core of the SAN node is the MPSoC which provides multiple processing cores for processing tasks as well as a programmable logic fabric for embedding MAC and PHY interfaces for various physical interfaces. Programmable logic also enables the ability to implement bespoke processing functions which require increased DSP capabilities not available in the processing core silicon. Volatile memory is leverage for the operating system running on the ARM cores in the MPSoC. Memory controllers in the FPGA fabric are then used to increase the amount of memory that is available to the operating system to overcome the limited memory available in typical embedded systems. For example, rather than limiting the ARM core to 8 or 16 Gigabytes of memory an additional memory controller can be placed in the programmable logic to double the memory density to 16 or 32 Gigabytes respectively. Increasing the memory capacity of the node allows the embedded processing node to have similar memory capacity as that which is found in higher performance machines. While this is still significantly more limited than the memory capacity found in a high-end server, it is a significant improvement from the previous limitations of 4 Gigabytes found on the prior generation FPGA based devices with ARM cores such as the Xilinx Zynq 7000 series devices.

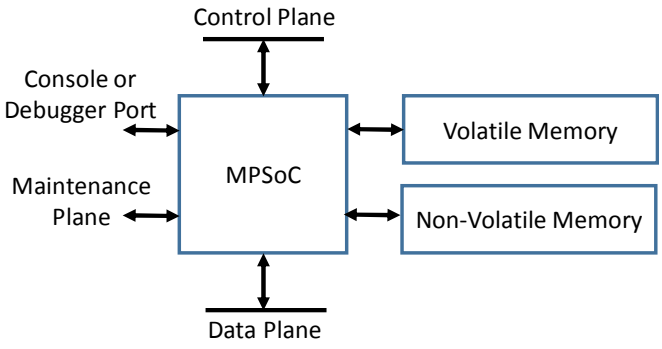

Figure 1: Common processing/memory node for a SAN architecture

Each node interfaces to the control, data, and maintenance planes of the SAN as well as to a console and/or debugger port to support software development. The Control Plane and the Data Plane may by be logically separated via VLANs (Virtual Local Area Network). Leveraging QoS (Quality of Service) technologies with Priority Flow Control (PFC) to classify the data streams allows the collapsing of the separate interfaces into a single physical interface which will reduce the SWAP of the design. Leveraging these technologies enables these data-center capabilities to be deployed in an embedded environment.

Figure 2 provides an example instantiation of a module or multi-socket MPSoC board for the SAN architecture. Multiple MPSoCs on a multi-socket module connect to the control, data and maintenance planes either through physical connections or logic connections depending on the deployment strategy and requirements. Each MPSoC has non-volatile memory to provide the backing for the distributed storage for the SAN. In this example instantiation, the architecture adds a baseboard management controller (BMC) for the interface to the maintenance plane where this controller function may be logically allocated to the R5 CPUs in the MPSoC or another device as appropriate for the target application. A discrete baseboard management controller may be required to interface with devices not supported by the R5s on the MPSoC thereby necessitating an alternate device such as a discrete programmable logic device to collect system telemetry (for example, temperature and voltage). Debugger ports such as JTAG and UARTs may also be required physical interfaces for each node during the development phase.

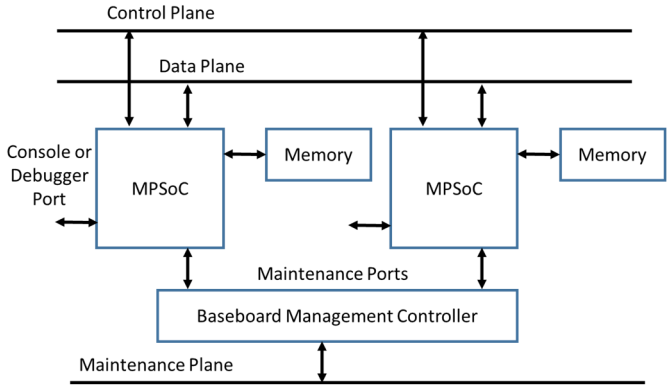

Figure 2: Example instantiation of a SAN module or slice

As described above, standard (rather than custom) interfaces support compatibility for hardware, software, and firmware. Table 1 provides examples of standard interfaces that can be used with the example instantiation in Figure 2.

Interface	Example Standard Interface
Control Plane	1 Gbit Ethernet
Data Plane	10 Gbit Ethernet
Maintenance Plane	1 Gbit Ethernet
Console Port	RS-232 UART

Table 1: Example standard interfaces for SAN node

Leveraging Ethernet interfaces for the Control and Data plane allows these planes to be converged to reduce the SWAP requirements for the system while leveraging widely used interfaces. Additionally, selecting standard Ethernet interfaces

also allows the selection of physically separate interfaces without changing the underlying supporting infrastructure. Where physically separate interfaces may be required to meet security requirements at the expense of SWAP. Other standard interfaces may also be selected for compatibility reasons.

Implementations and demonstrations of SAN applications for SWAP-constrained systems – including space applications – are in progress by the authors of this paper and others at Northrop Grumman. The authors look forward to future papers describing the results.

SUMMARY

This paper describes the application of SAN technology to SWAP-constrained systems on the edge of the mission space. Using embedded processing with MPSoC devices and distributed memory provides distributed, accessible, high-availability storage for decision-making processing systems with challenging form factors.

ACKNOWLEDGMENT

The authors acknowledge the support of Northrop Grumman Corporation for the work described in this paper.

REFERENCES

[1] "The Industrial Internet of Things, Vocabulary Technical Report", < https://www.iiconsortium.org/vocab/IIC_Vocab_Technical_Report_2.2.pdf>, retrieved 2020-06-08

[2] "The Industrial Internet of Things, Volume G1: Reference Architecture", https://www.iiconsortium.org/pdf/IIRA-v1.9.pdf, retrieved 2020-06-08

[3] J. Holland, J. Horner, M. Corbin, E. Glaser, and G. Petrosky, "Processor Building blocks for Space Applications," IEEE High Performance Extreme Computing (HPEC) Conference, September 2014.

[4] "XFS Filesystem Structure", 2nd Edition, Revision 1, Silicon Graphics Inc., 2006

[5] "ZCU106 Evaluation Board User Guide", UG1244 (v1.4), Xilinx, October 2019

[6] Dr. Eric A. Brewer, "Towards Robust Distributed Systems", PODC Keynote, July 2000