Juri Schmidt
3rd year PhD candidate

Computer Architecture Group
Heidelberg University
juri.schmidt@ziti.uni-heidelberg.de

»NAM«

Network Attached Memory

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

About Network Attached Memory

Motivation

The problems with the traditional memory organization are:
Waste of time and energy because a Processor must wait for the data to become available (Fig.1)
Energy (and time) required to transport data correlates with the physical distance (Fig.2)
Time = Energy

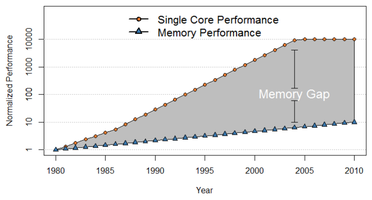


Figure 1: Historical evolution of the processor-memory gap (the 'memory wall')

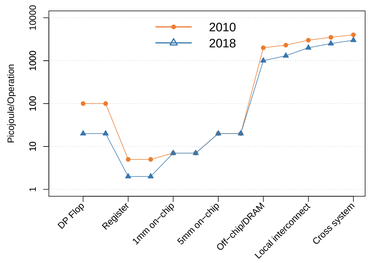


Figure 2: Comparison of the energy required to transport information . 2010 to 2018 (projected)

Network Attached Memory (NAM) as an approach for HPC:
Process data in the network domain as it streams through (Processing in Network, PIN)
PCIe card with FPGA + Hybrid Memory Cube and EXTOLL + PCIe interface
No real Processing in Memory (PIM) since processing elements are not part of the memory stack
NAM performs Near Data Computing/Processing (NDC, NDP) to prototype PIM

NAM use-case: The DEEP-ER project

In the european funded Dynamical Exascale Entry Platform - Extended Reach (DEEP-ER) project[1] the NAM is connected to the EXTOLL high performance interconnection network as a dedicated device to perform an N+1 XOR parity checkpoint-restart.

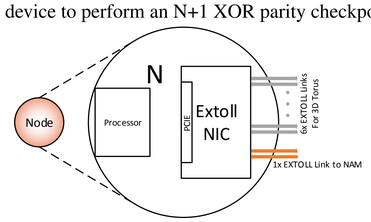


Figure 3: Abstract view of a Node in the DEEP-ER system

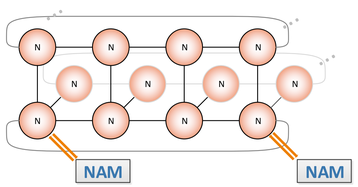


Figure 4: Abstract view of the DEEP-ER network environment

Use-cases other than checkpoint-restart

- Global Operations
- Graph processing
- General purpose shared memory, metadata server

Component Status

NAM Hardware	Extoll FPGA Link	HMC controller openHMC	Checkpoint/Restart
Done ✓	**Done ✓**	**Done ✓**	**Under evaluation**

[1]The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013)

The NAM Hardware

Aspin v2

The NAM hardware 'Aspin v2': Xilinx Virtex7 FPGA + Hybrid Memory Cube (HMC) on a single PCB. Access to the FPGA through:

- PCIe x16 (as generic Gen3x8 or 16x SerDes)
- Two HDI-6 12x connectors over Extoll
- Top connector connected to HMC to increase capacity

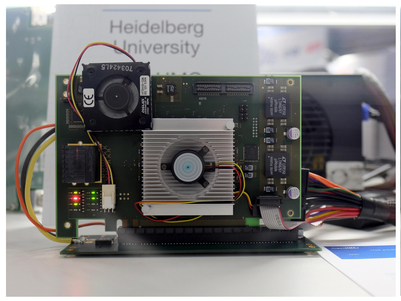


Figure 5: The NAM Hardware 'Aspin v2'

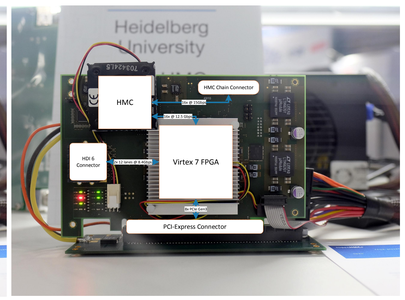


Figure 6: Aspinv2 - Component Block Diagram

Implementation Details

Figure 5: Structure of the FPGA design. It consists of four main building blocks:

- HMC host controller *openHMC*[1][2]. Own development. Used to evaluate HMC[3]
- FPGA implementation of an EXTOLL link
- Bridge to connect the EXTOLL link to openHMC
- The actual 'NAM intelligence'. Application specific processing elements

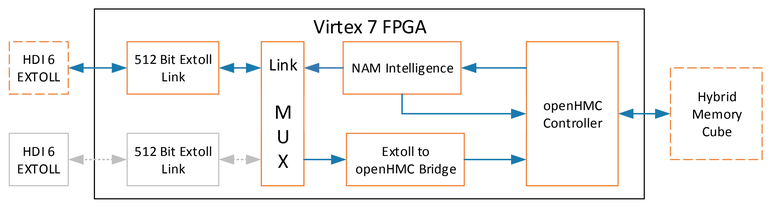


Figure 7: NAM Design Block Diagram

The NAM FPGA logic has the following characteristics:

- 312.5MHz NAM / HMC clock domain (matches the HMC bandwidth at 16x, 10Gbps)
- ≈210MHz Extoll clock domain
- Approximately 30% FPGA fabric utilization with one Extoll link (50% with two links)

Memory Subsystem Topologies

Figure 6 and 7: Two example topologies where memory capacity is increased by adding ("chaining") additional HMCs

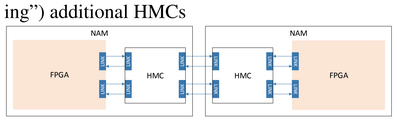


Figure 8: Memory Subsystem with 2 NAMs

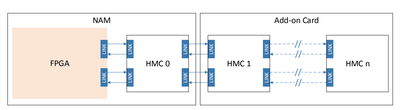
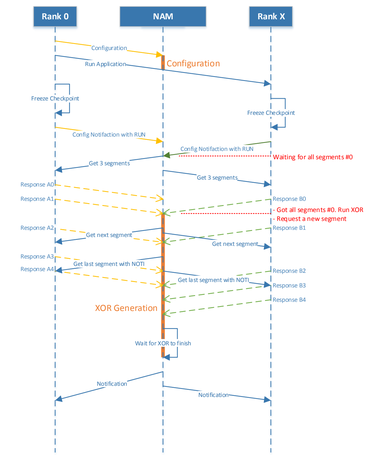


Figure 9: Attach an add-on HMC equipped card

Improve Resiliency with Parity Checkpoints

One of the objectives of the DEEP-ER project is to speed-up and reduce overhead of checkpoint/restart mechanisms. As one approach the NAM serves as a dedicated resource to calculate and store the parity of checkpoints, also known as *N+1* parity. N+1 allows recovery after any single-node failure.



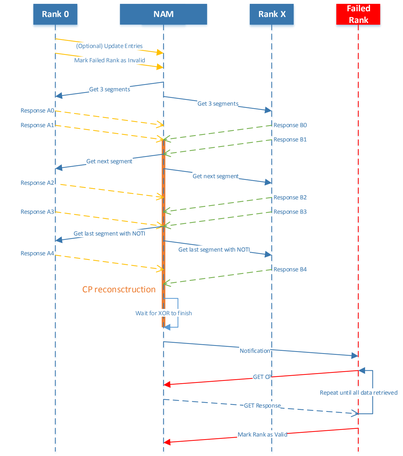


Figure 10: Checkpoint Scheme: The NAM is first configured and then acts as an active device to fetch data and calculate the XOR parity in a sliding window approach. XOR. The result is the missing checkpoint. The failed rank can then read the checkpoint from the NAM

Figure 11: Restart Scheme: The failed rank is invalidated. NAM retrieves checkpoints from remaining ranks and applies XOR. The failed rank Checkpoint size: 5 segments. Sliding window size: 3 segments

NAM Software

libNAM is based on the Extoll RMA engine. It extends function calls of the *libRMA* by application specific functionality such as checkpoint writing and reading. A NAM manager (NM) allocates address space and administrate access rights to participating processes.

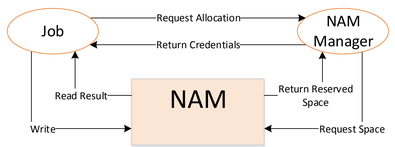


Figure 12: NAM Manager: Address Space Allocation

References

[1] Computer Architecture Group, University of Heidelberg. *openHMC Home*. Apr. 2015. URL: www.uni-heidelberg.de/openhmc.

[2] J. Schmidt and U. Bruening. "openHMC - a configurable open-source hybrid memory cube controller". In: *ReConFigurable Computing and FPGAs (ReConFig), 2015 International Conference on*. Dec. 2015, pp. 1–6. DOI: 10.1109/ReConFig.2015.7393331.

[3] J. Schmidt, H. Froening, and U. Bruening. "Exploring Time and Energy for Complex Accesses to a Hybrid Memory Cube". In: *Memory Systems, 2016 International Symposium on*. Oct. 2016. DOI: 10.1145/2989081.2989099.

For more information visit us at:
EXTOLL booth #3836