Efficient Multi-Ported Memories for FPGAs

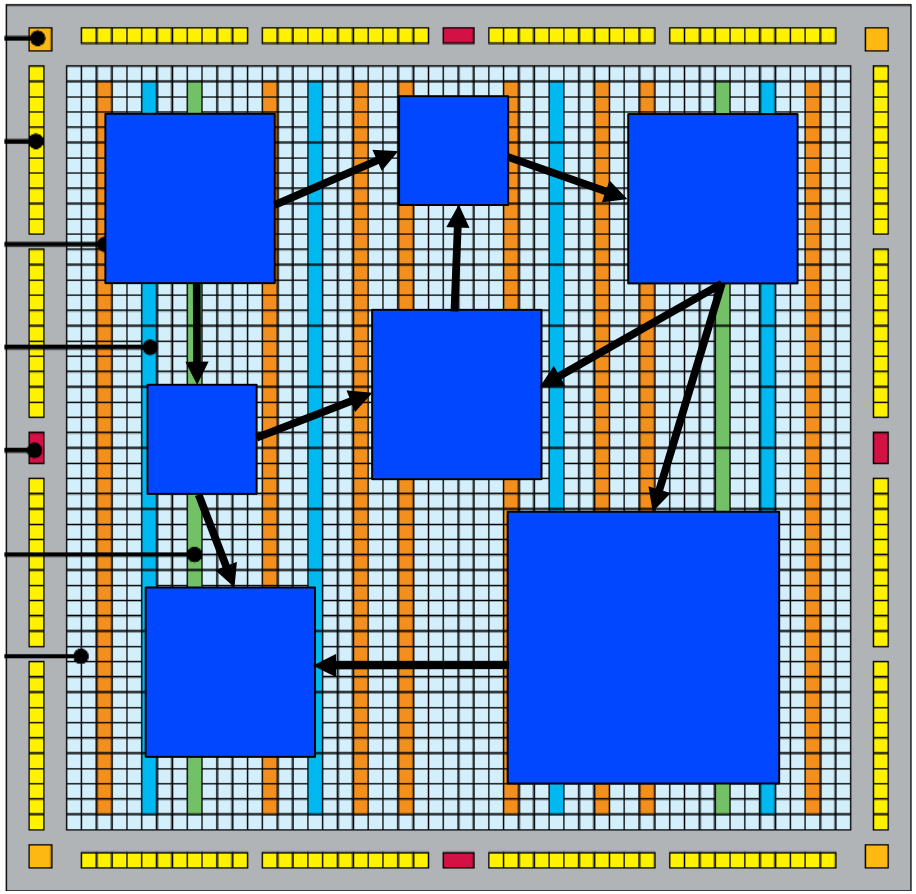
Eric LaForest
Greg Steffan

University of Toronto
Computer Engineering Research Group

February 22, 2010

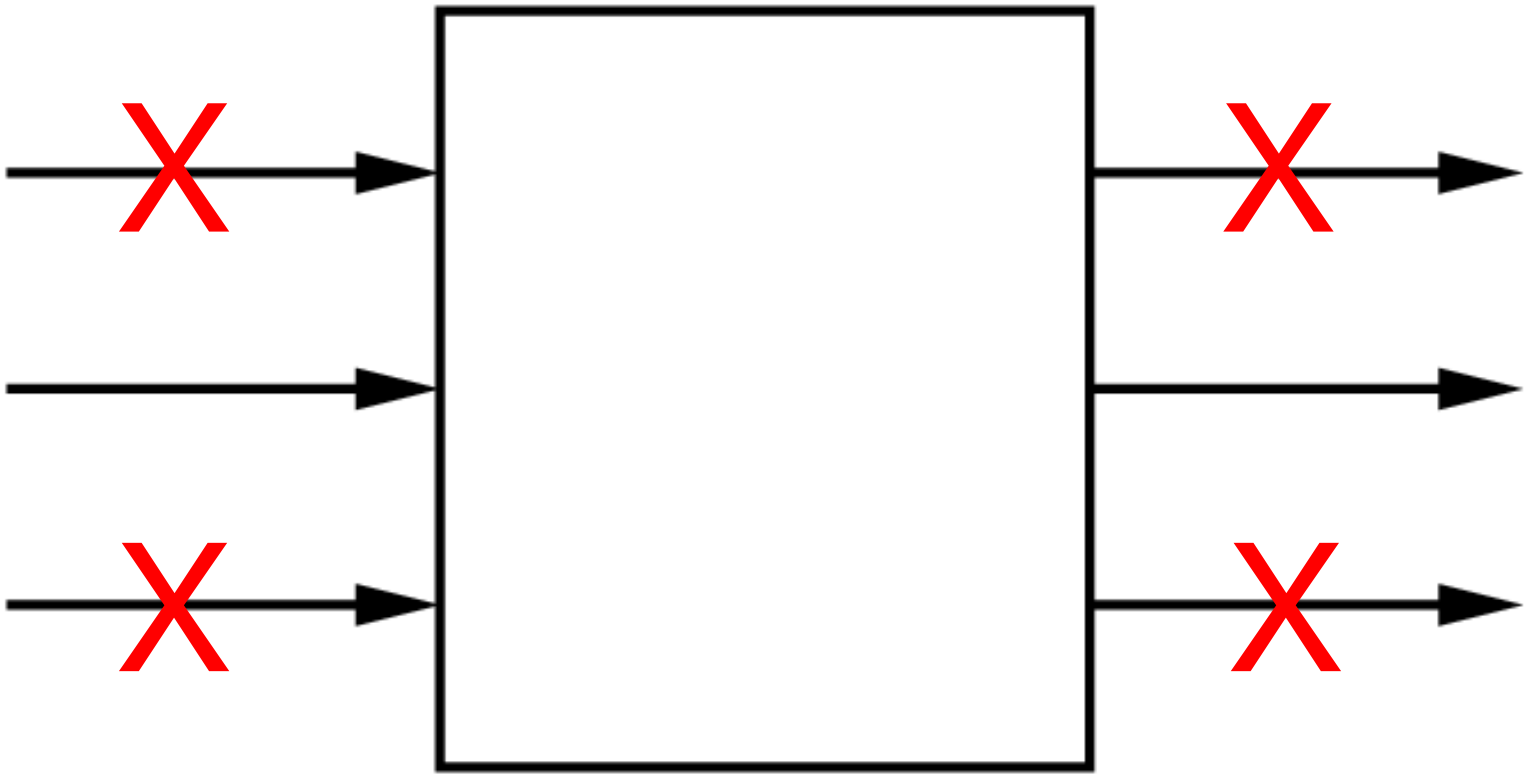
Parallelism in FPGAs

- Larger SoCs on FPGAs →Parallel Systems
- Parallel systems on FPGAs will need:
  - Queueing
  - Data sharing
  - Communication
  - Synchronization
- Boils down to:
  - FIFOs
  - Register files



We can do all these with multi-ported memories

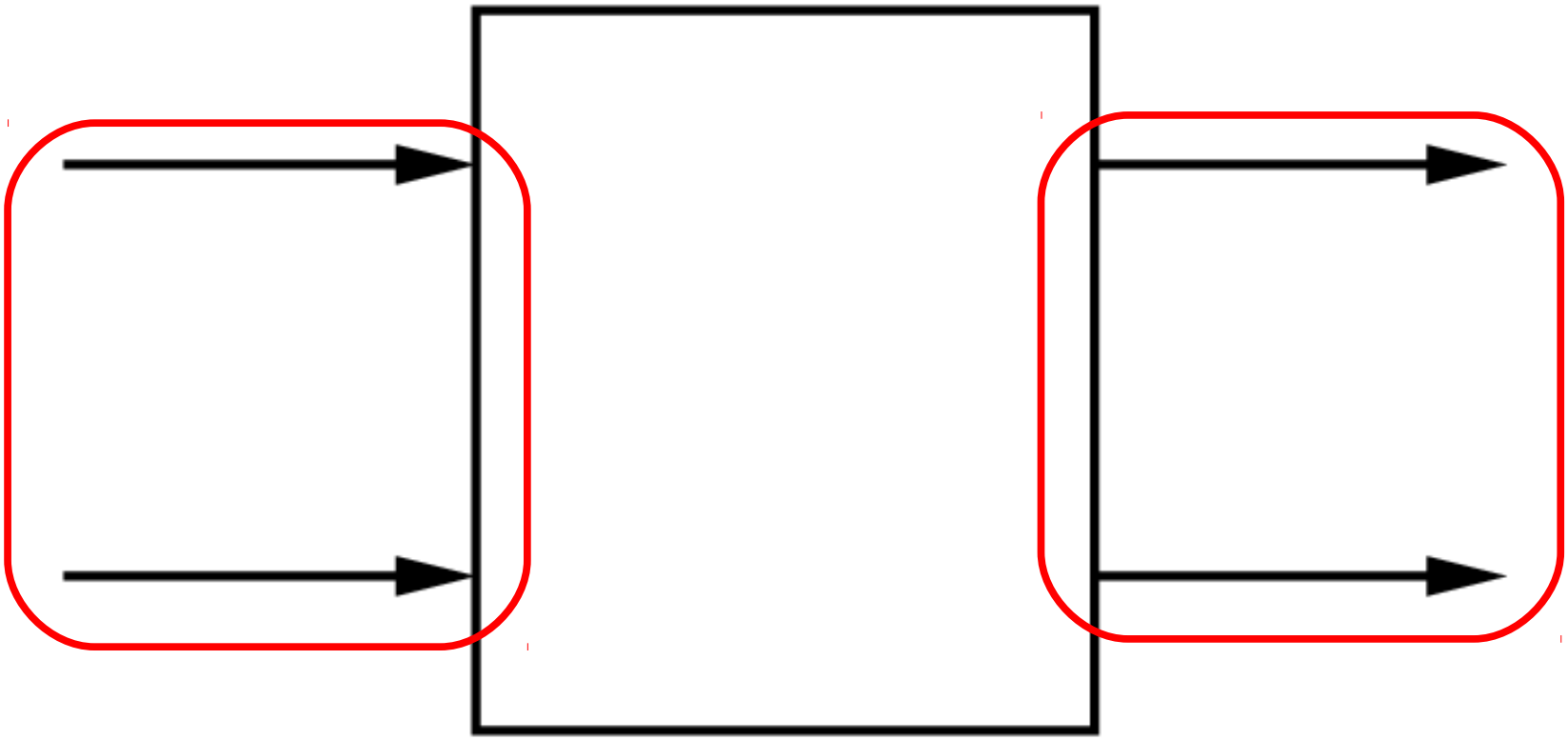
Multi-Ported Memory



Existing workarounds are ad-hoc, “roll-your-own”, and have limited parallelism.

Conventional Approaches

2W/2R Multi-Ported Memory



Doesn't exist on FPGAs

Altera used to have one (Mercury)

Stratix III Building Blocks

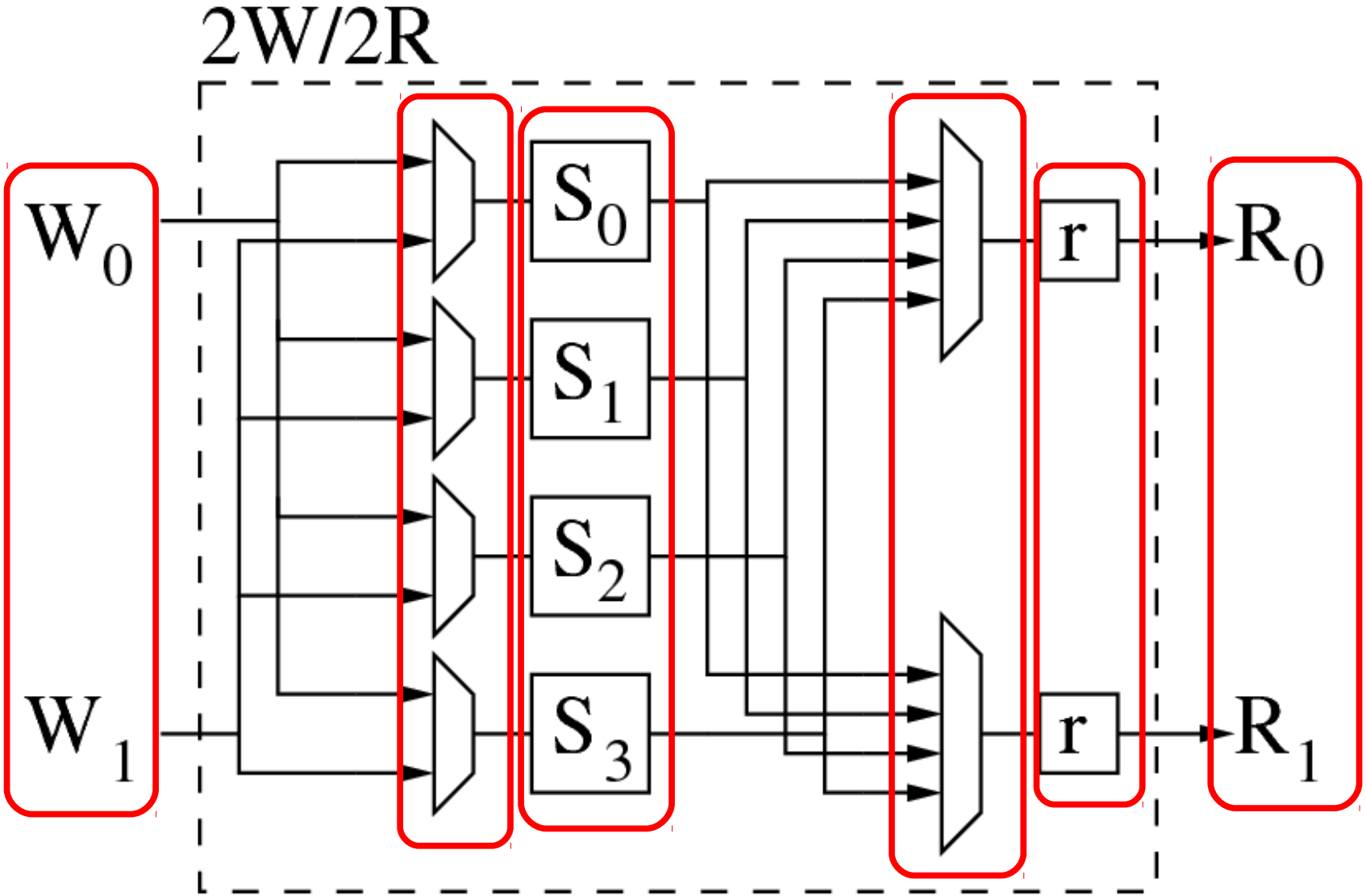
Adaptive Logic Modules
- Registers
- LUTs
- Adders

Flexible, but slow

Block RAMs
- M9K (eg: 32 x 256)
- M144K (eg: 32 x 4098)

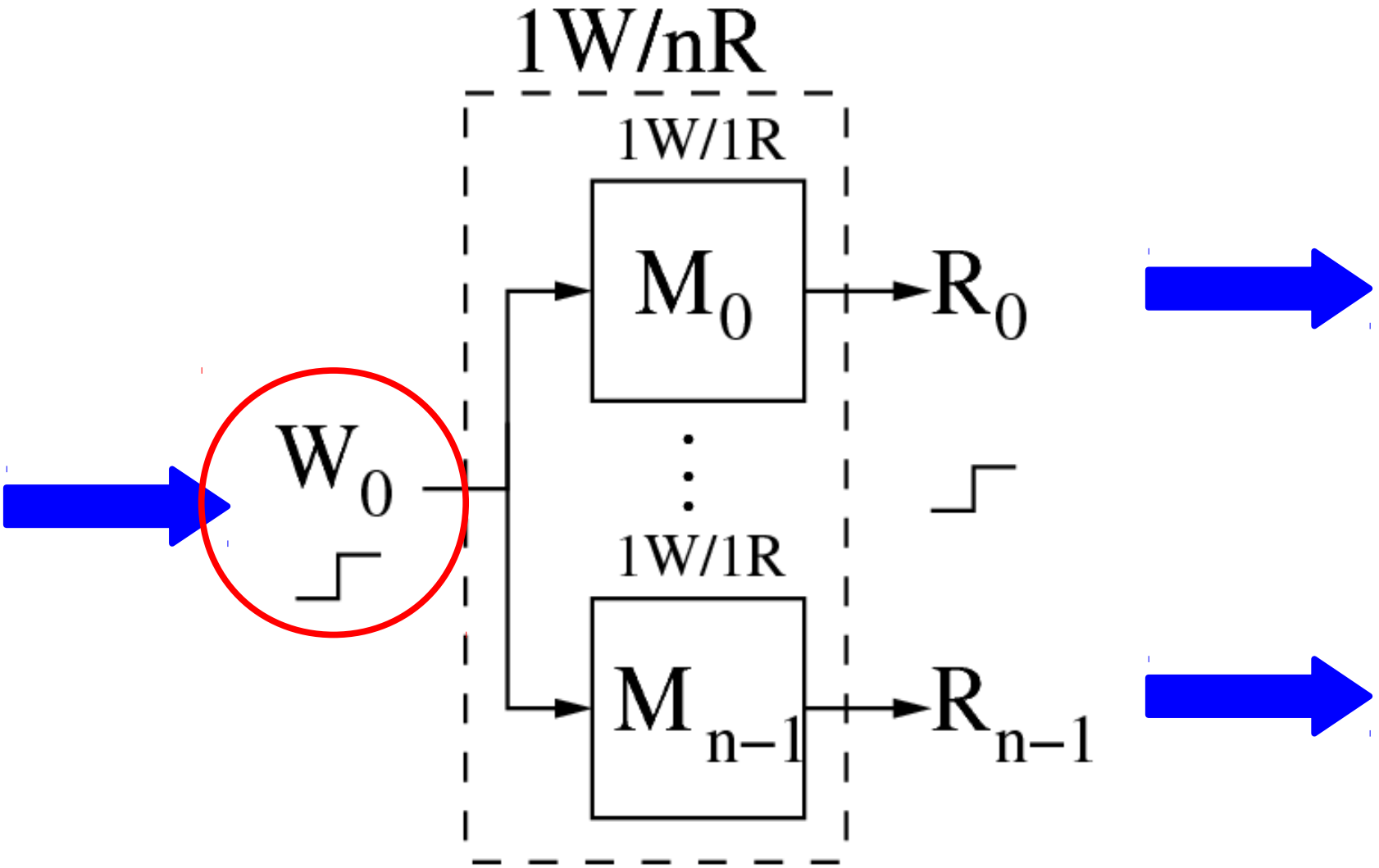
Fast, but inflexible

2W/2R Pure-ALM



Scales very poorly with memory depth

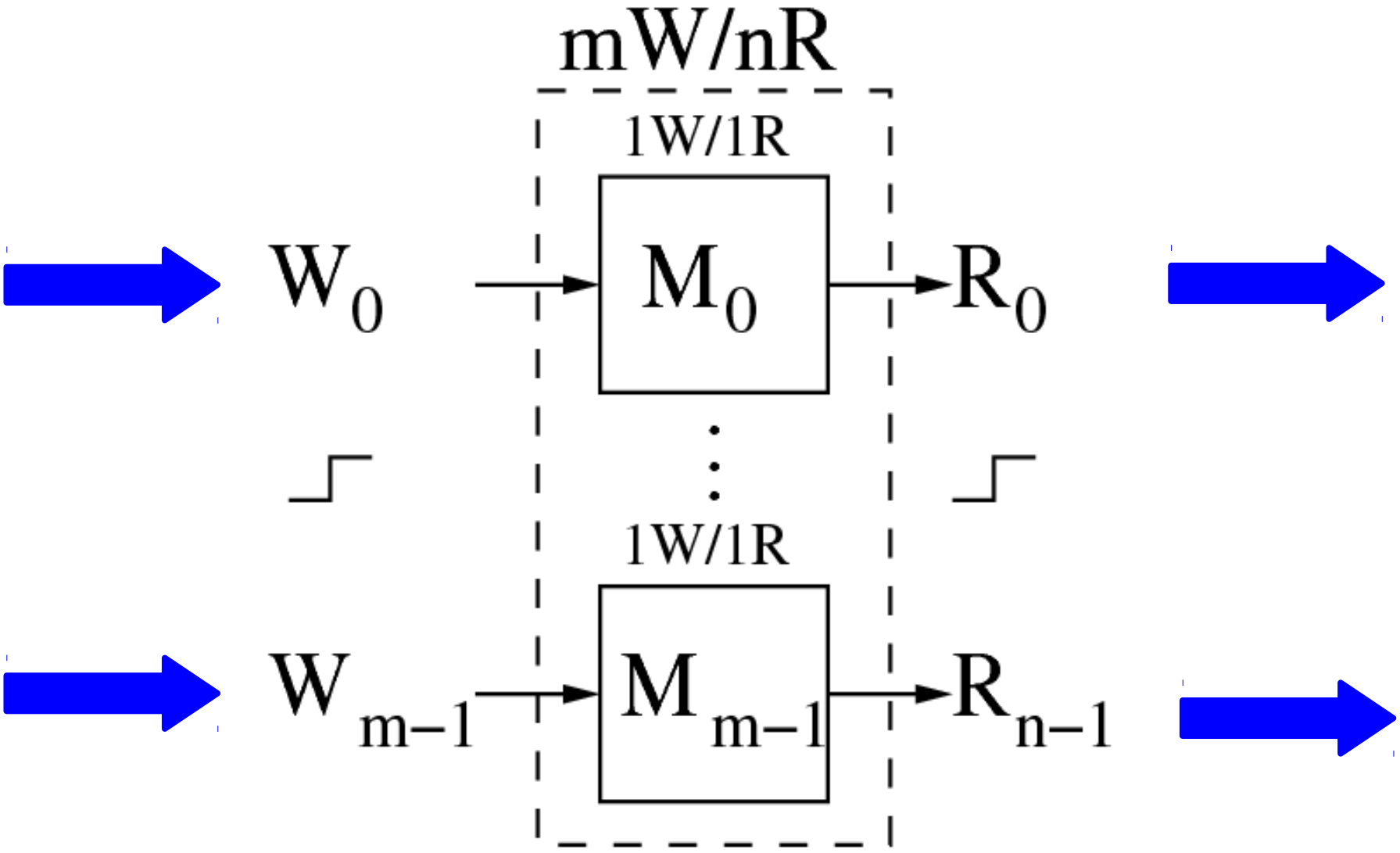
1W/nR Replication



Only one write port

Multiple read ports

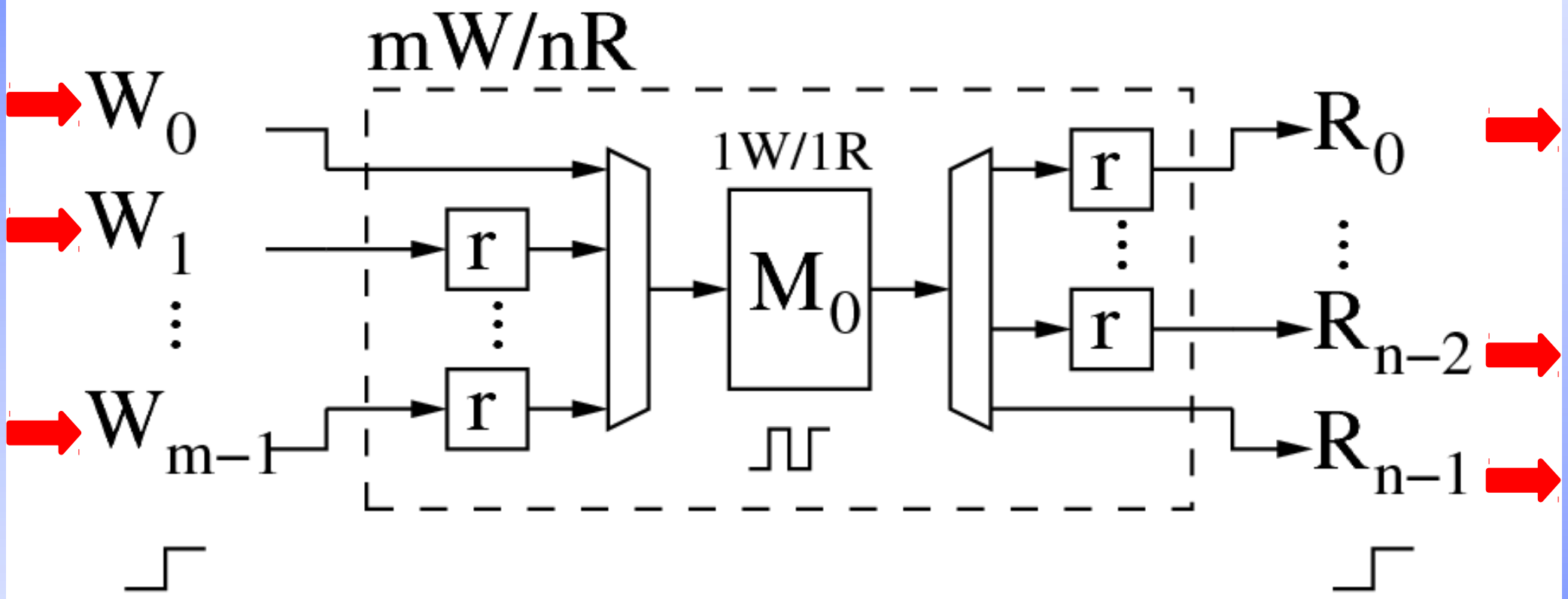
mW/nR Banking



Multiple write ports

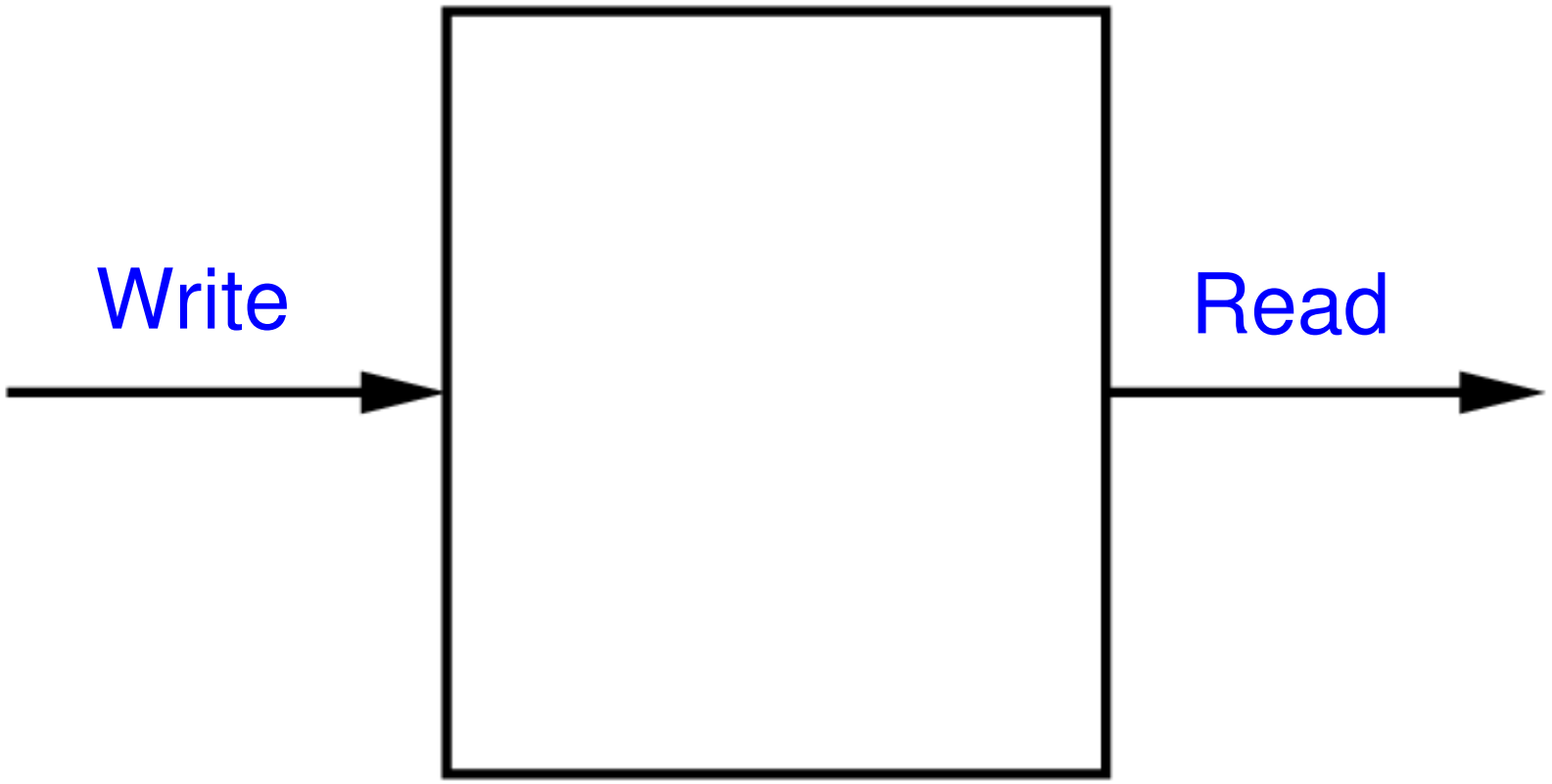
Fragmented data

mW/nR “Multipumping”

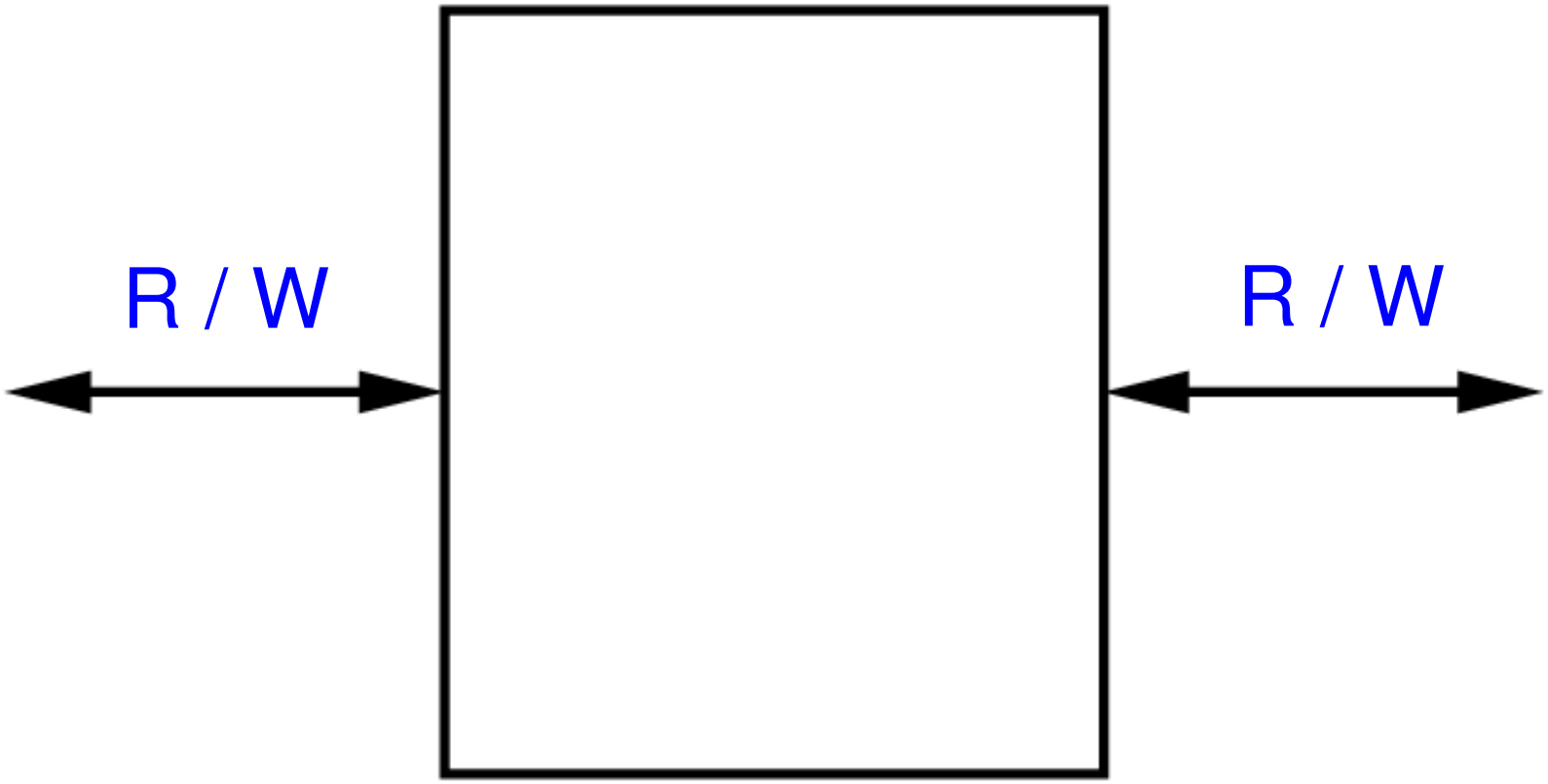


- Multiple read/write ports
- No fragmentation
- Divides clock speed
- Read/write ordering

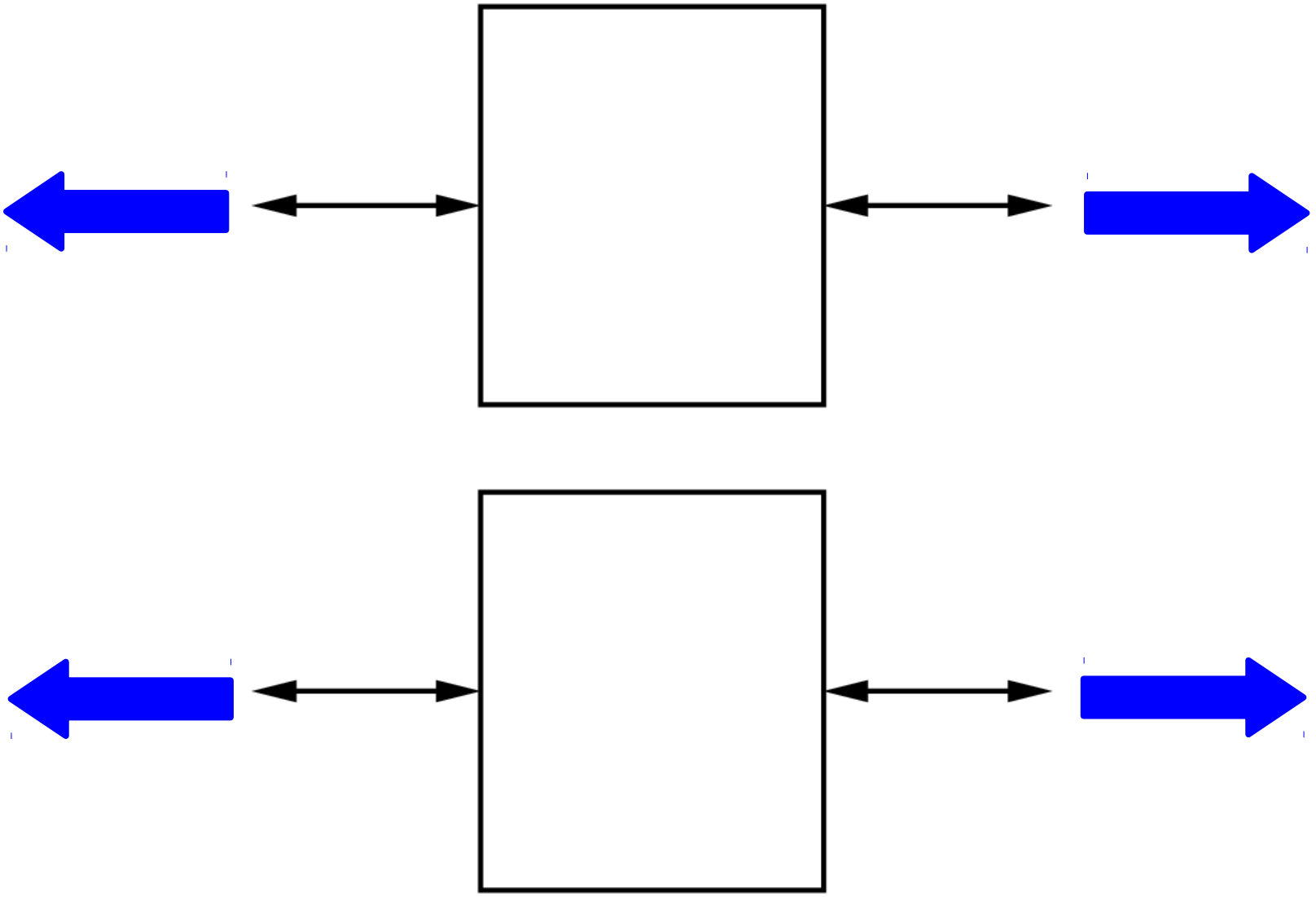
Block RAMs: Simple Dual Port



Block RAMs: True Dual Port

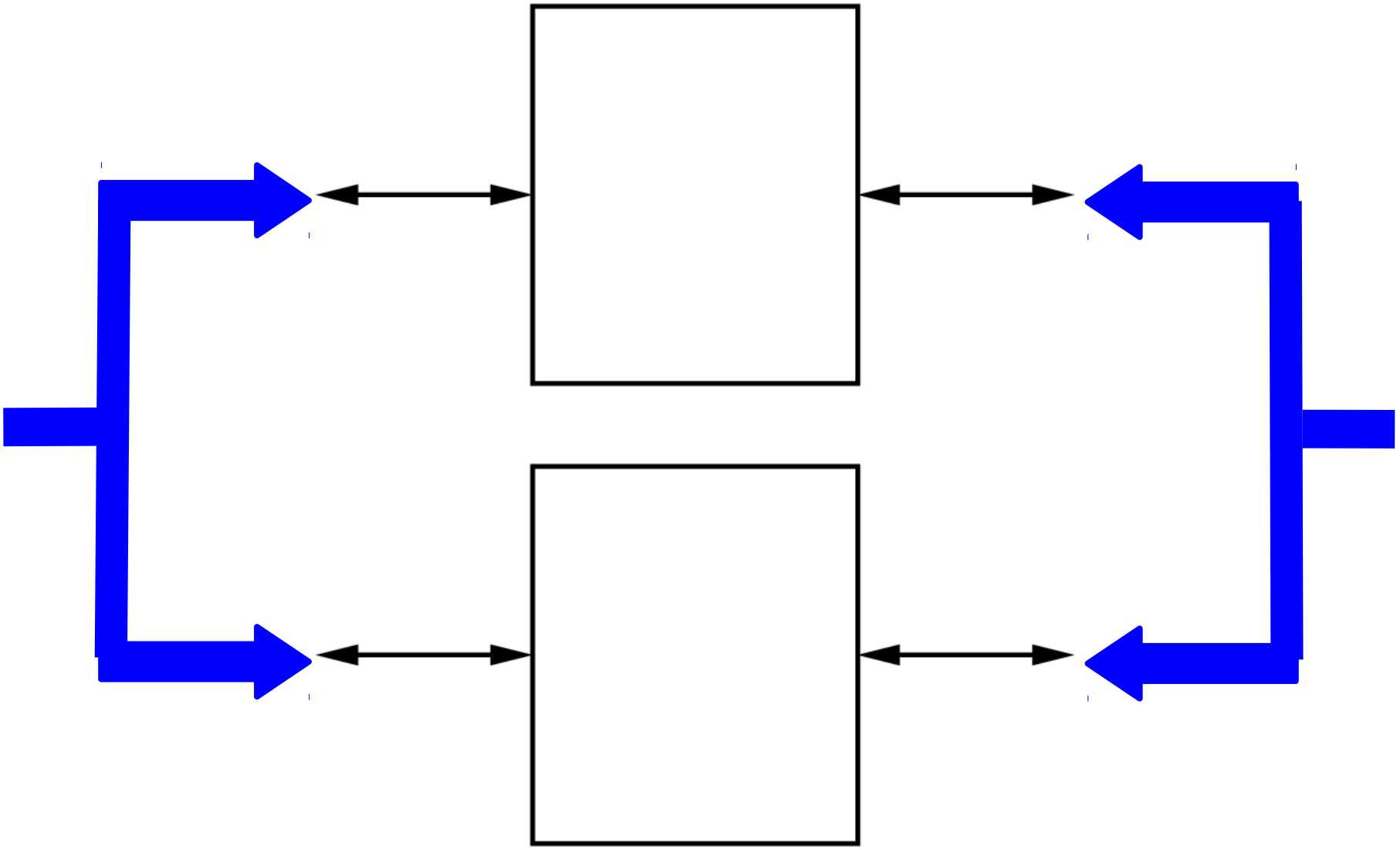


“Pure Multipumping”



Read as banked memory (multiple reads)

“Pure Multipumping”



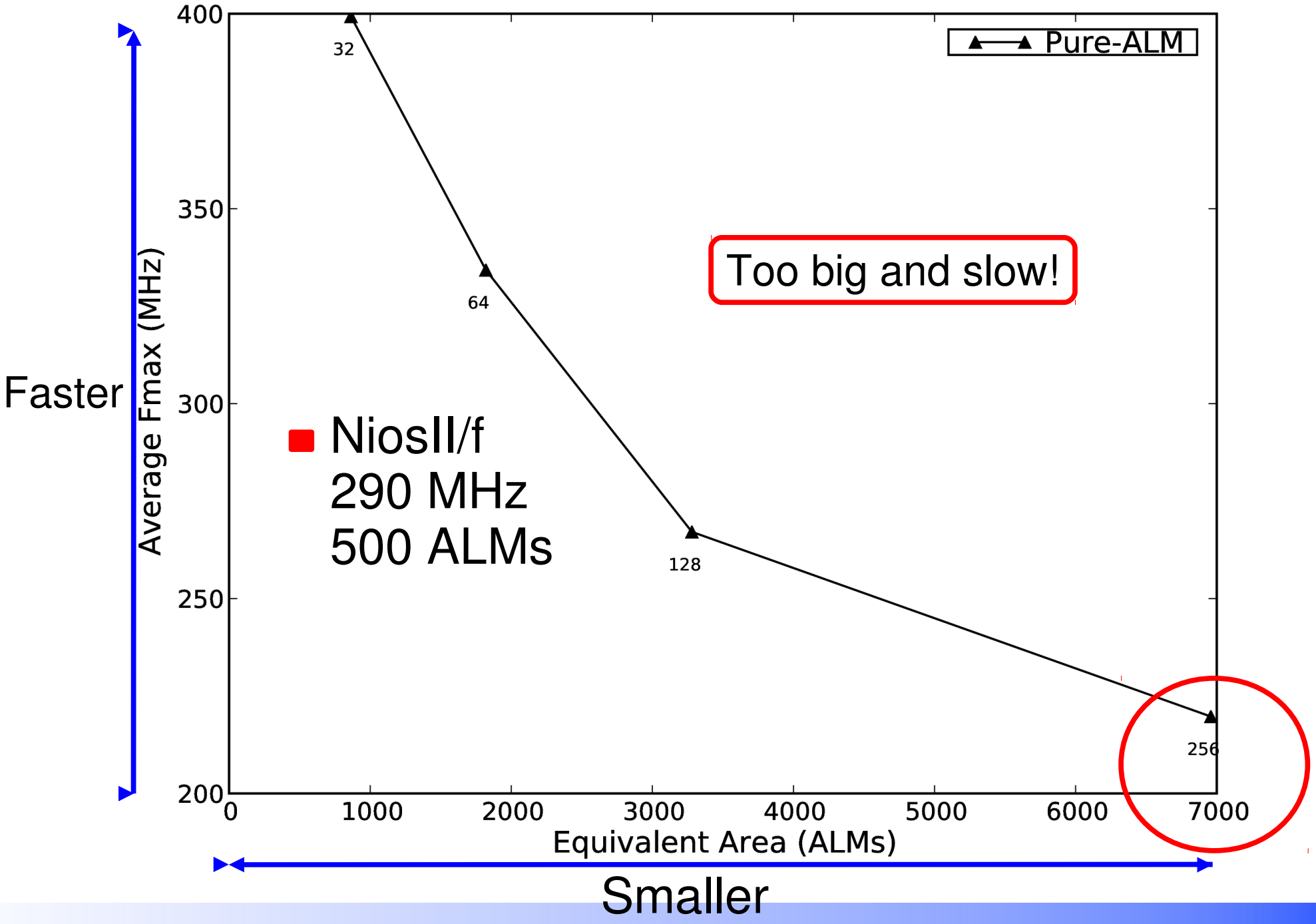
Write as replicated memory (avoids fragmentation)

Methodology

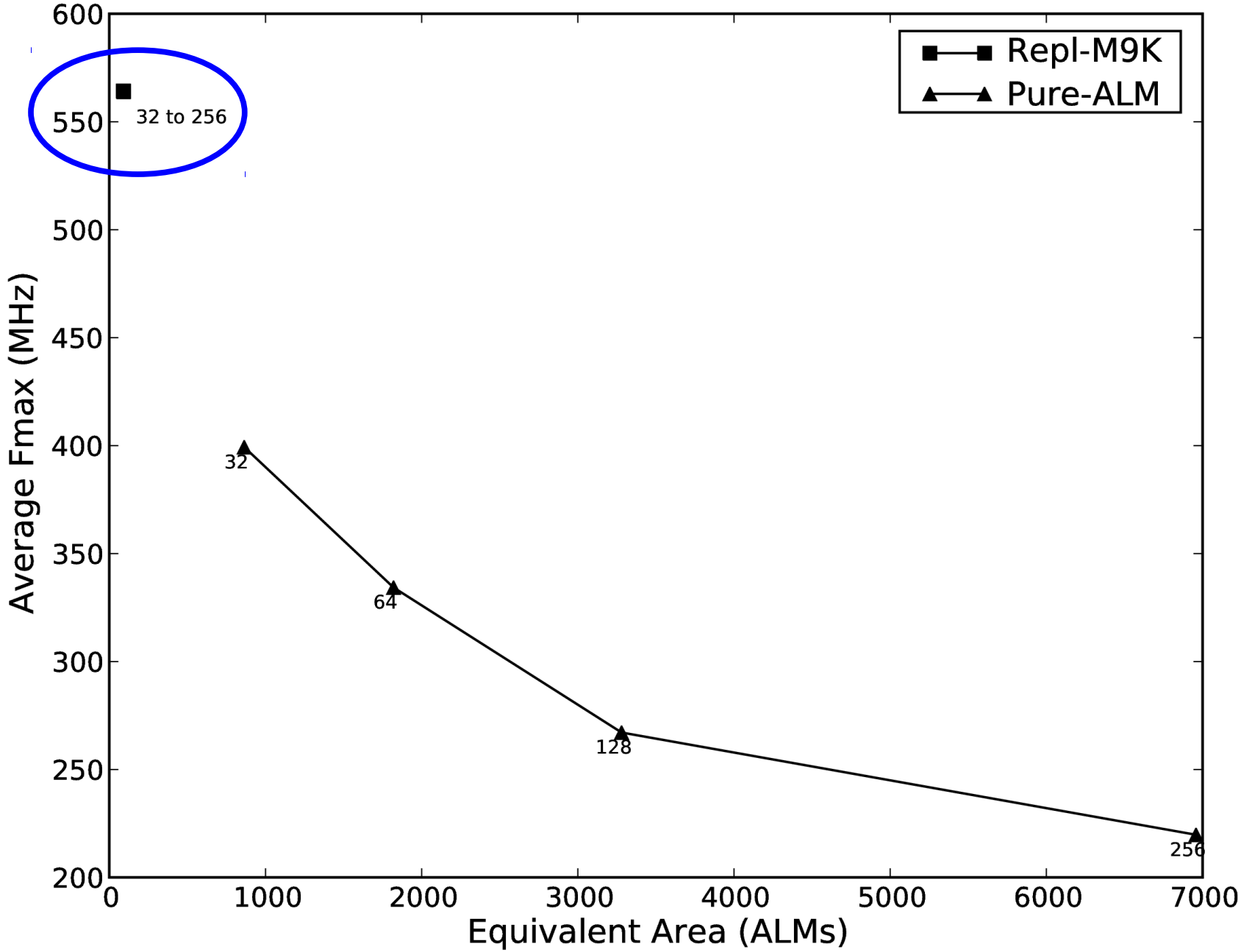
- Generate design variations over space
  - Vary # of ports, depth, type of memories
    - 1W/2R to 8W/16R
    - 2 to 256 elements deep
    - Pure-ALM, M9K, MLAB, Multipumped
  - Wrap in testbench for timing and correctness
- Target Quartus 9.0 to Stratix III
  - No synthesis optimizations for speed or area
  - Standard P&R effort (speed, avg. over 10 runs)
- Measure area as *Total Equivalent Area*
  - Expresses area in a single unit (ALMs)

Conventional Multi-Porting Performance

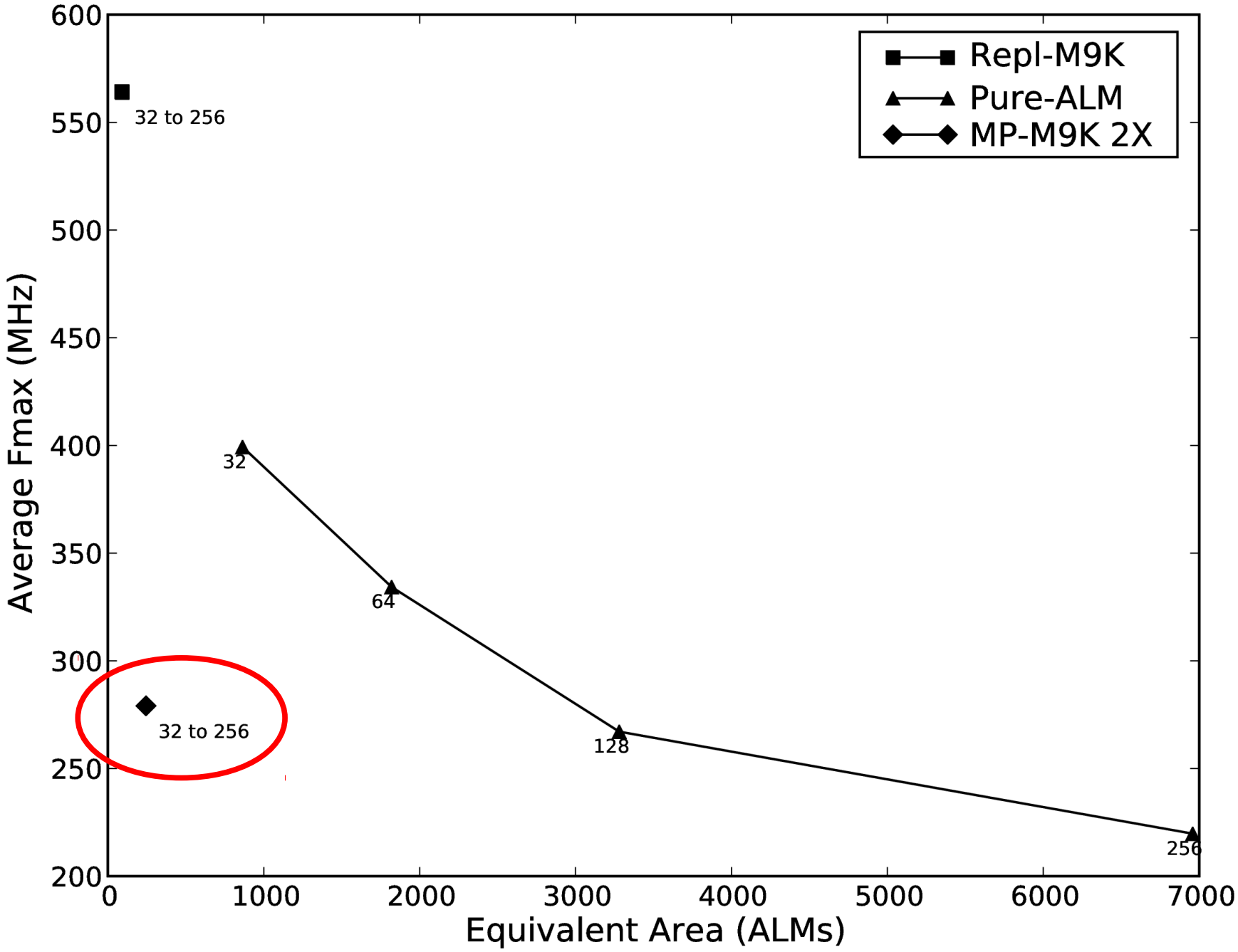
1W/2R Pure-ALM Area vs. Speed



1W/2R Replicated vs. Pure-ALM

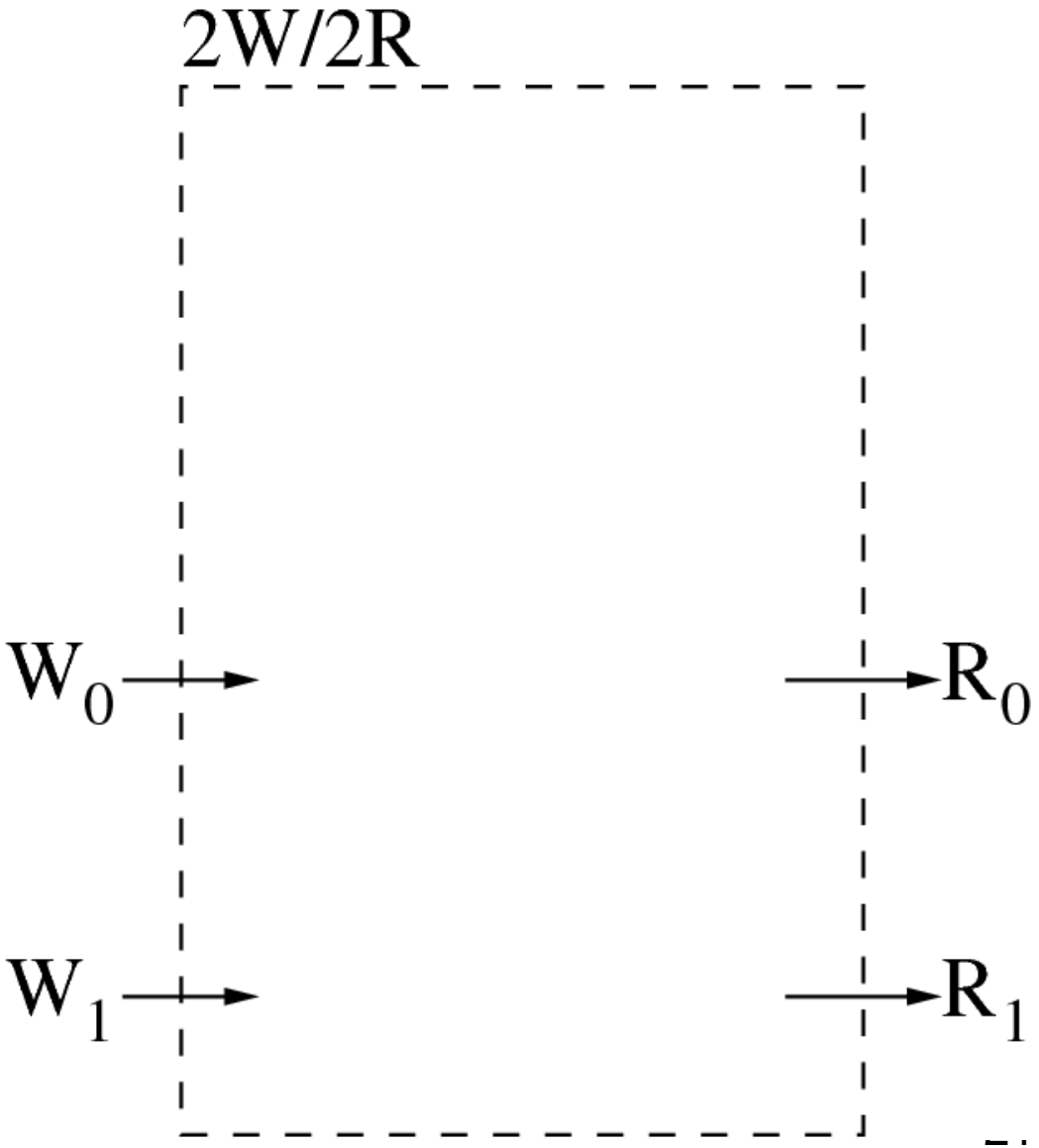


1W/2R “Pure Multipumping”



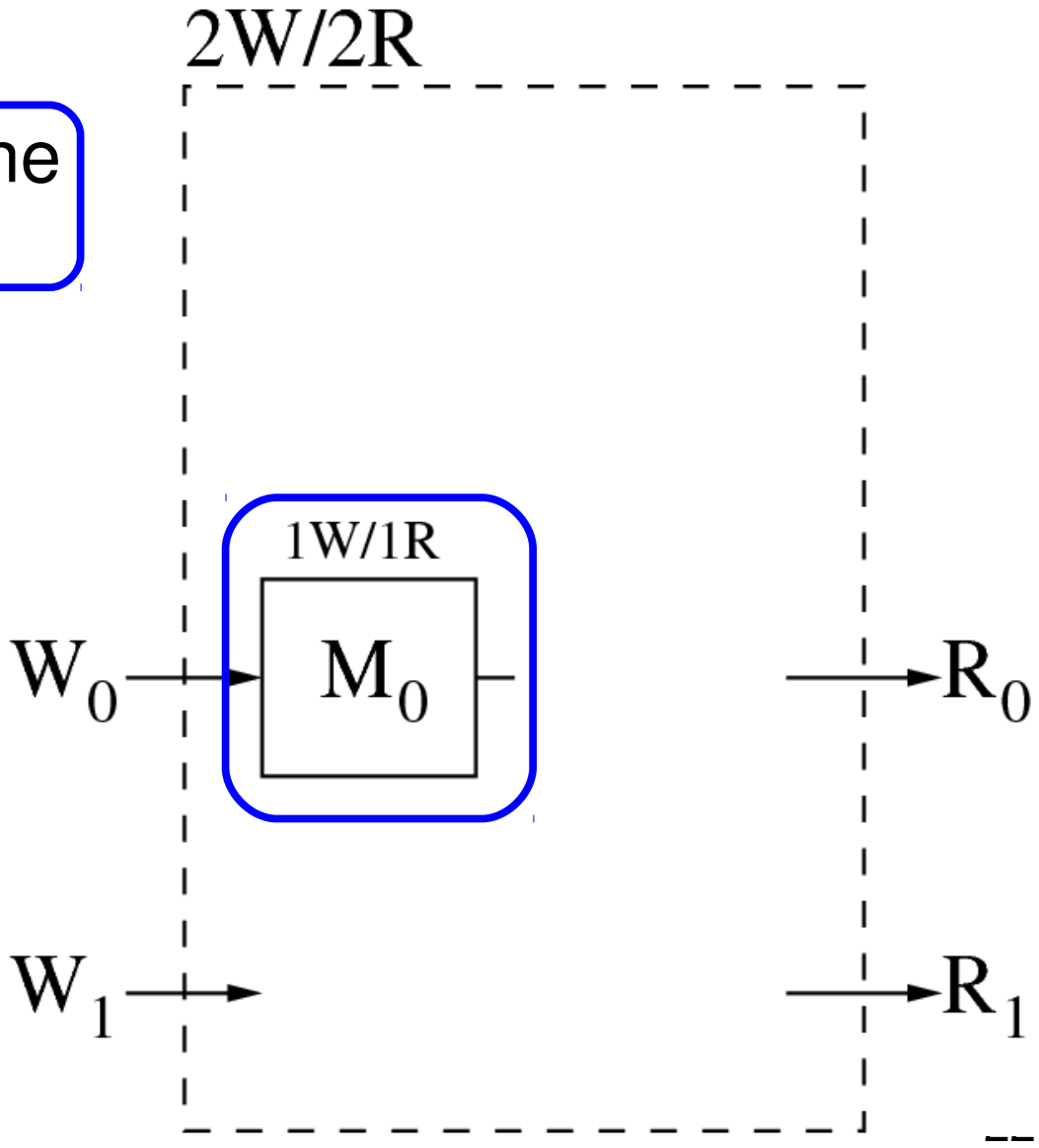
LVT-Based Multi-Ported Memories

LVT-Based Memory



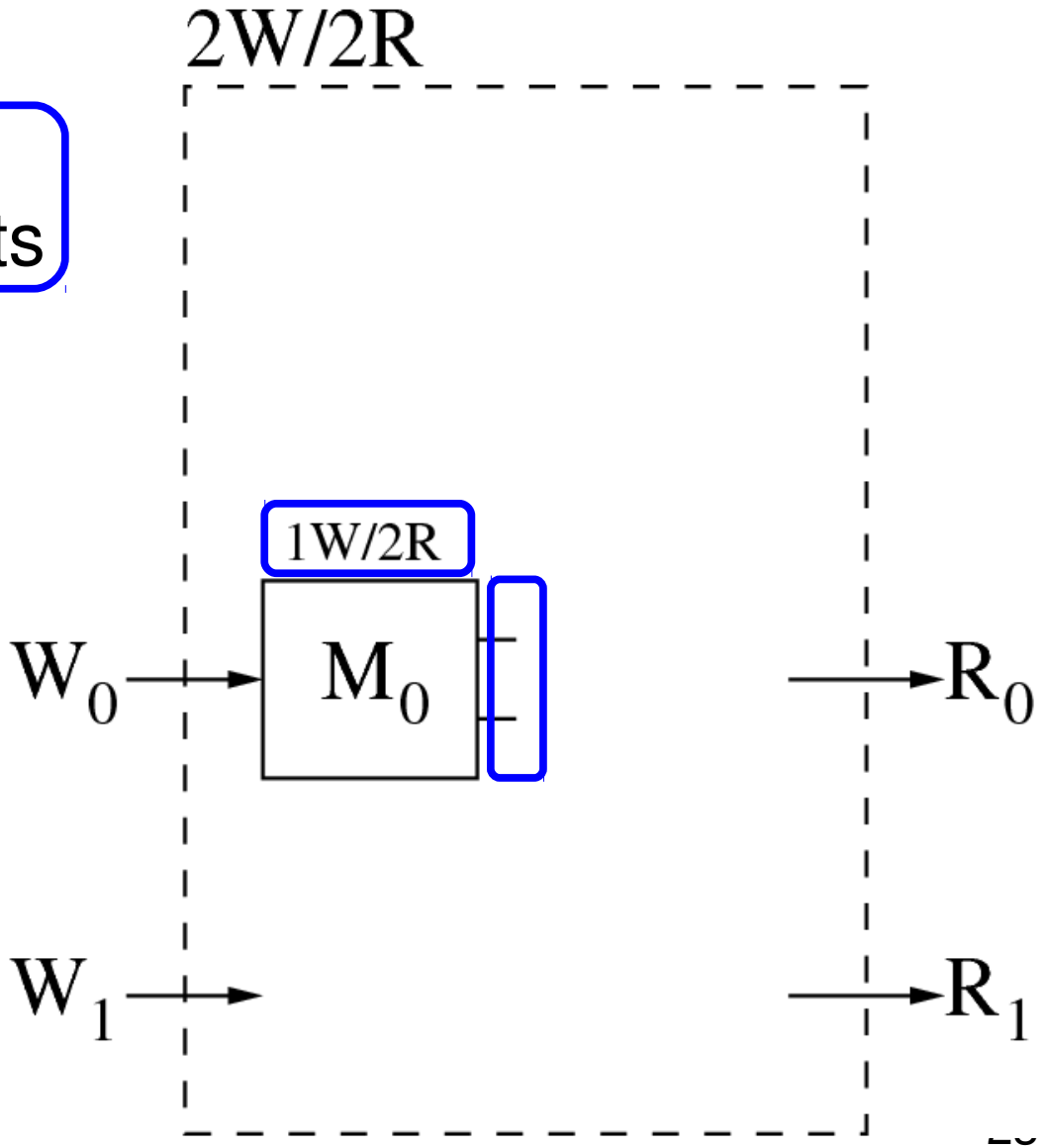
LVT-Based Memory

Begin with one block RAM



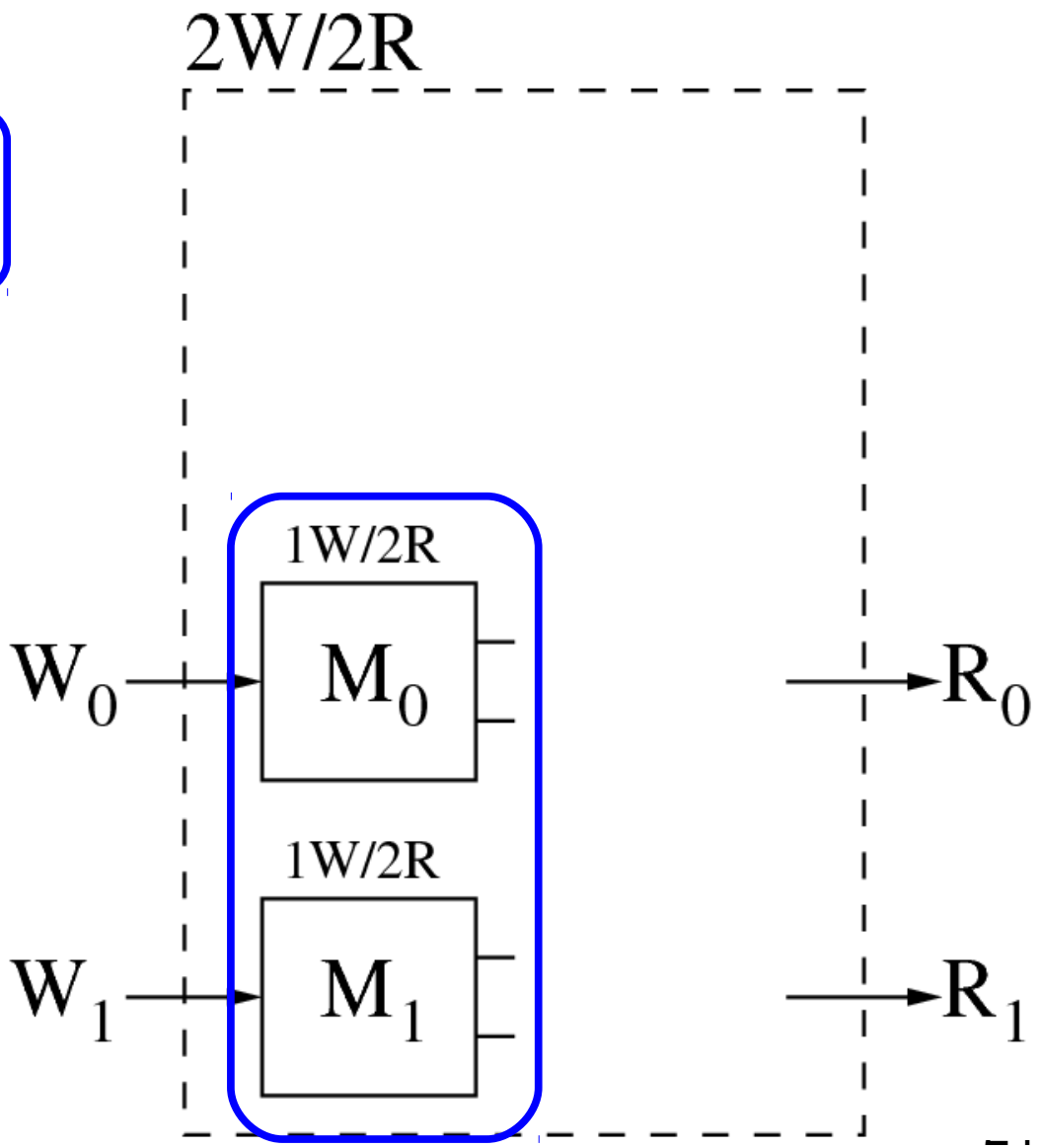
LVT-Based Memory

Replicate for two read ports



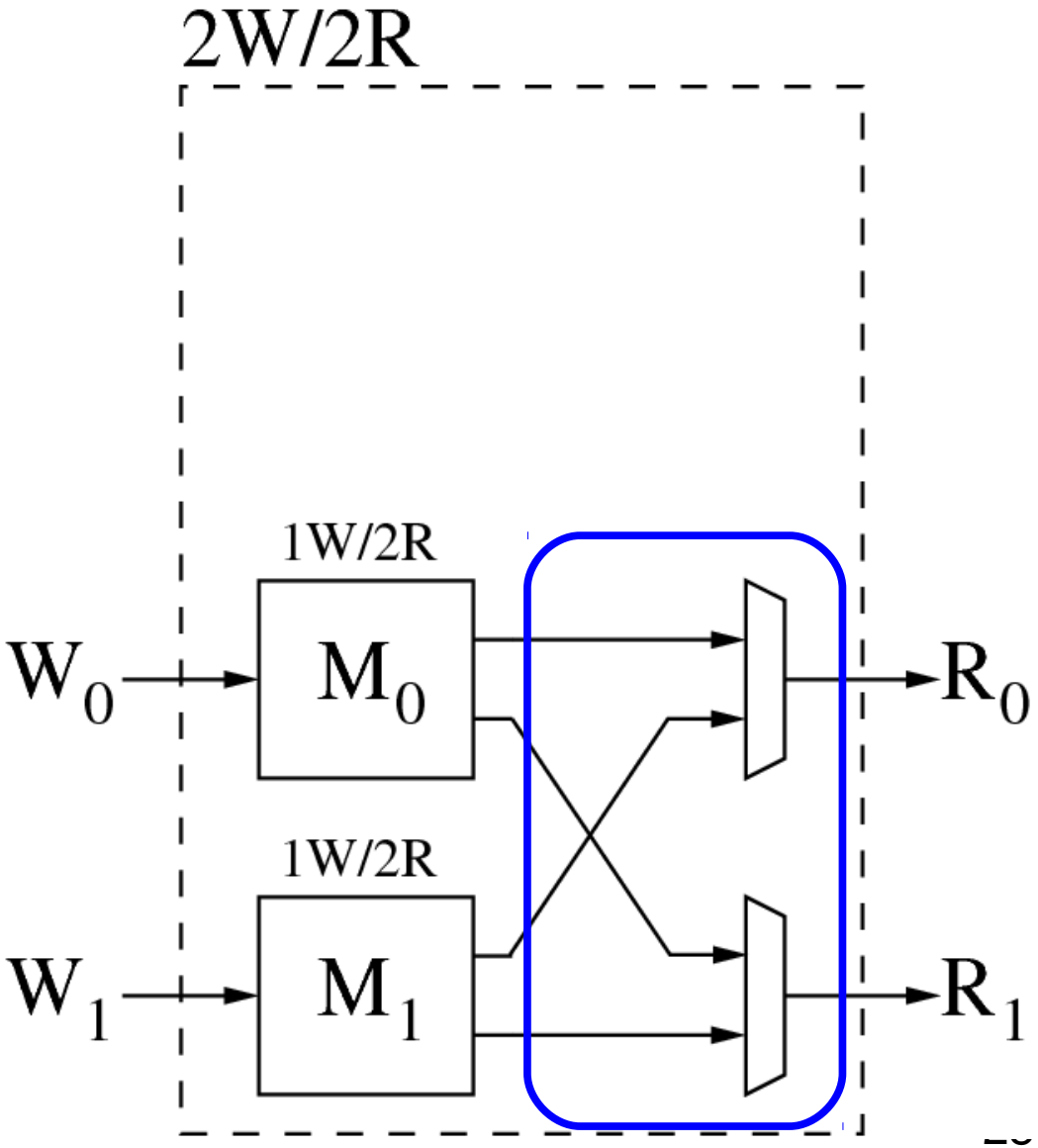
LVT-Based Memory

Bank for two write ports

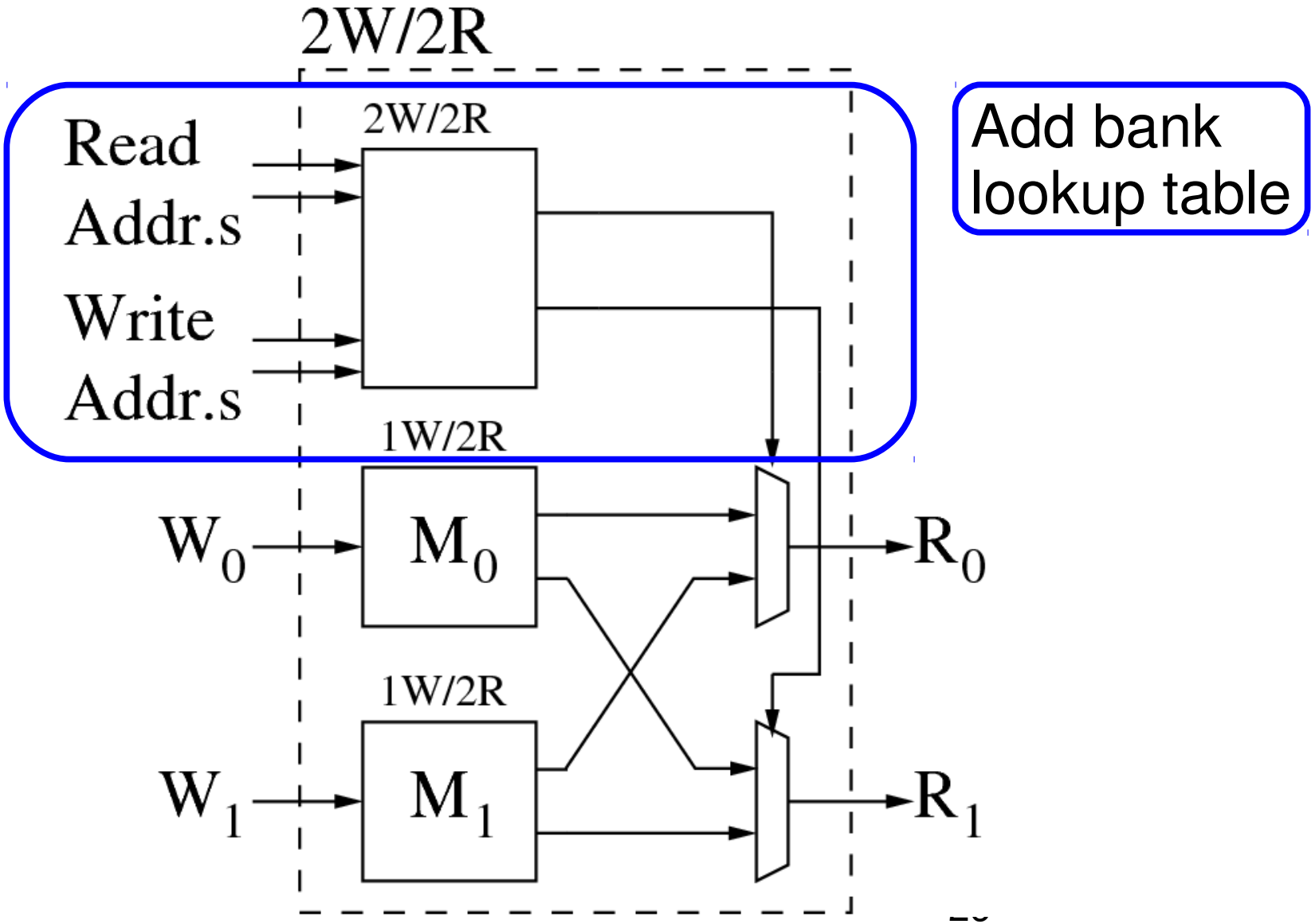


LVT-Based Memory

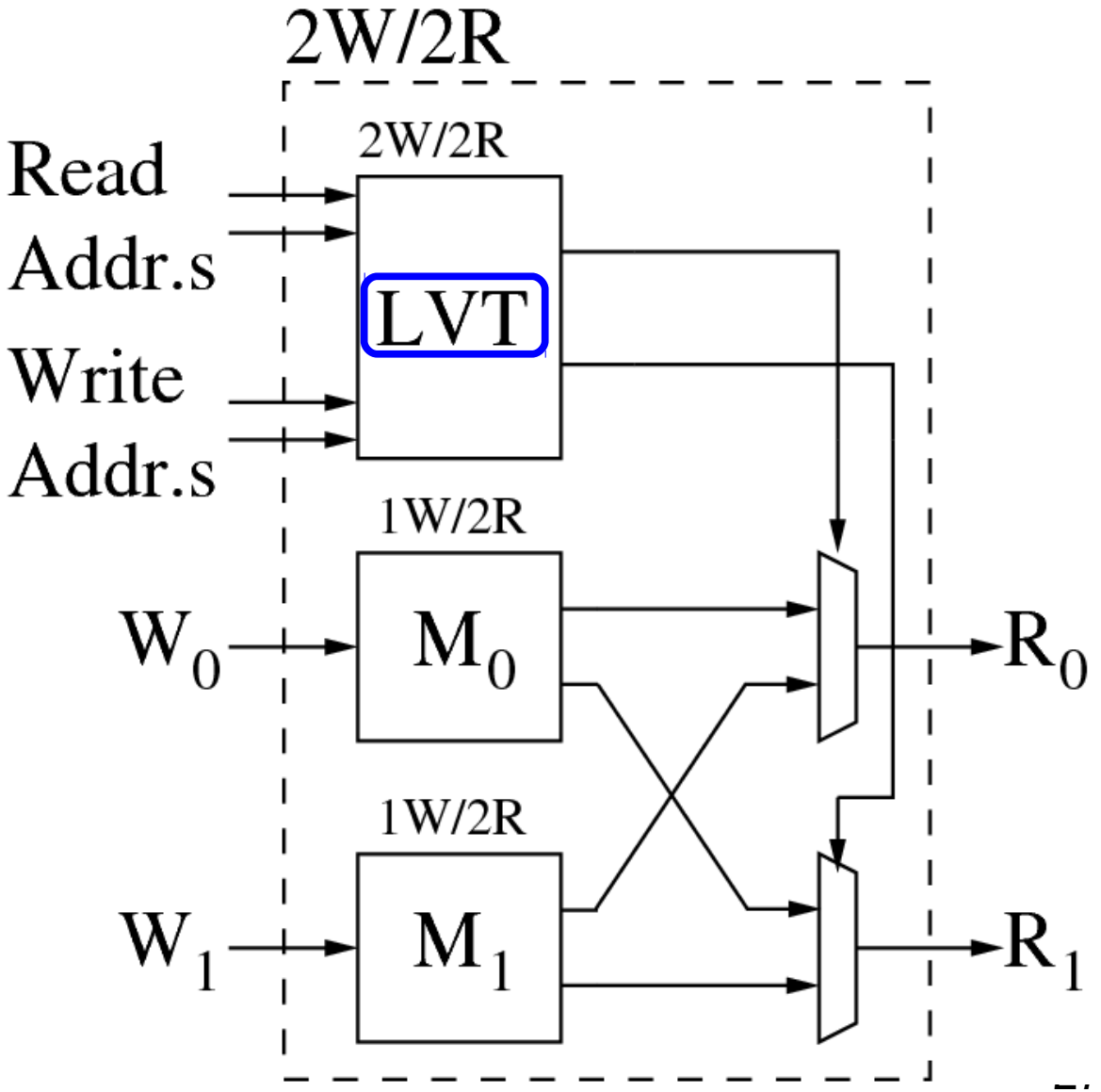
Select bank to read from



LVT-Based Memory

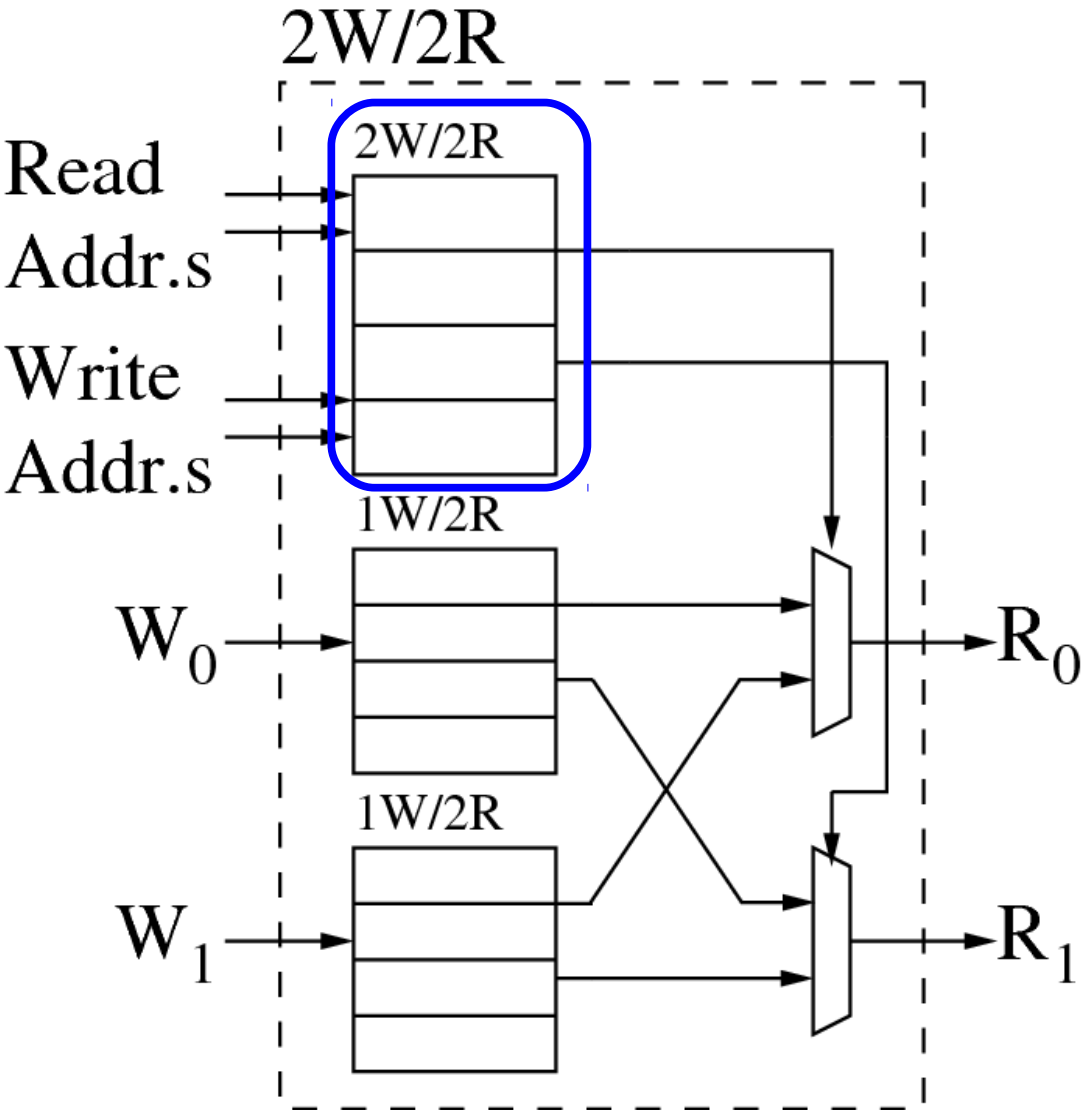


LVT-Based Memory



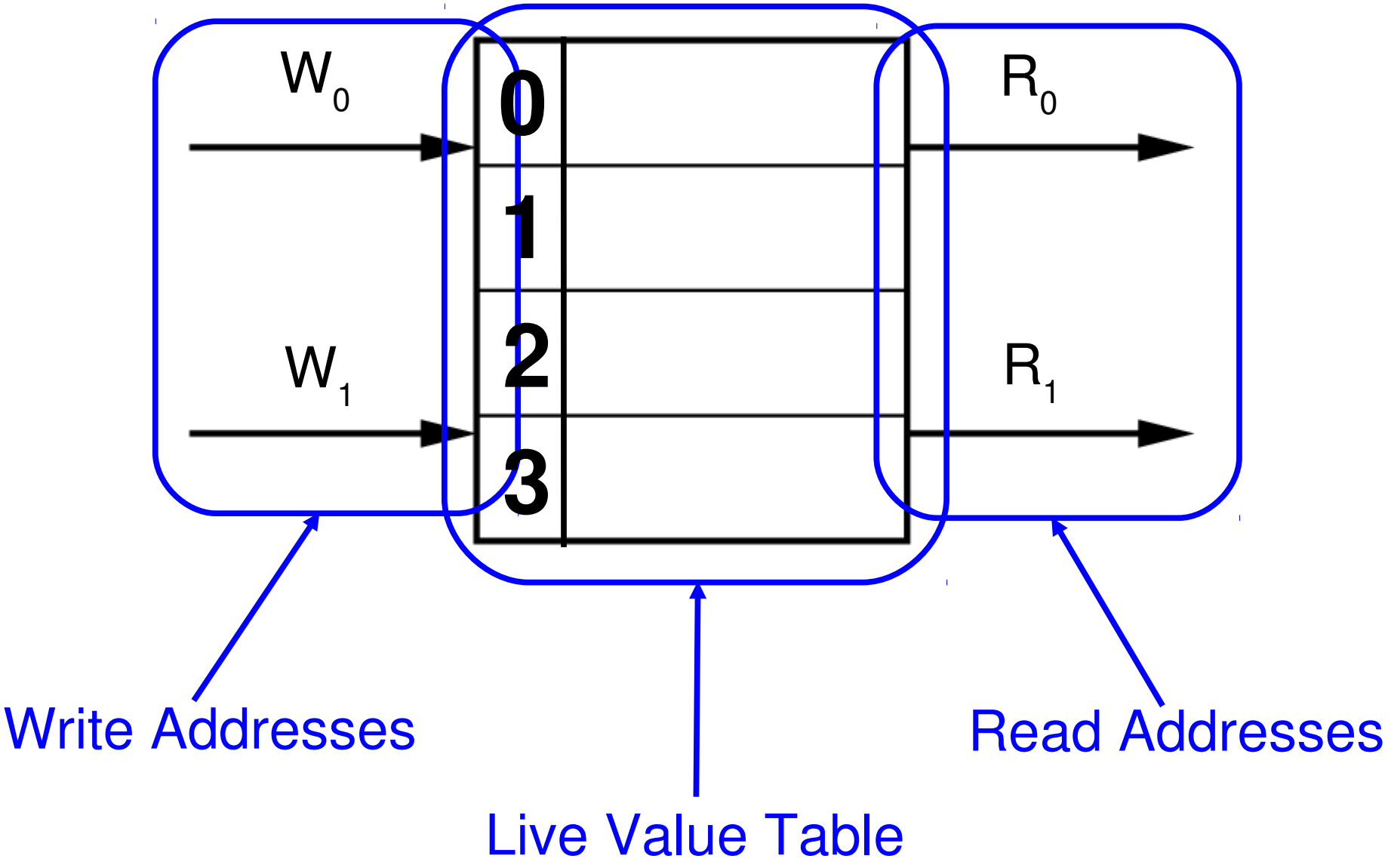
Live Value Table Operation

LVT Operation

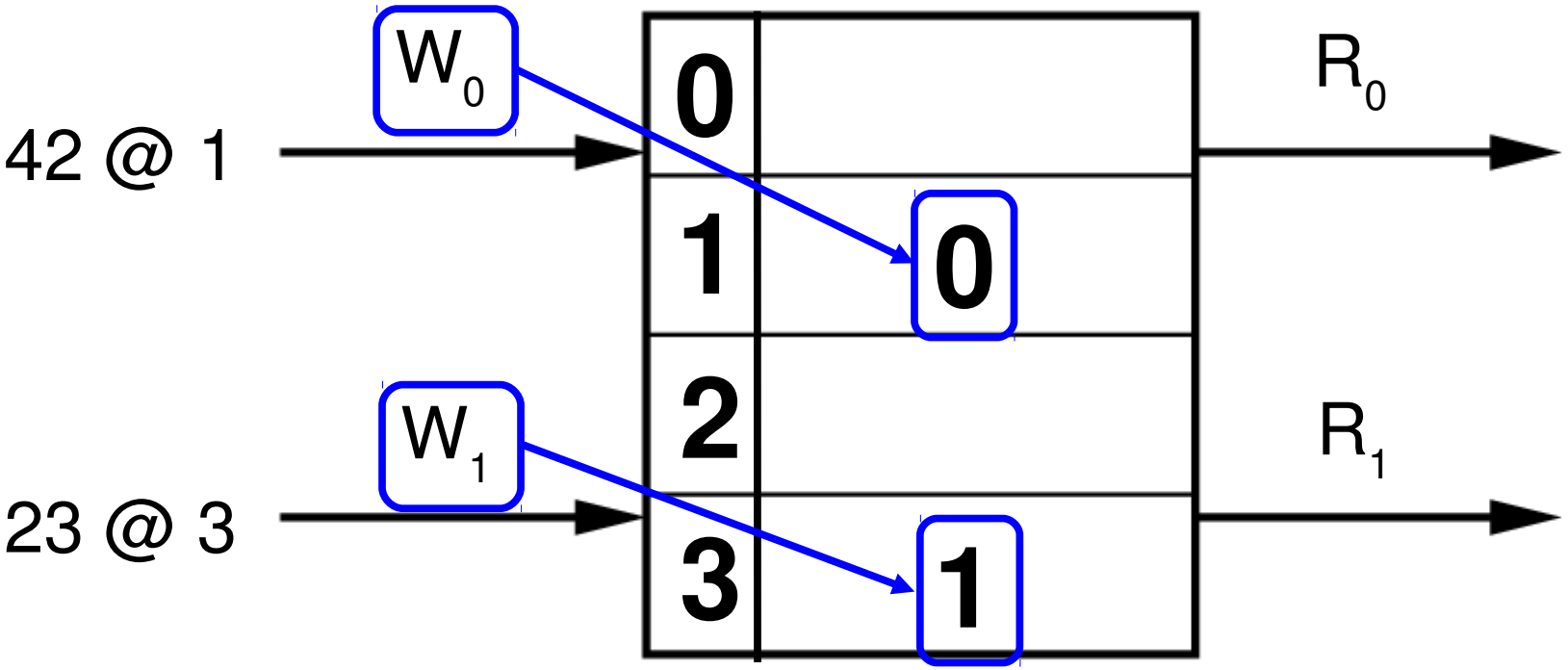


2W/2R, 4-deep

LVT Operation

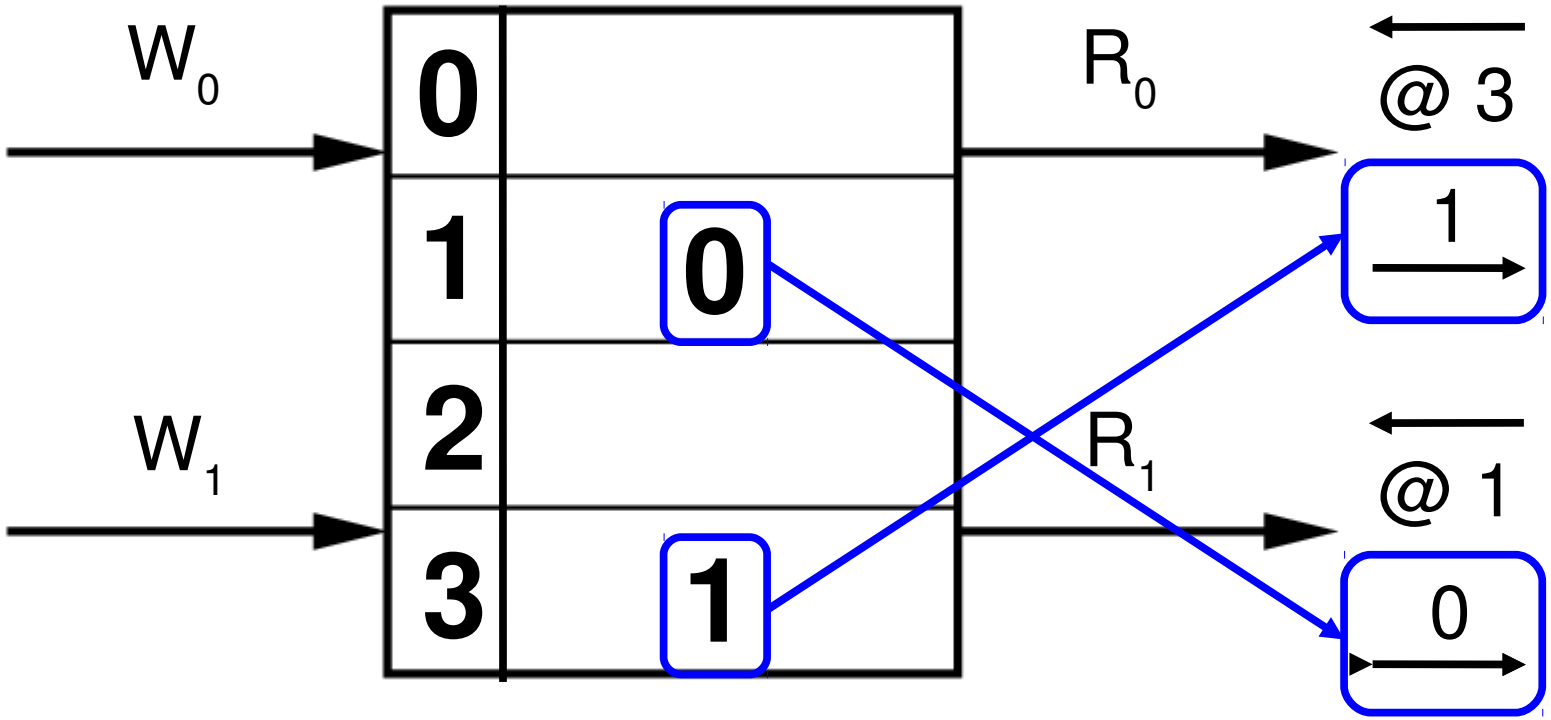


LVT Operation: Write



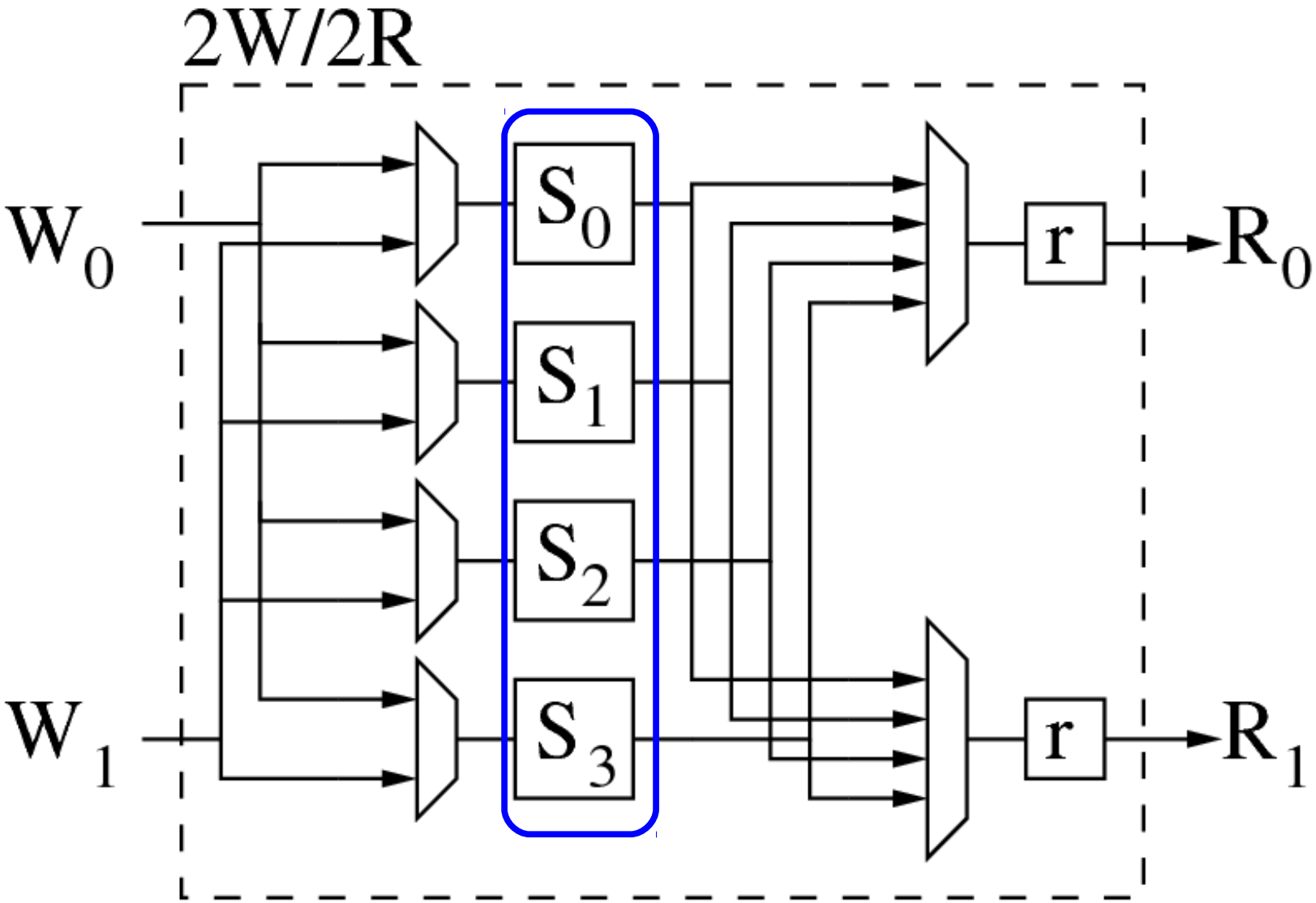
Records which write port last updated a location

LVT Operation: Read



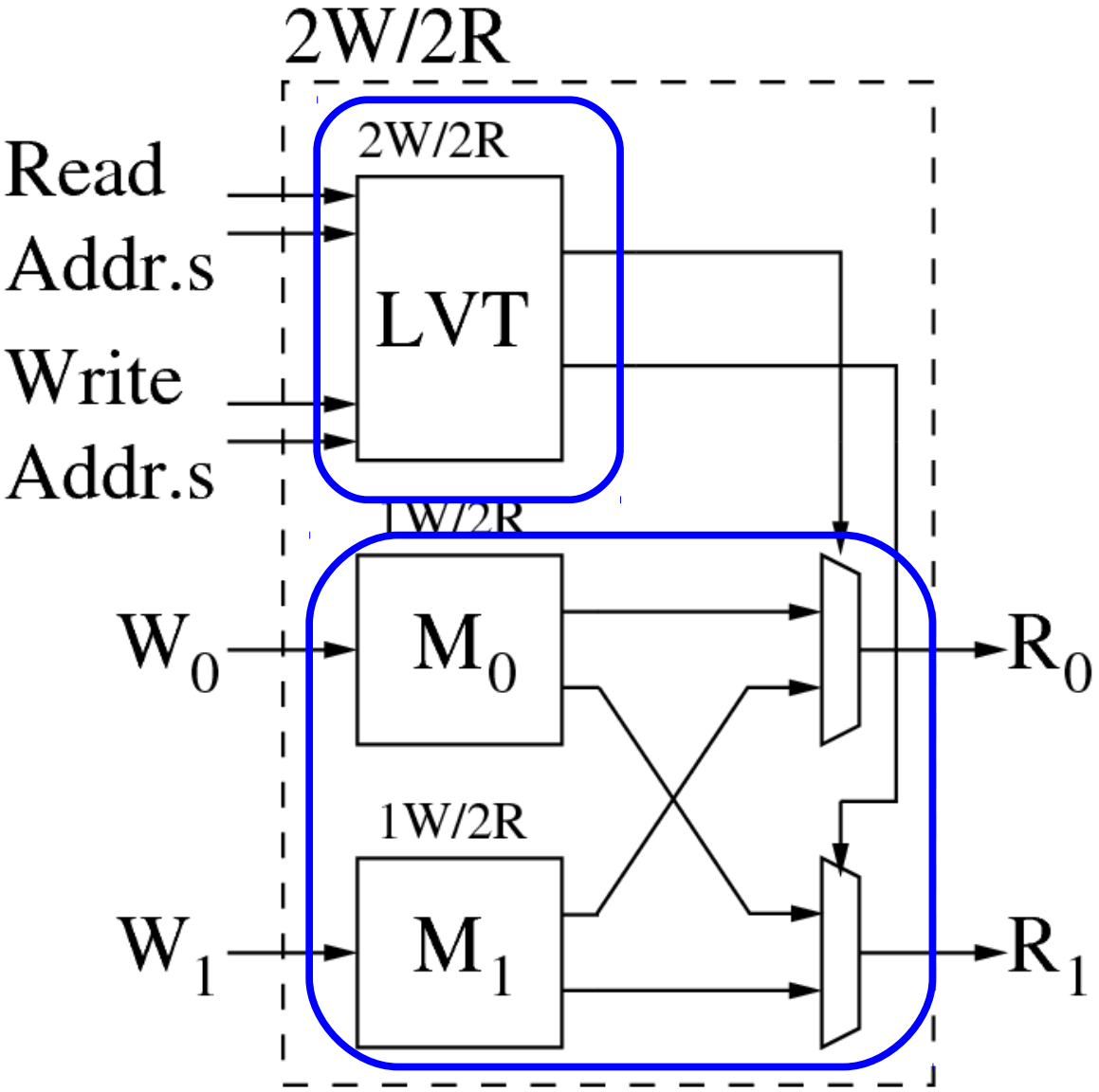
Steers read port to correct memory bank

LVT Implementation



LVT remains practical because it is very narrow

LVT Operation



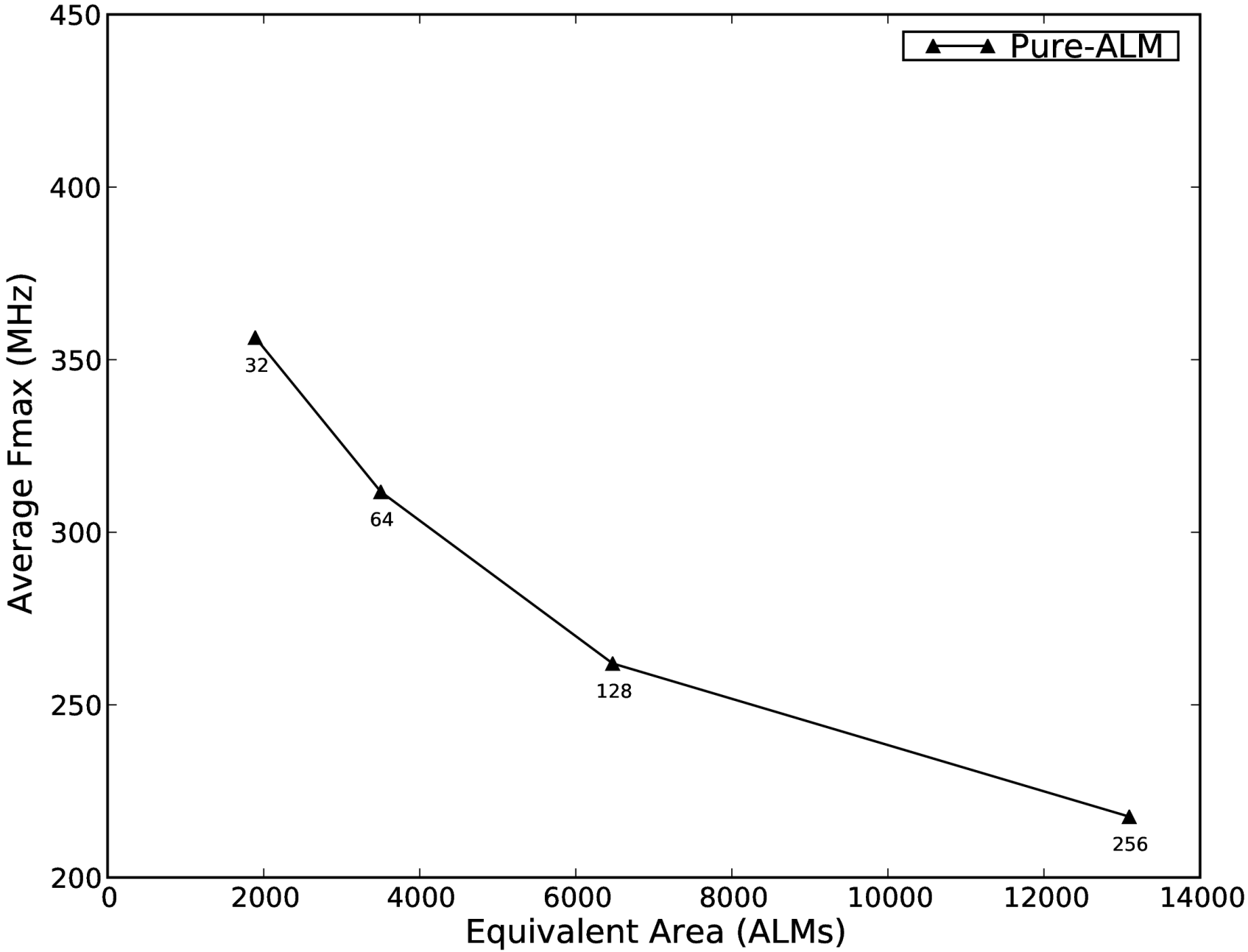
Small Pure-ALM memory controlling larger block RAMs

Advantages of LVTs

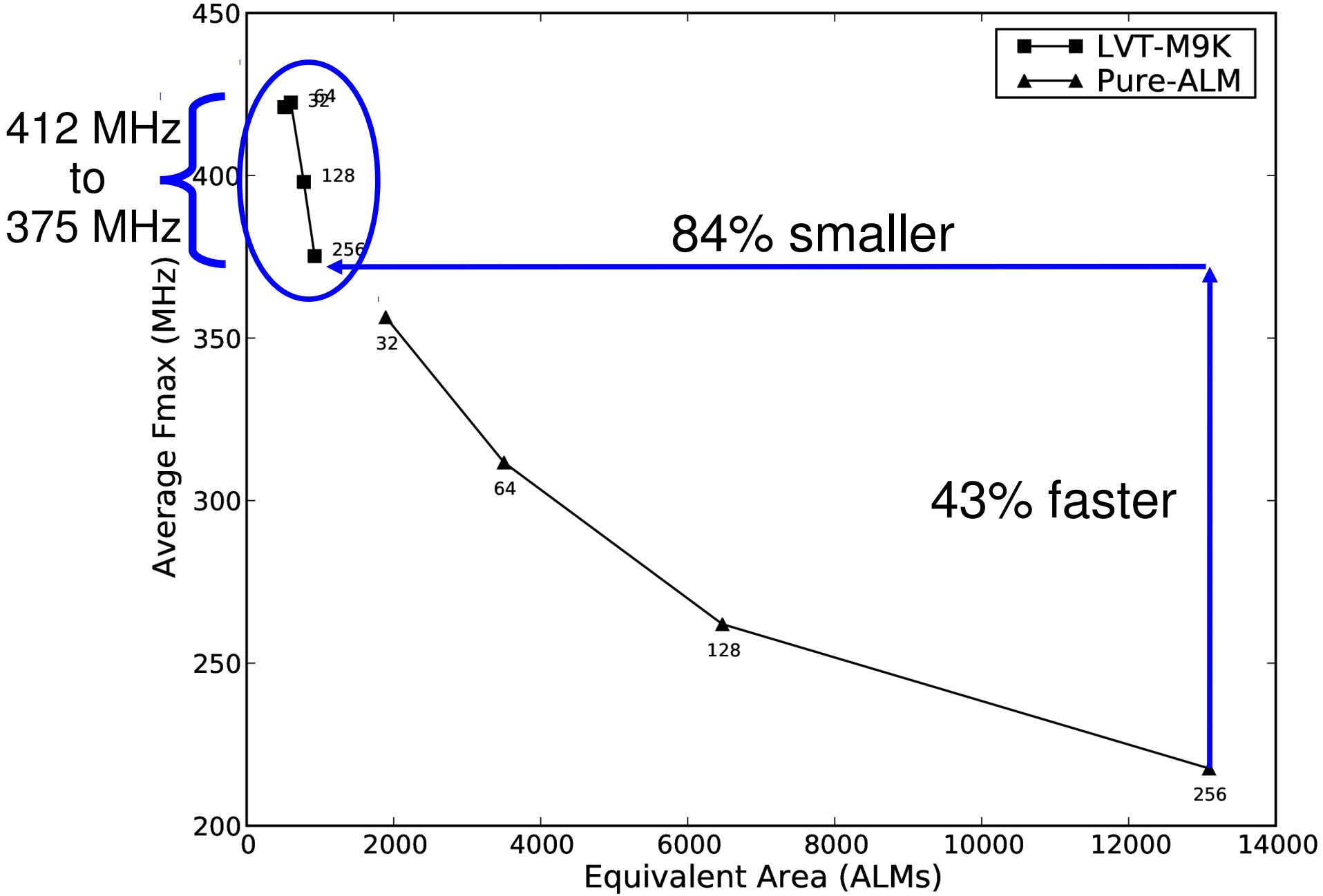
- LVTs add a layer of indirection
  - Everything operates in parallel
  - Makes banked memory behave as consistent unit
- LVTs are narrow
  - Word width = $\log_2$(# of write ports) < 4 bits typically
  - Pure-ALM, but practical size and speed

LVT Performance

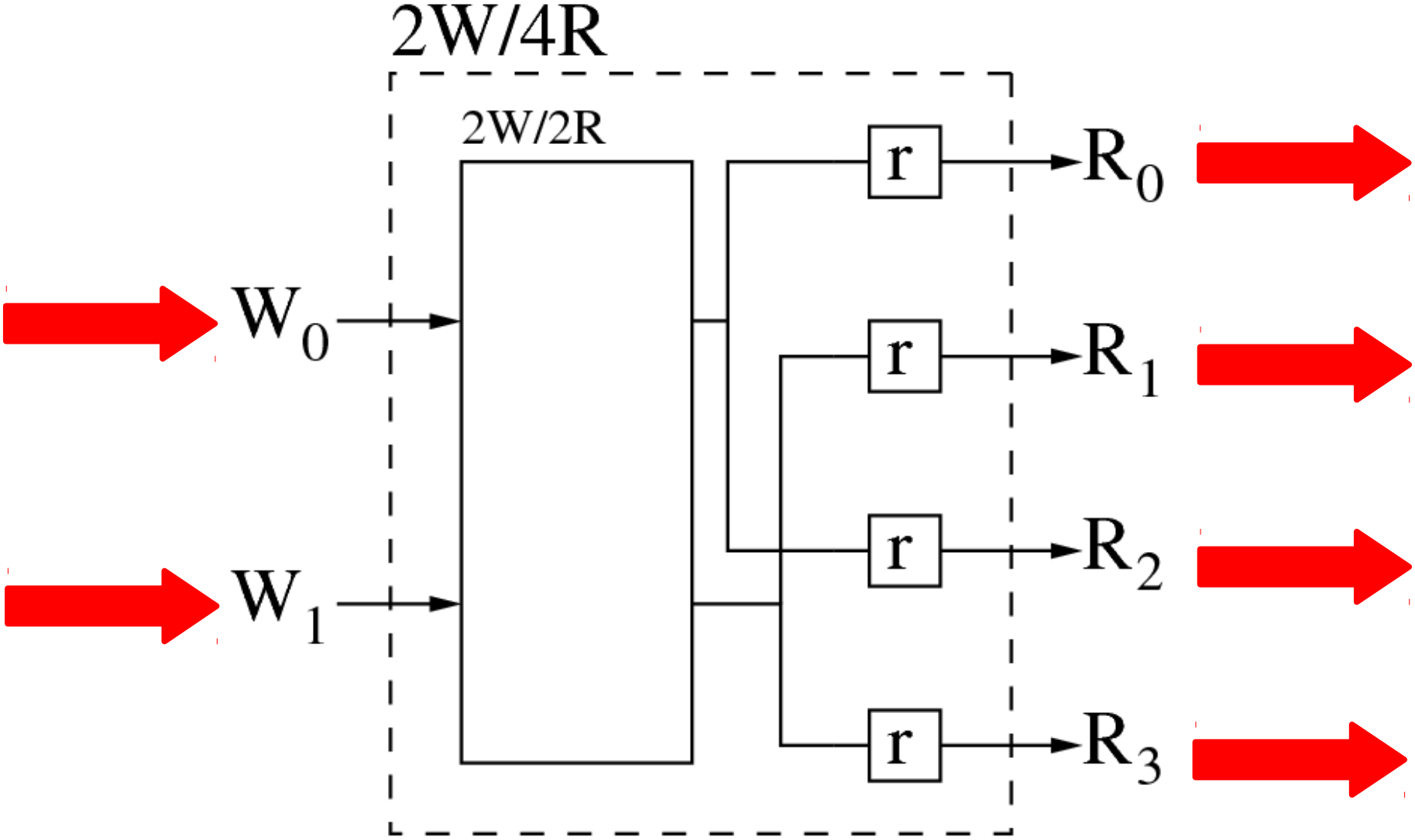
2W/4R Pure-ALM



2W/4R LVT-based vs. Pure-ALM



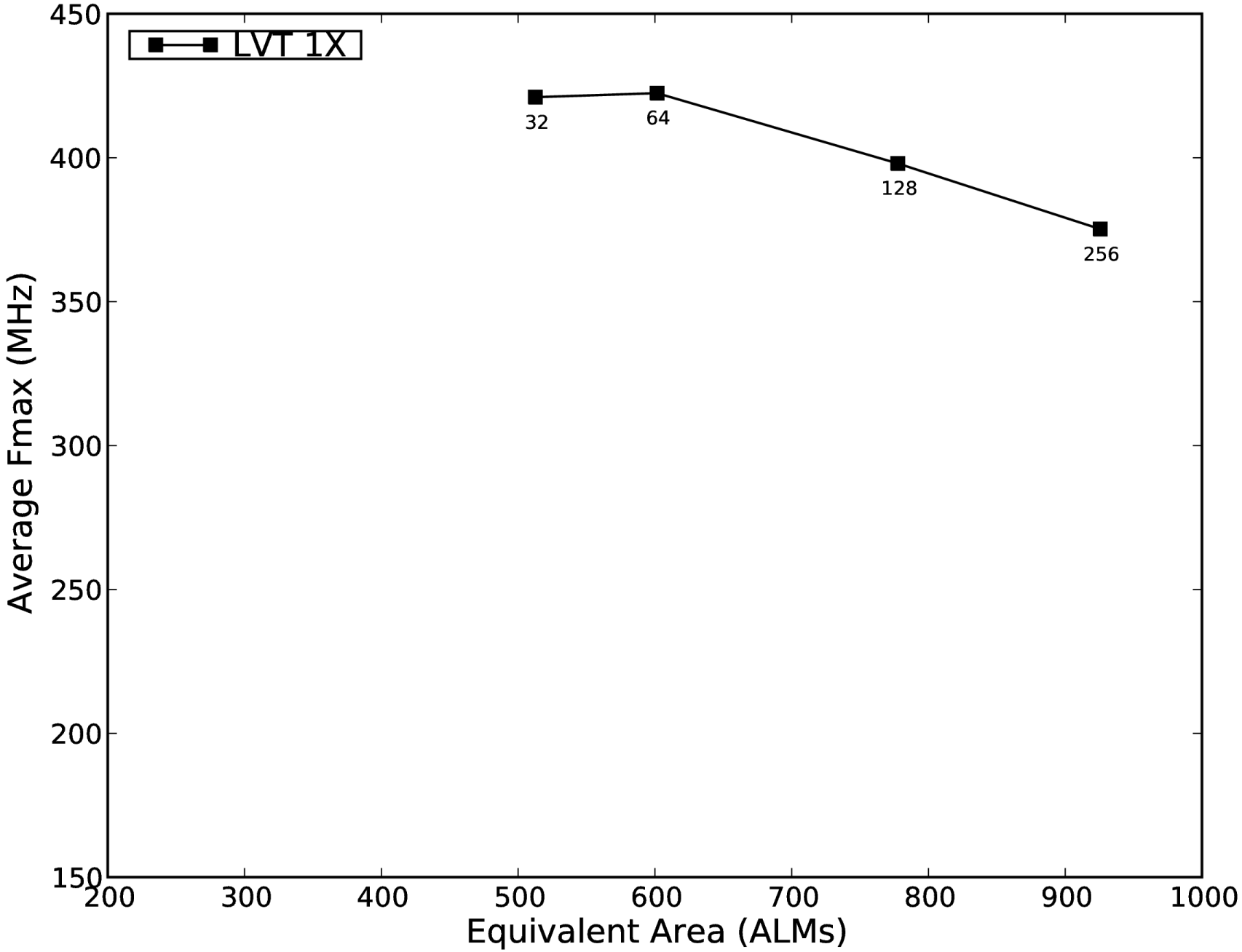
2W/4R Multipumping



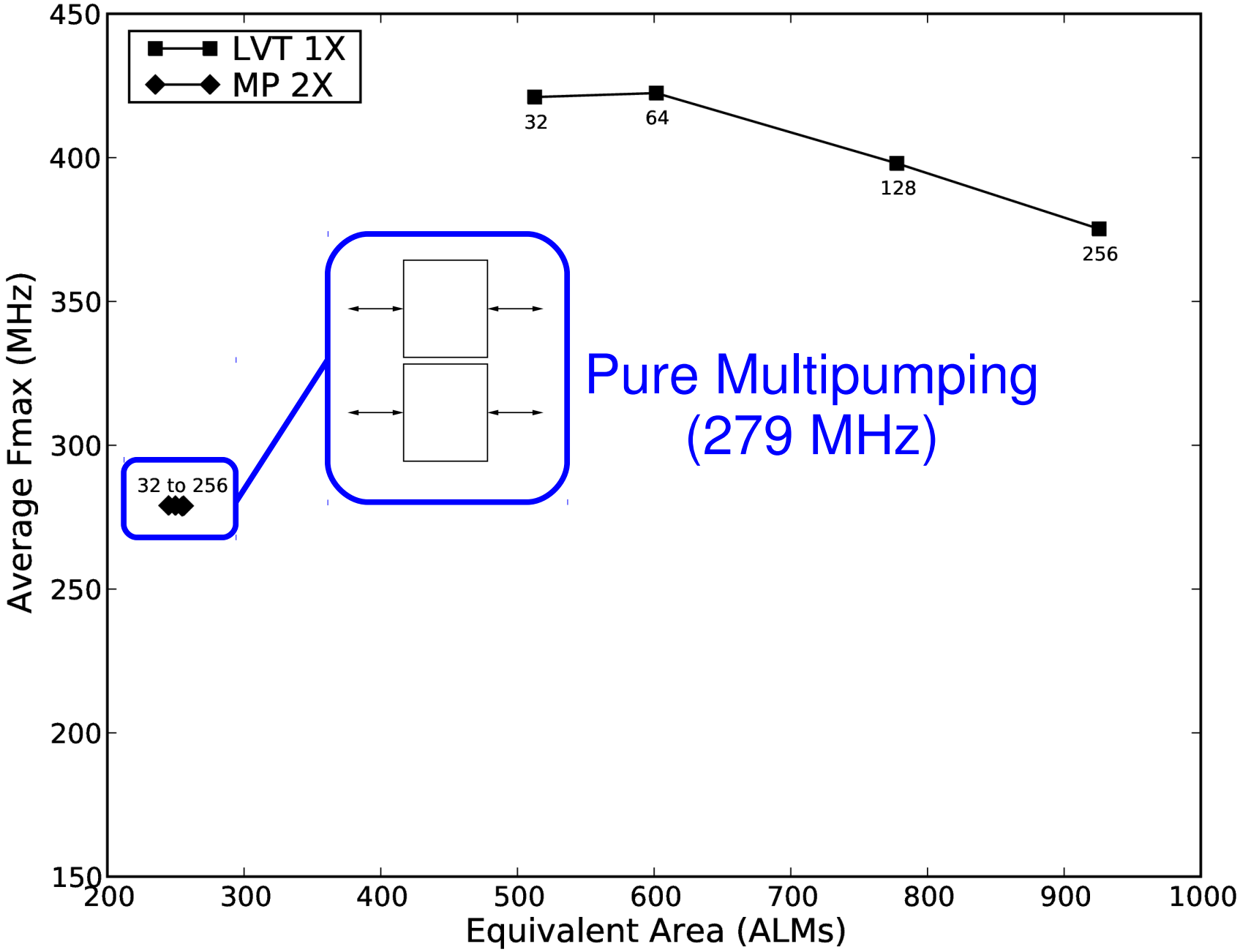
Must be careful about read/write ordering!

Multipumping Performance

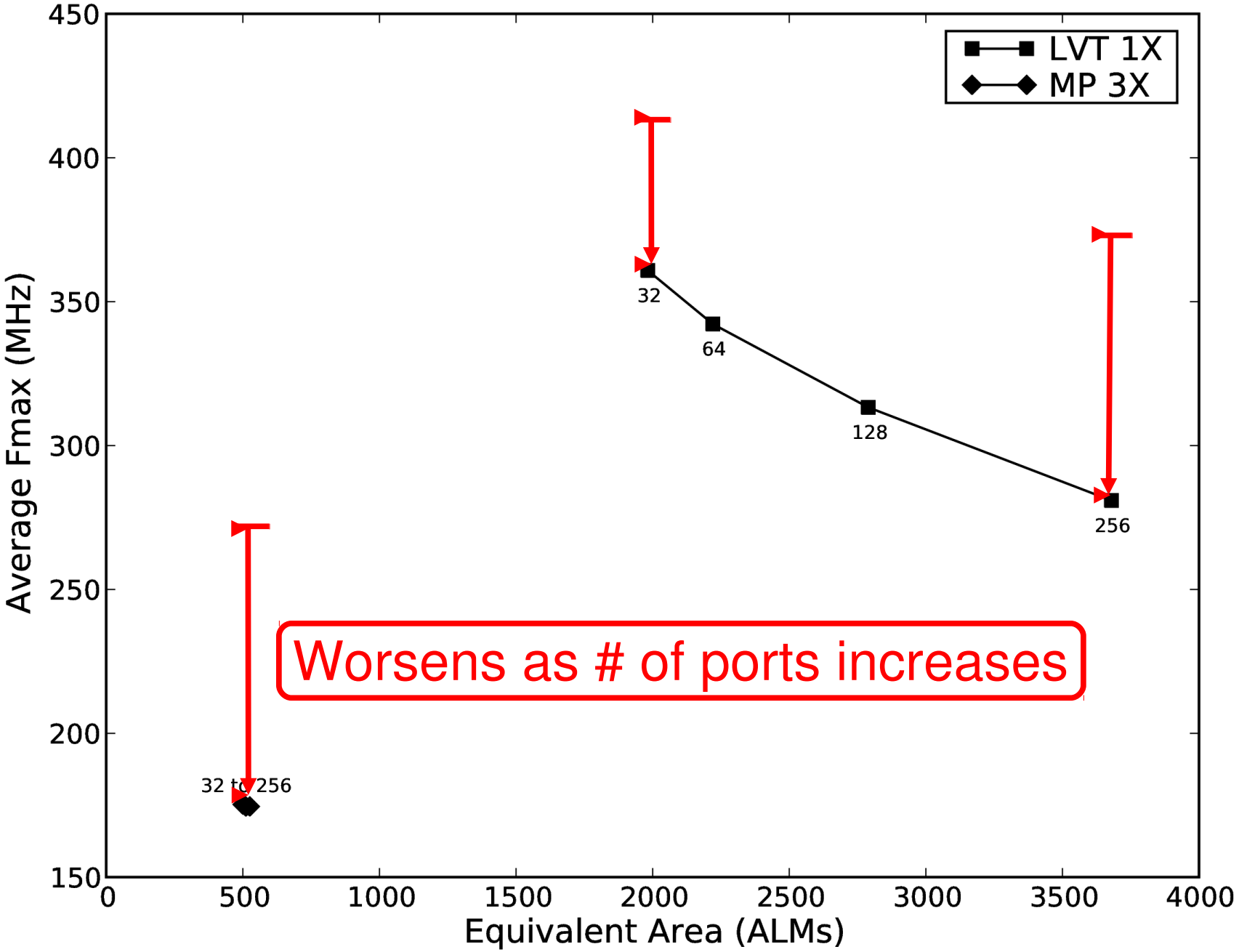
2W/4R Multipumping



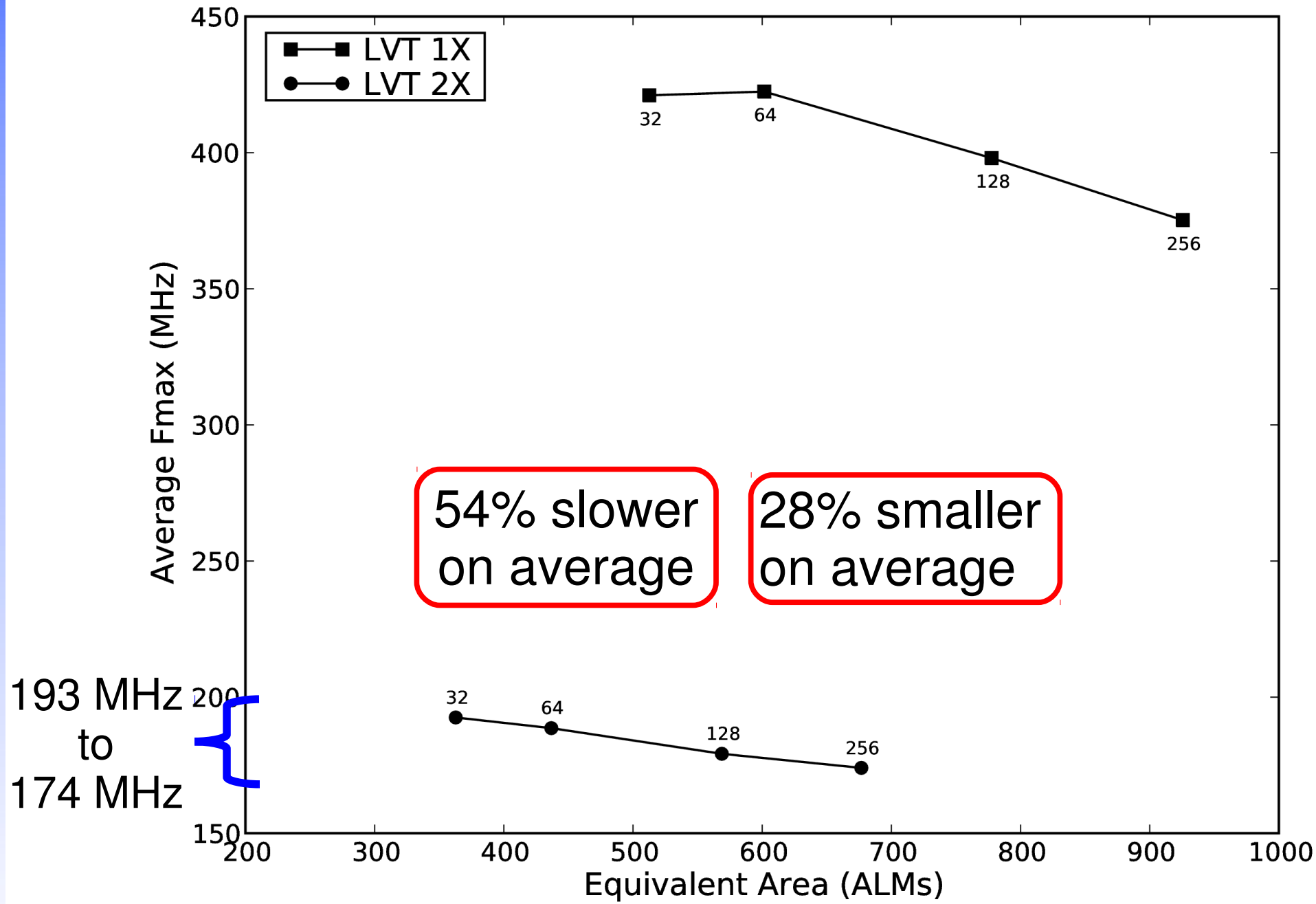
2W/4R Multipumping



4W/8R Multipumping



2W/4R Multipumping



Conclusions

- LVT-based memories are faster and smaller than Pure-ALM memories.
- LVT-based memories are faster than pure multipumping, but at a cost in area.
- Pure multipumped memories are better for memories with few ports or low speed.

Future Work

- Pure multipumping for LVT-based memories
  - Build banks with 2W/4R pure multipumping blocks
  - Possible further area improvement
- Relaxing the read/write order for multipumping
  - Allows multiplexing the write ports
  - Leaves designer to watch for WAR violations

Thank You