ETH zürich

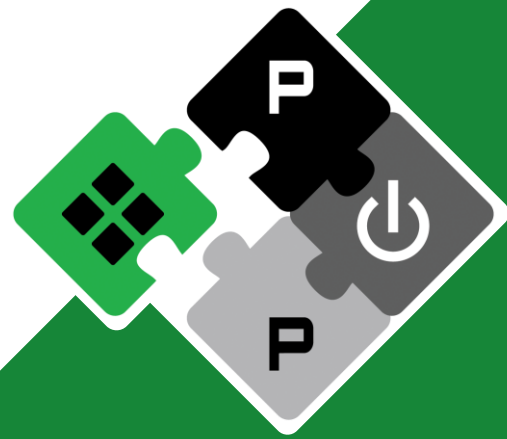
ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

NextRAN-AI – 05/09/2025

Integrated Systems Laboratory (ETH Zürich)

Marco Bertuletti mbertuletti@iis.ee.ethz.ch
Yichao Zhang yiczhang@iis.ee.ethz.ch
Mahdi Abdollahpour mahdi.abdollahpour@unibo.it
Alessandro Vanelli-Coralli avanelli@iis.ee.ethz.ch
Luca Benini lbenini@iis.ee.ethz.ch

PULP Platform
Open Source Hardware, the way it should be!



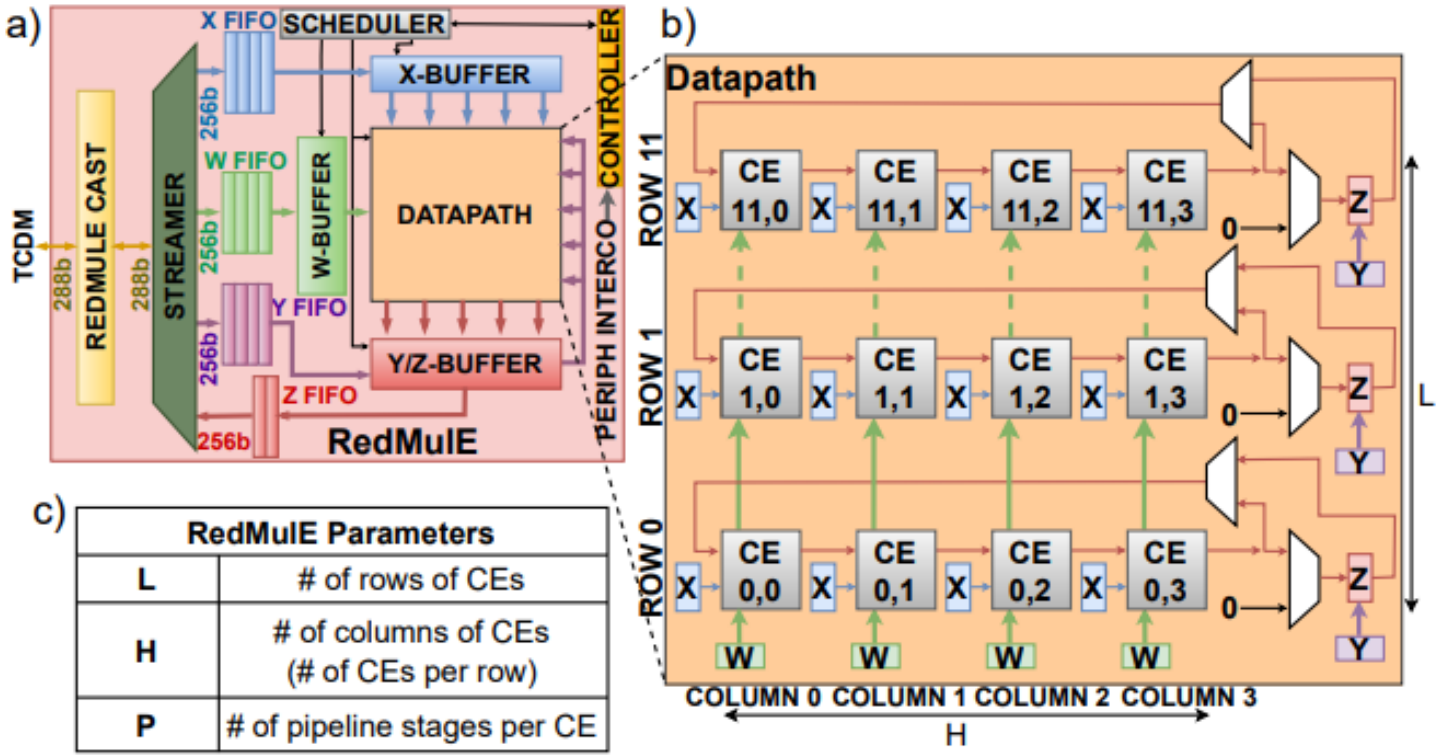
pulp-platform.org
@pulp_platform
company/pulp-platform
youtube.com/pulp_platform

Roadmap

- **TensorPool:**
  - Optimize RedMulE to L1 connection
  - TeraPool physical design in 7nm
  - TeraPool + RedMulE (TensorPool) physical design in 7nm
  - PPA on model microkernels and operators (combined RedMulE&Cores)
- **System Performance:**
  - Simulation speed is impaired by large design size → from RTL simulation to higher abstraction level, e.g. GVSoC
  - TensorPool GVSoC model developement
  - Data-Movement and end-end performance

RedMulE tensor-core

- Inner product GEMM engine Z = Y + X*W
- Parametrizable FMA array size (scales quadratically with the mem. port width)

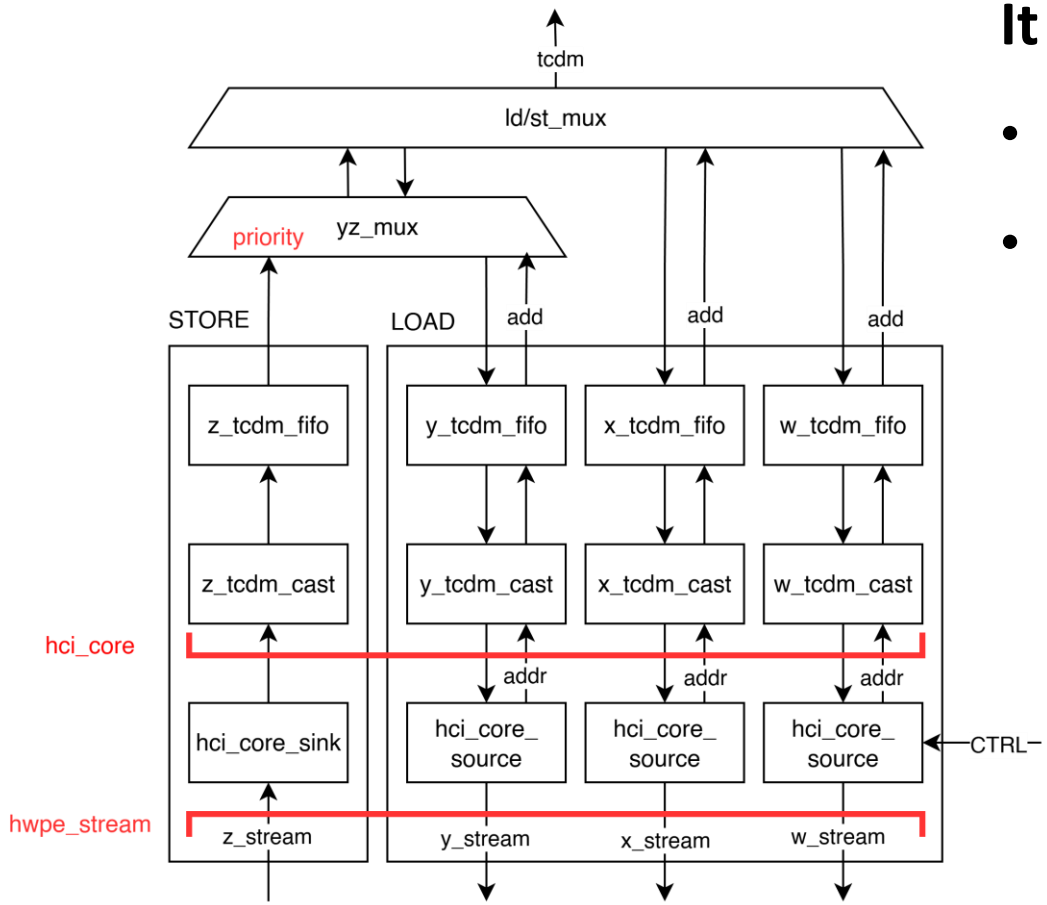


c)

RedMulE Parameters	
L	# of rows of CEs
H	# of columns of CEs (# of CEs per row)
P	# of pipeline stages per CE

TARGET = integrate RedMulE in MemPool (NUMA interconnect)

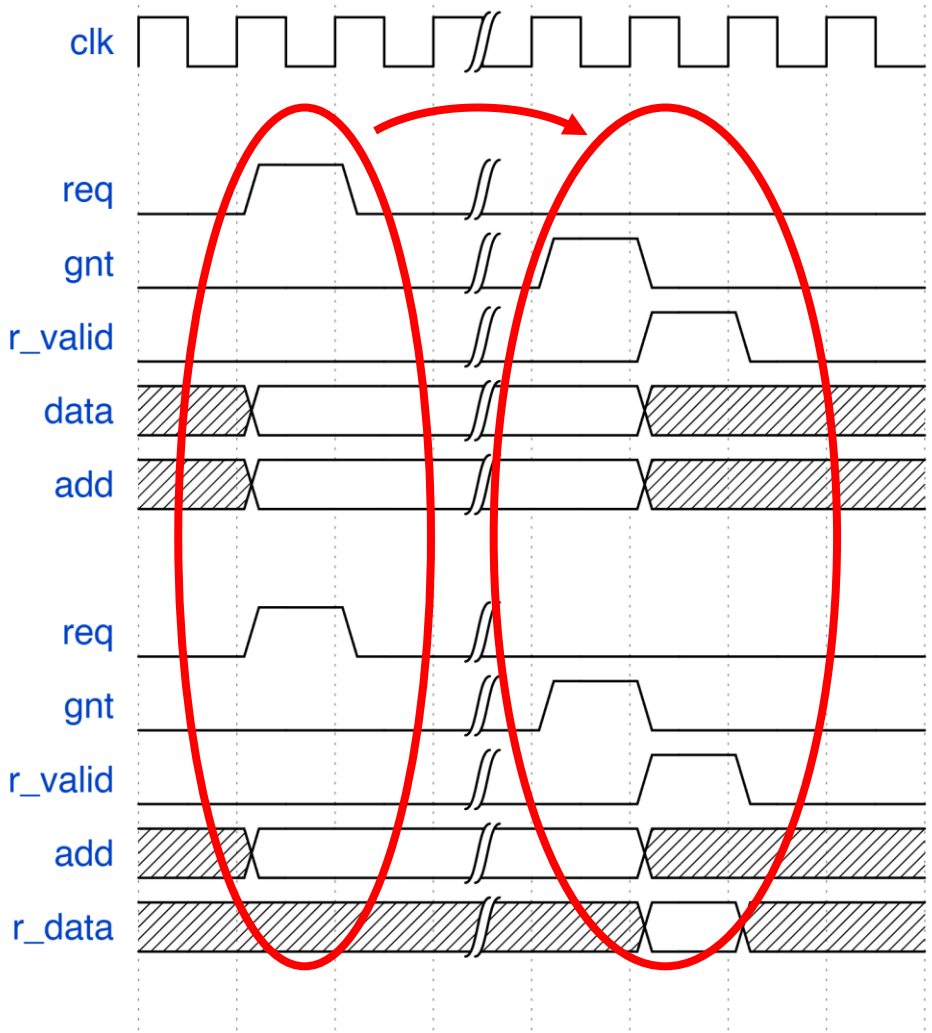
RedMulE Streamer



It is RedMulE's memory interface:

- Control generates addresses (streams)
- Streamer:
  - Generates wide H*(P+1)*16bit + 32bit TCDM request, based on the addresses from control
  - Applies conversions (e.g. 16/8bit operands)
  - Multiplexes x, w, y, z streams

RedMulE Streamer: Protocol



Requests are based on a req/gnt/r_valid protocol:

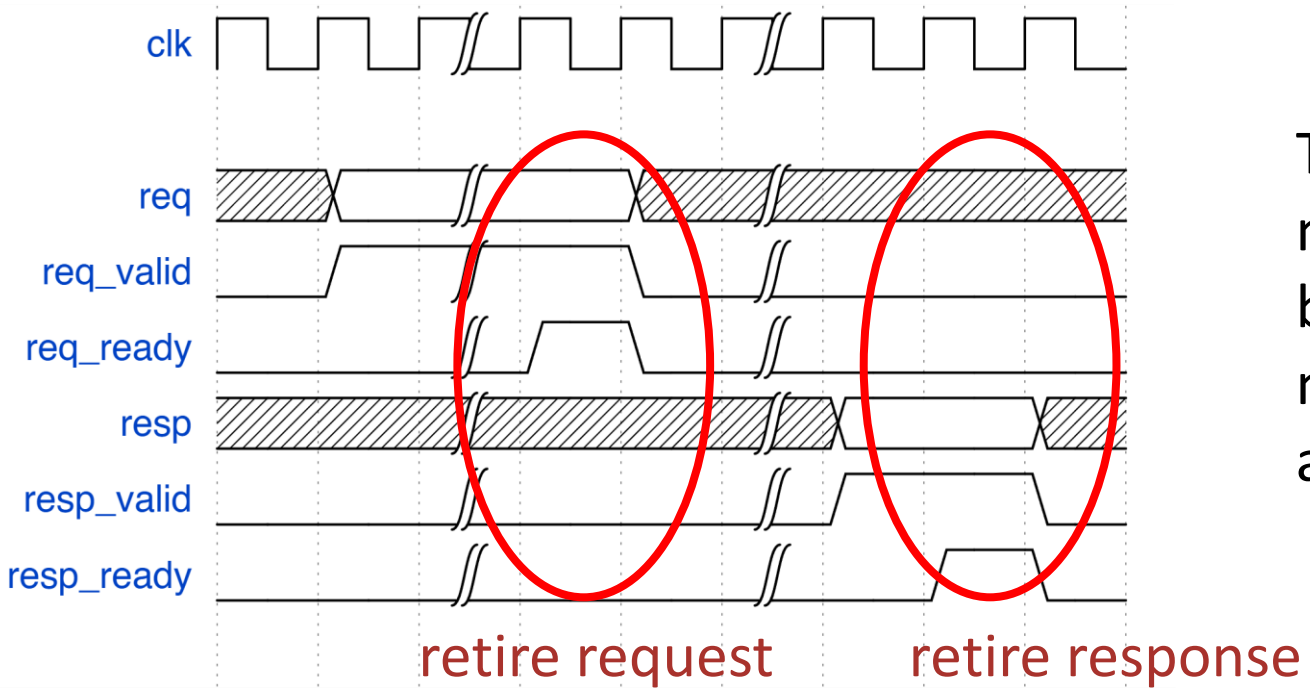
- req is asserted on valid requests
- A new request can be issued when the slave grants the transaction
- The response must be valid one cycle after the transaction is granted

When the memory takes multiple cycles to handshake no new request can be posted on the interconnect!

Enabling outstanding requests

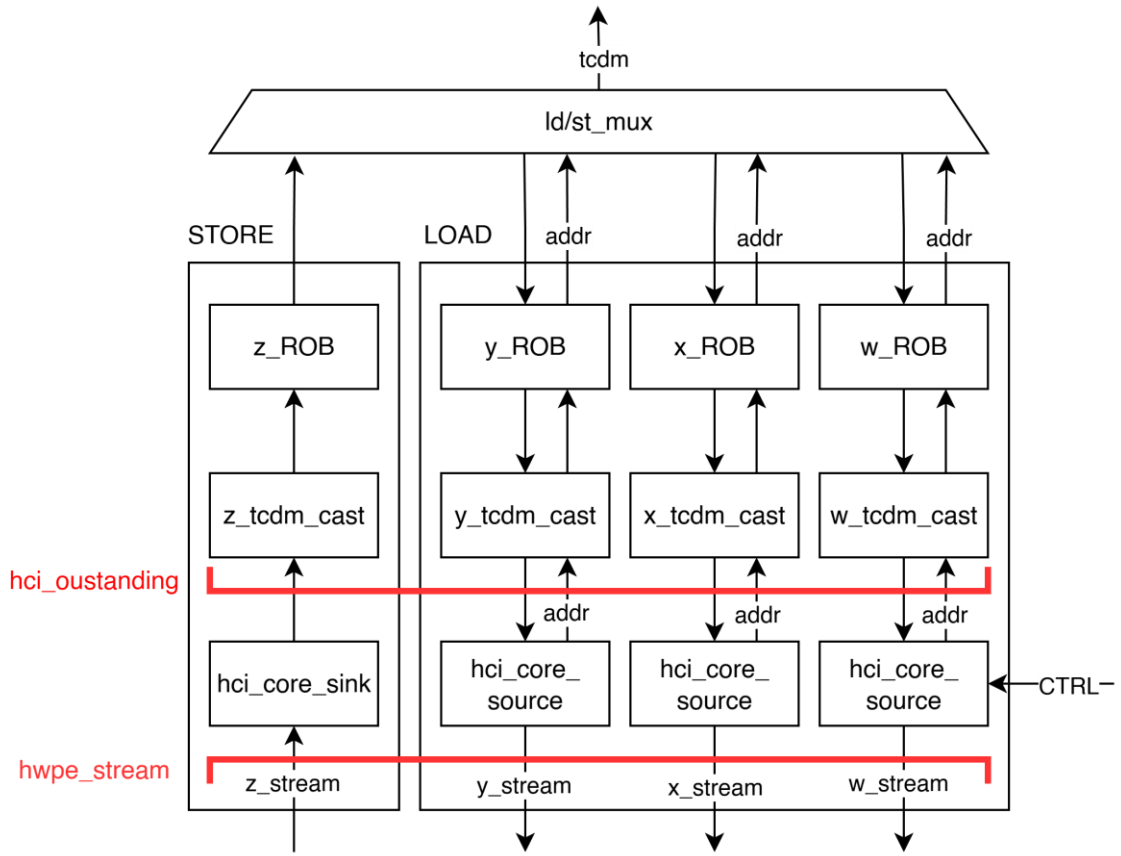
A req_valid/ready, resp_valid/ready protocol decouples req and resp

- A new valid req can be posted as soon as req_valid = resp_ready = 1’b1
- Valid responses can be collected later, when resp_valid = resp_ready = 1’b1



This however implies that multiple transactions could be in-flight, therefore we need to send transaction IDs and reorder responses

Enabling outstanding requests

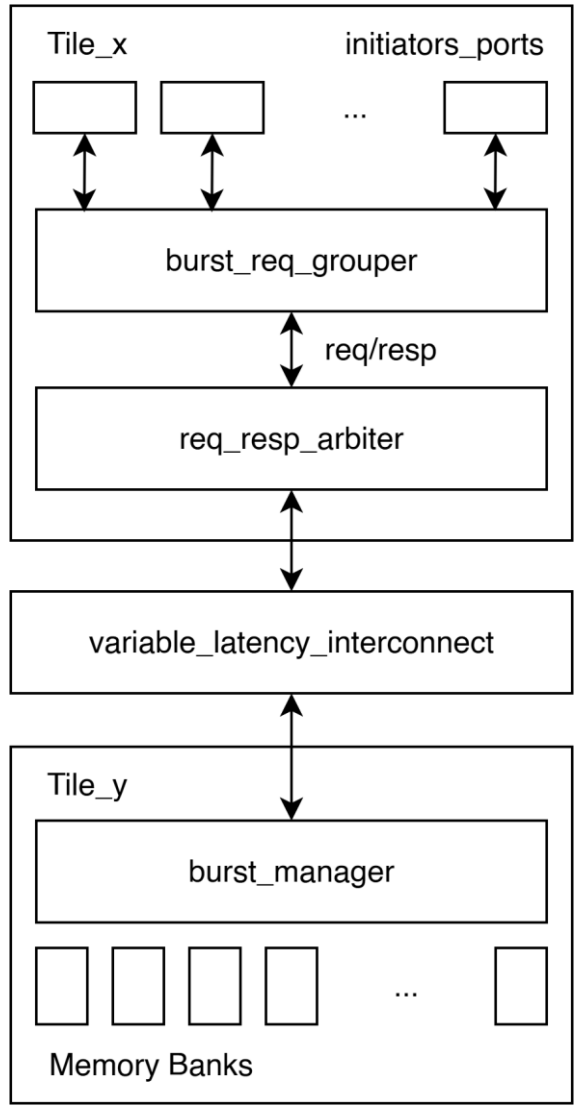


Progress:

- Replaced the req/gnt/r_valid protocol with a req_valid/ready resp_valid/ready protocol
- Added ROB with parametrizable number of transactions on each stream

Integration in MemPool: Recap on bursts

- System with distributed shared-memory and hierarchical core-memory interconnect
- Arbitration of wide parallel TCDM requests in shared interconnect resources → performance penalty
- **Burst the transaction:**
  - Group a request to consecutive addresses in a burst (only first port is valid and is arbitrated)
  - Propagate request in the shared-memory hierarchical interconnect
  - Identify burst request and redistribute to memory banks with consecutive addresses in the destination Tile
  - Send back wide-response



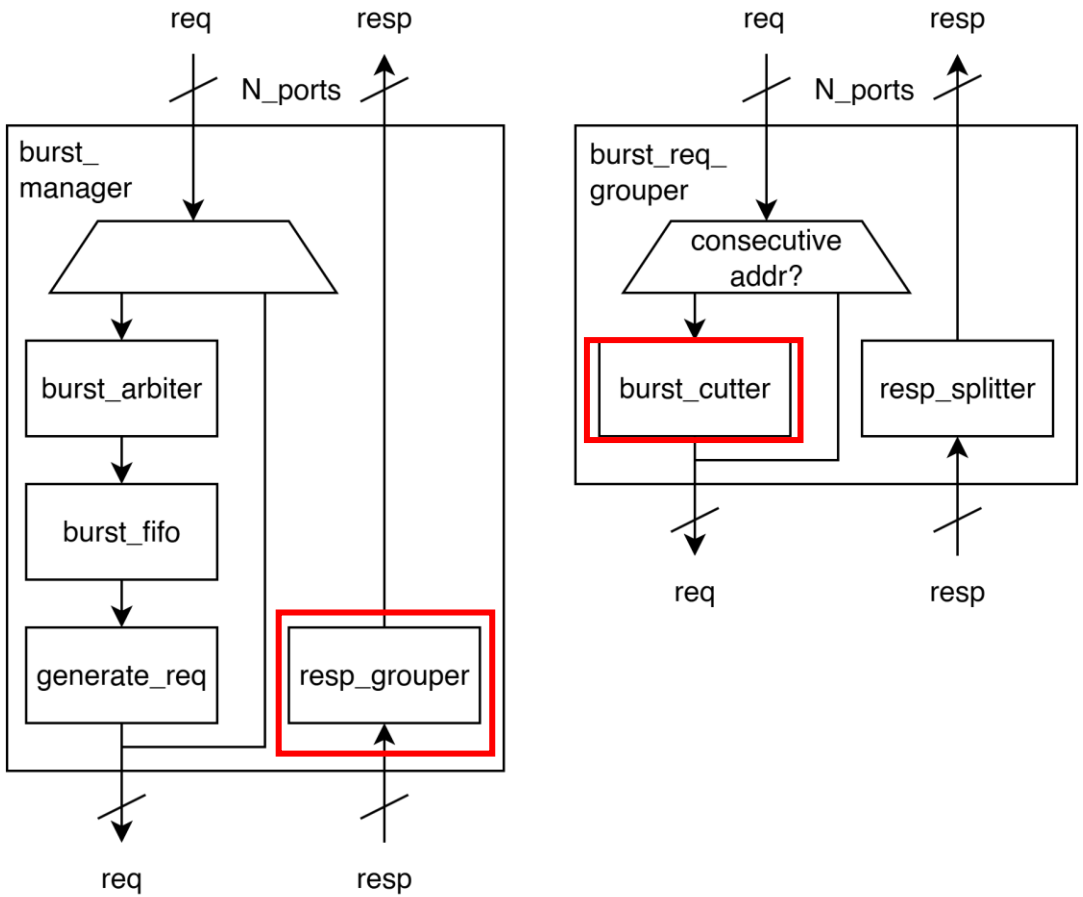
Integration in MemPool: Recap on bursts

- We can reduce arbitration also in the response path by grouping responses on the same response transactions

  (Done by the resp_grouper)

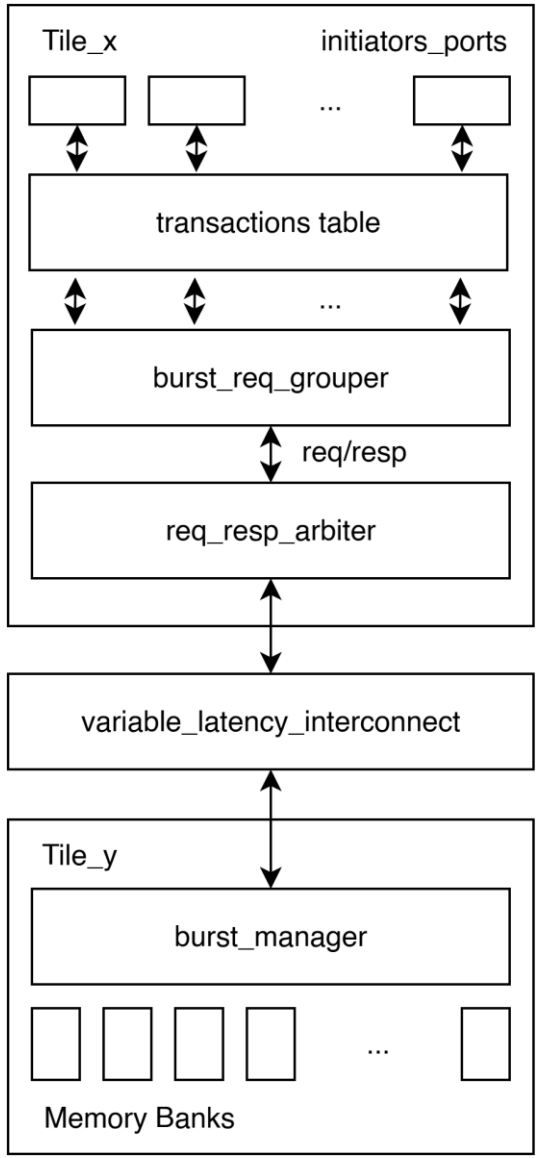
- When requests cross the boundary of a Tile two burst must be sent

  (Done by the burst_cutter)

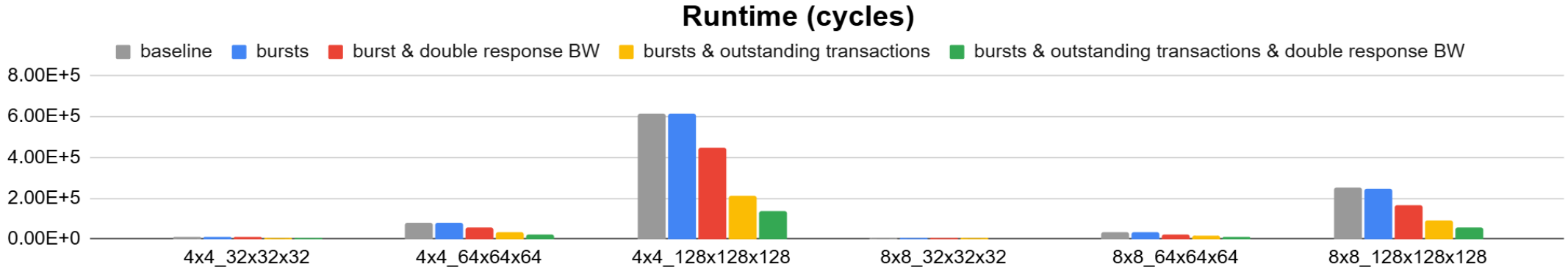


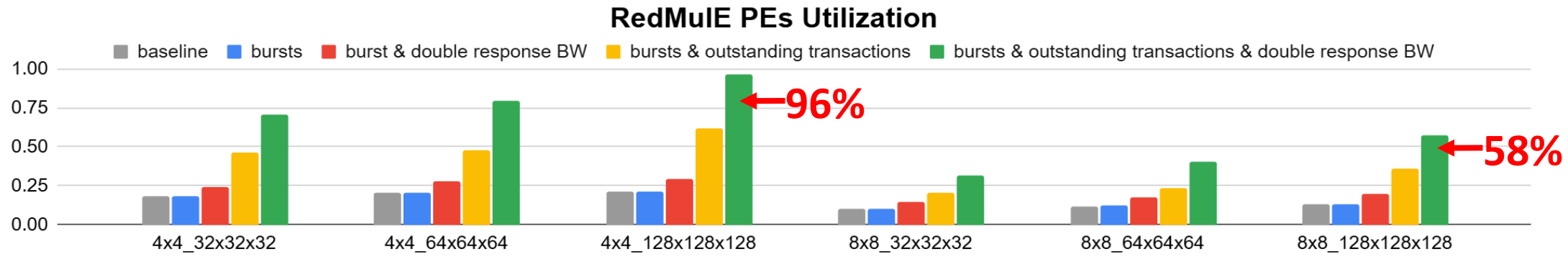
Integration in MemPool: transactions table

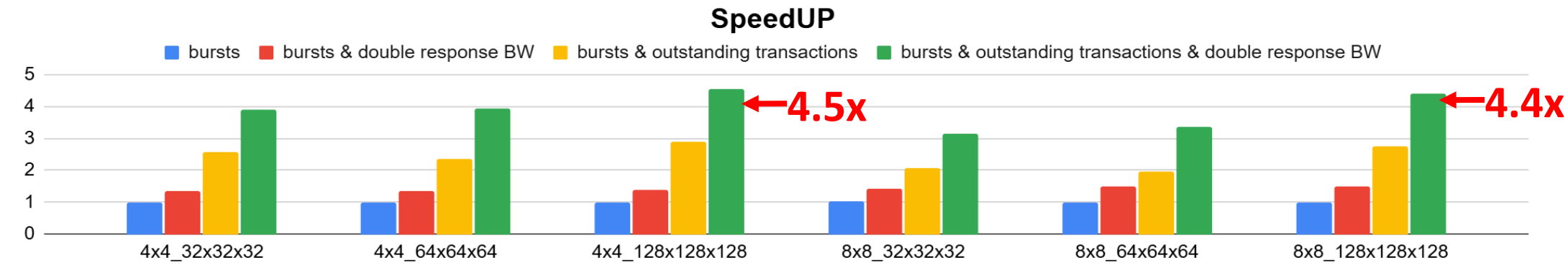
- Narrow responses come back out-of-order
- With outstanding transactions a transaction table reorders the narrow memory requests



Results (single RedMulE)



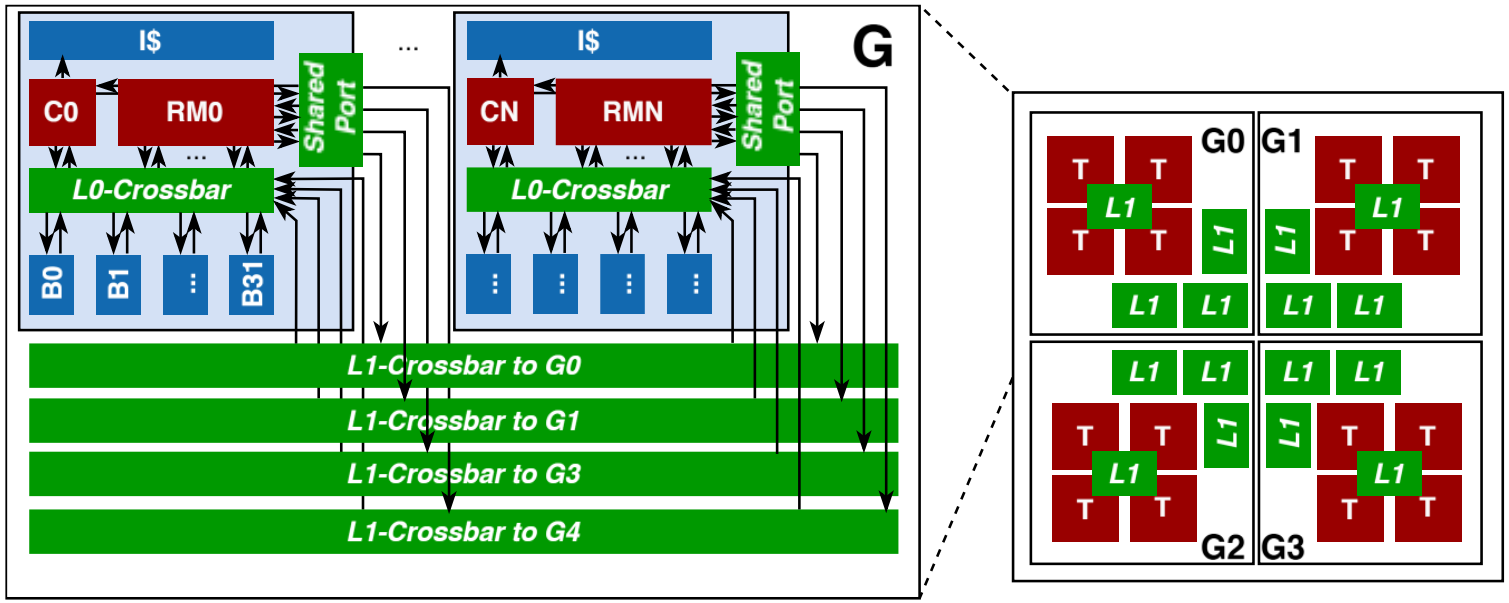




Next Steps

- **TensorPool:**
  - Optimize RedMulE to L1 connection → modify TeraPool interconnect for TensorPool-5 config.
  - TeraPool physical design in 7nm
  - TeraPool + RedMulE (TensorPool) physical design in 7nm
  - PPA on model microkernels and operators (combined RedMulE&Cores)
- **System Performance:**
  - Simulation speed is impaired by large design size → from RTL simulation to higher abstraction level, e.g. GVSoC
  - TensorPool GVSoC model developement
  - Data-Movement and end-end performance

Recap TensorPool-5



Best option is highlighted and increases the BW of RedMulE

FLOPS	NCores	NRM	NBank /Tile	NRM Tiles	LV1RM Ports	LV2RM Ports	NTiles	NG	NSGs	LEVEL 0		LEVEL 1		LEVEL 2	
										INxOUT	N	INxOUT	N	INxOUT	N
~~4096~~	~~0~~	~~16~~	~~512~~	~~1~~	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~	~~512x512~~	~~1~~	-	-	-	-
~~4096~~	~~0~~	~~16~~	~~32~~	~~8~~	~~1~~	~~0~~	~~0~~	~~1~~	~~0~~	~~64x64~~	~~8~~	~~8x8~~	~~1~~	-	-
~~4096~~	~~0~~	~~16~~	~~32~~	~~16~~	~~1~~	~~0~~	~~0~~	~~1~~	~~0~~	~~32x32~~	~~16~~	~~16x16~~	~~1~~	-	-
~~4096~~	~~0~~	~~16~~	~~32~~	~~16~~	~~2~~	~~0~~	~~0~~	~~1~~	~~0~~	~~32x32~~	~~16~~	~~32x32~~	~~1~~	-	-
4096	0	16	32	16	1	0	0	4	0	32x32	16	4x4	16	-	-
4096	0	16	32	16	8	0	0	4	0	32x32	16	32x32	16	-	-