# Software-defined, programmable L1 dataplane: demonstration of fabric hardware resilience using optical switches

**Giannis Patronas[1], Dimitris Syrivelis[1], Paraskevas Bakopoulos[1], Prethvi Kashinkunti[2], Louis Capps[2], Nikos Argyris[1], Nikos Terzenidis[1], Eitan Zahavi[3], Luke Yeager[2], Elad Mentovich[3], Julie Bernauer[2]**

*[1] NVIDIA, Ermou 56, Athens 10563, Greece*
*[2] NVIDIA, 2788 San Tomas Expy, Santa Clara, CA 95051, USA*
*[3] NVIDIA, Hakidma 26, Ofer Industrial Park, Yokneam 2069203, Israel*

*Contact: giannisp@nvidia.com*

**Abstract:** We propose a programmable optical fabric design for Data Center networks that extends SDN to L1. We present experiments on our HPC/ML testbed leveraging the programmable network to automatically failover from hardware or software failures. © 2022 The Author(s)

## 1. Introduction

Software-defined control planes have revolutionized networking. Applications are provided with the power to configure the network according to their requirements, even when they need to share the networking resources with other workloads. Network infrastructures today are deeply programmable down to Layer 2 (L2), with the InfiniBand (IB) subnet manager (SM) being the lowest level example of a software-defined controller. This implies the programmability of the network stops at the physical infrastructure cabling, which is typically considered fixed after deployment. We propose to lift this limitation by introducing a workflow that extends the software-defined control capabilities down to L1. Software-defining the physical layer transforms network cabling from a rigid infrastructure to a programmable resource that allows physical topology changes to be carried out at runtime. This new functionality sets the foundations for a variety of new network operations that were not feasible before, but also introduces new implications that need to be dealt with at the higher levels of the network stack.

The ability to program L1 at runtime using optical switches enables several new use cases. The first one, which is also the focus of our current evaluation, is providing resiliency against HW and SW failures in the fabric (switches, transceivers and/or servers). The effect of failures on the utilization and efficiency of computing clusters is clear across the industry [1-3], highlighting the importance of creating resilient networks. Business-critical applications require continuous availability; downtime means lost revenues, lost customers and damage to the reputation of the company. Another potential use case for L1 programmability is modifying the physical topology of a network to fit application requirements on demand, e.g. to create a torus/mesh among the leaf switches of a fat tree to reduce communication time for latency-sensitive applications. Alternatively, in oversubscribed networks, bandwidth can be allocated on demand to parts of the network to offer different QoS based on the physical topology. Finally, isolation can be applied in the physical layer to disconnect network elements between multiple tenants or to isolate hosts that have been identified as potential threats. Our PoC targeted the IB fabric, but these concepts can be applied to NVLINK and Ethernet as well.

Without the ability to change the physical connectivity, current solutions for failure recovery focus on adapting the forwarding configuration to exclude the failed paths where possible. Examples of software features in IB are SHIELD [4] and adaptive routing that takes advantage of alternative paths. These protocols have two significant limitations. First, they can only be used in the case that alternative paths exist; failures on the leaf switches (that will disconnect servers from the network) or on the servers cannot be mitigated this way. Secondly, they are not able to recover the full performance of the cluster. Another approach to enhance resiliency is to add redundant hardware to replicate the whole or parts of the network (e.g. Dual ToR). This approach has the disadvantage of requiring significantly more hardware and leads to underutilized resources.

## 2. Reconfigurable fabric for resilient systems

We leverage optical circuit switches to implement the L1 programmable dataplane. The optical circuit switches redirect the light based on I/O permutations defined by an electrical control interface. By introducing them among the switching layers of a given network topology, as in Figure 1a, we enable the programmable change of the point-to-point fiber connections permutation. Figure 1a shows the network architecture that targets the resilience use case in a small scale 2-level (leaf-spine) Fat Tree. We add redundant electrical switches (RS – Redundant Spine and RL –

Redundant Leaf) and redundant servers to the network. The redundant devices are connected to available ports of the optical switches, as any other main network element. When a device failure is detected, the corresponding optical switches are properly configured to disconnect it from the network and replace it with a redundant unit. The proposed design allows a programmable degree of resilience: the ratio between the main and redundant devices may vary depending on the system requirements. In addition, the proposed architecture can isolate security threats and minimize the downtime during maintenance periods, while serving as a generic programmable dataplane.

We designed and implemented appropriate control plane software that can be viewed as an SDN stack extension for L1 dataplane control. An appropriately designed graph representation backend that reflects the physical network topology (which also includes the optical switching elements), provides the required system modelling support for our controller logic. Subsequently, we introduce a collection of concepts and algorithms that allow the described SDN L1 controller to identify the different topology possibilities of a given deployment, carry out the physical topology changes and signal the L2 layer controller to adapt to the changes of the physical network. Figure 1 (b and c) shows the described control loop of the system, in purple the SDN L1 (Optical Fabric Manager - OFM) is the SW that carries out physical changes and delivers notifications to L2 (which in case of IB is the Subnet Manager). Likewise, L2 can be extended to request physical topology changes. A failure detection mechanism (out of the scope of the current work) notifies the OFM that a device needs to be replaced. The OFM calculates and enforces the appropriate optical connections, e.g. in the case of a Leaf switch failure its replacement by RL1 in Fig 1a. Subsequently, the network controller includes the newly introduced device in the network. The proposed workflow enables the recovery of the network's capacity to 100% in a few seconds. In addition, as discussed in the following sections, we are preventing application crashes caused by device failures.

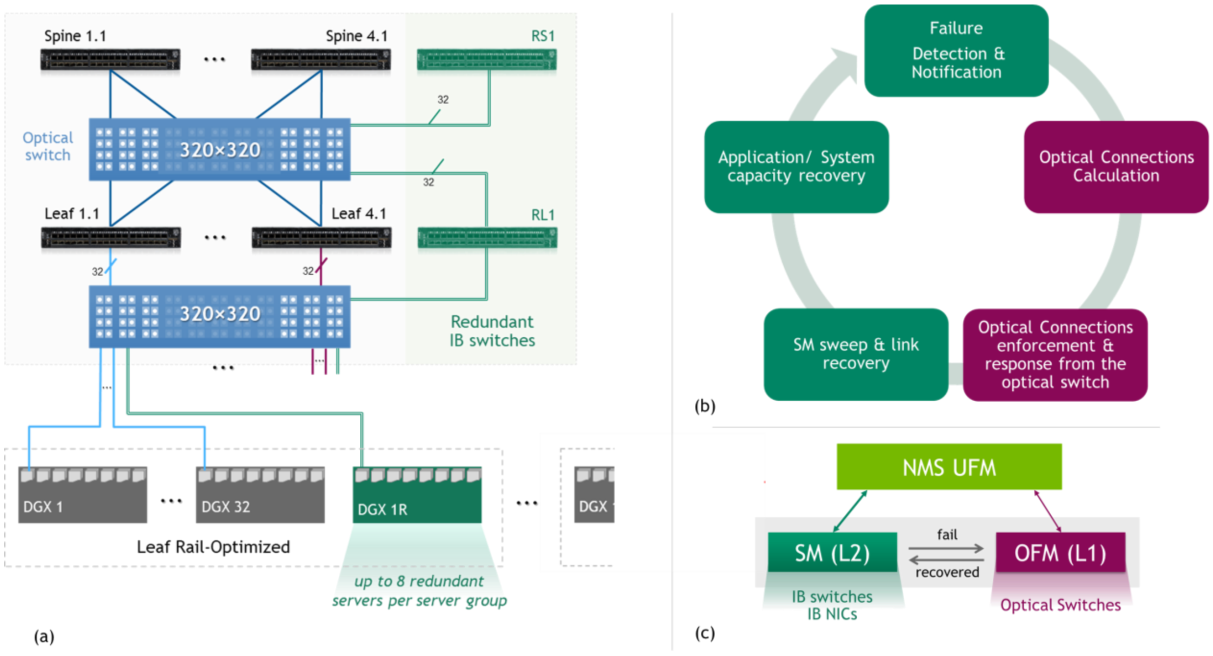

Fig. 1: a) Resilient Architecture Overview, b) Control Loop flowchart and c) SW overview

## 3. Testbed description

To demonstrate resilience, an optical switch is interleaved in all small-scale POD connections and a redundant IB switch is added to the leaf and spine level to replace failed leaf or spine switches. The testbed consists of 4x DGX servers [5] and 14x IB Quantum switches [6]: 8 of them connected as Leaf switches and 4 of them as Spine switches. The two additional IB switches are the redundant ones: one redundant Leaf (RL, as in Fig. 1a) and one redundant Spine (RS). We use off-the-shelf L1 optical switches [7]. For the optical links we opted for 200 Gb/s CWDM pluggable transceivers [8] since they significantly reduce the required optical switch ports and have sufficient link budget to support the interception by the optical switch. The DGXs have 8x IB interfaces (8x rails), with each rail connected to a different Leaf switch. The Leaf switches are fully connected with the Spines, i.e. there is no oversubscription. All the connections are intercepted by the optical switch allowing for a variety of experiments; for the current evaluation we focus on the replacement of IB switches.

## 4. Experimental procedure & Results

We emulate switch failures and trigger the SDN L1 controller to search physical topologies for failure mitigation. As a result, the redundant switch takes over the role of the failed one in the physical topology, and the IB subnet manager receives a topology change notification which instructs it to repair the L2 network configuration. With appropriate handling of IB transport timeout, the running applications can resume execution after this change.

Figure 2 shows the results of our tests with UCX [9] and NCCL [10] collective communications libraries. In the presented testing scenario, we emulate IB switch failures. The diagrams show the bandwidth (y-axis) on the IB interfaces of one of the DGXs involved in the experiment over time (x-axis), for all-to-all and all-reduce microbenchmarks. The benchmarks produce identical traffic among the 4x DGXs and among the interfaces. We run the microbenchmarks, emulate the failures and monitor the performance and status of the application over time. During Spine failures the capacity of the system is reduced, due to the reduction of active links, but the application does not crash since alternative paths are available. When enabled, our solution restores the full performance of the cluster (Spine failover) in a few seconds. In the case of failures on the Leaf layer (Leaf failover), the application would crash and the IB interface affected would remain offline until the problem is fixed. With the resilience solution enabled, the application keeps running after a few seconds interruption (currently ~7s, but it is subject to optimization) and the full capacity of the system is restored.

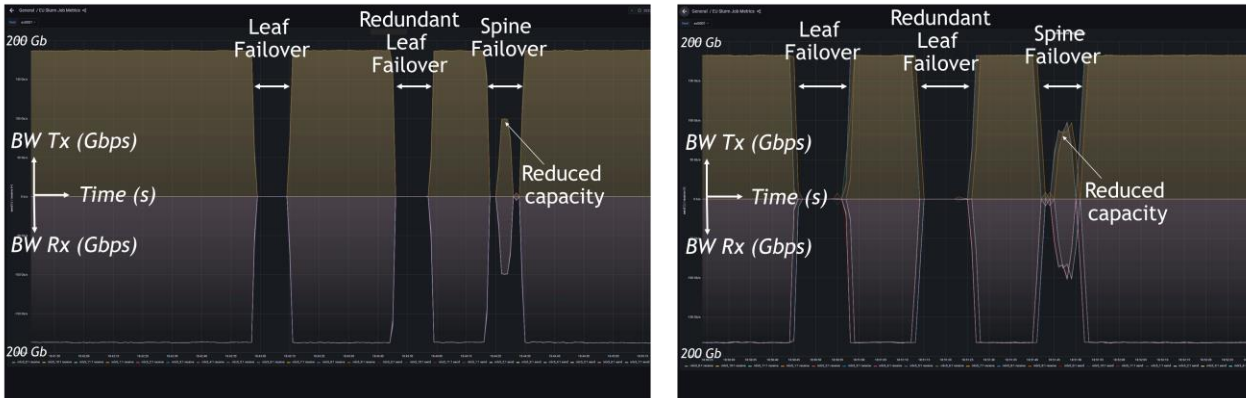

Fig. 2: Demonstrating BW recovery for microbenchmarks with the resilience solution enabled: a) on the left OSU all-to-all and b) on the right NCCL all-reduce. Showing BW for all links over time: the yellow part shows the Tx bandwidth, while the purple part shows the Rx BW. The plots are overlapped for all links of one DGX.

## 5. Conclusion & future work

We presented a system design and a workflow that combined enable L1 programmability. We have built an HPC/ML testbed and evaluated the resilience use case by emulating switch failure scenarios. By adding the optical switching network and the redundant devices, we show automated recovery of the full capacity in a few seconds; and in addition, we avoid application crashes during Leaf-level failures. In a future work, we plan to present a cost analysis and detailed results from additional experiments including the rest of the use cases.

## References

[1] Zhang, Susan et al. "OPT: Open Pre-trained Transformer Language Models." ArXiv abs/2205.01068 (2022): n. pag.
[2] Justin Meza, Tianyin Xu, Kaushik Veeraraghavan, and Onur Mutlu. 2018. A Large Scale Study of Data Center Network Reliability. In Proceedings of the Internet Measurement Conference 2018 (IMC '18). Association for Computing Machinery, New York, NY, USA, 393–407.
[3] Rachee Singh, Muqeet Mukhtar, Ashay Krishna, Aniruddha Parkhi, Jitendra Padhye, and David Maltz. 2021. Surviving switch failures in cloud datacenters. SIGCOMM Comput. Commun. Rev. 51, 2 (April 2021), 2–9. https://doi.org/10.1145/3464994.3464996
[4] https://www.nvidia.com/content/dam/en-zz/Solutions/networking/infiniband/wp-the-shield-self-healing-interconnect.pdf
[5] https://images.nvidia.com/aem-dam/Solutions/Data-Center/nvidia-dgx-a100-datasheet.pdf
[6] https://nvdam.widen.net/s/zmbw7rdjml/infiniband-qm8700-datasheet-us-nvidia-1746790-r12-web
[7] https://www.polatis.com/series-7000-384x384-port-software-controlled-optical-circuit-switch-sdn-enabled.asp
[8] https://www.nvidia.com/en-us/networking/ethernet/optical-transceivers/
[9] https://docs.nvidia.com/networking/display/hpcxv24/Unified+Communication+-+X+Framework+Library
[10] https://developer.nvidia.com/nccl