Utilizing Magnetic Tunnel Junctions in Digital Systems

Michael J. Hall
Advisor: Dr. Roger Chamberlain
Co-advisor: Dr. Viktor Gruev
Washington University in St. Louis

April 10, 2015

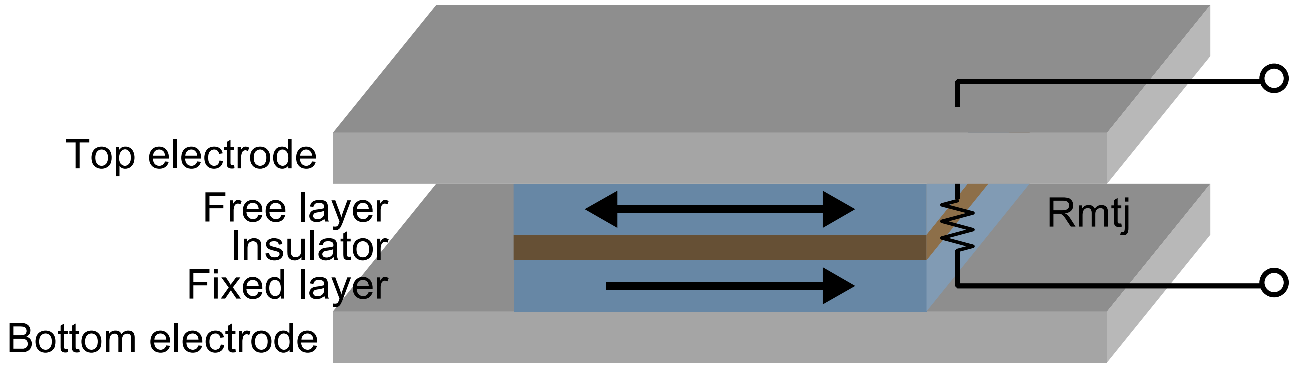
Funded by AFOSR, NSF, Exegy, Inc., and VelociData, Inc.

What this dissertation is about

- This dissertation presents work on:
  - A resistance-to-voltage read circuit using a continuous read for sensing magnetic tunnel junctions (MTJs) motivated by magnetic global clocking as a way to distribute a clock signal
  - Hardware virtualization as a way to utilize deeply-pipelined circuits with feedback motivated by magnetologic circuits that are inherently deeply-pipelined circuits with feedback

Magnetic Tunnel Junction (MTJ)

Demonstrated at 45 nm
[Lin et al. 2009]



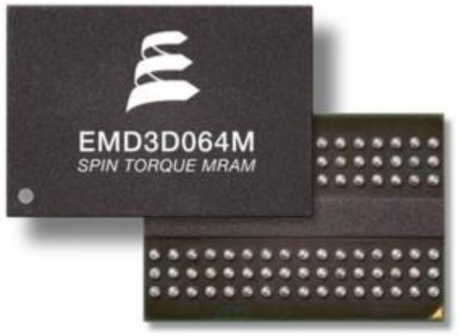
- Thin-film magnetic device
- Set via field or current
- Read via resistance output

MTJ properties

- Radiation-hard
- Non-volatile
- High write endurance
- Can be integrated on chip in the CMOS process

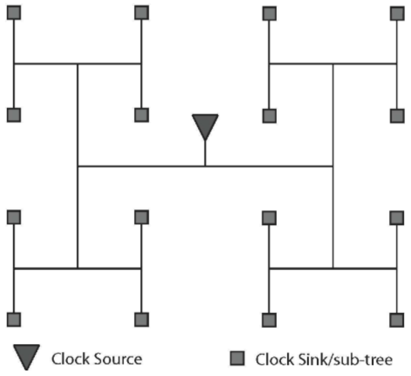
Magnetic Tunnel Junction (MTJ) Uses

Memory

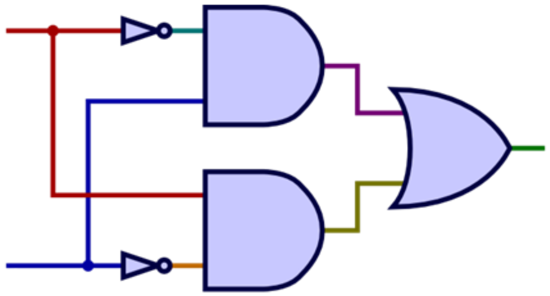


Commercially
available

Clocking



Logic



Investigate:

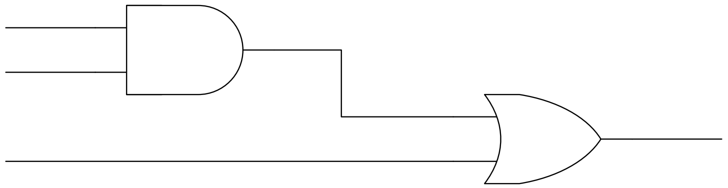
Read Circuit for Magnetic Global Clocking

Hardware Virtualization for Magnetologic Circuits

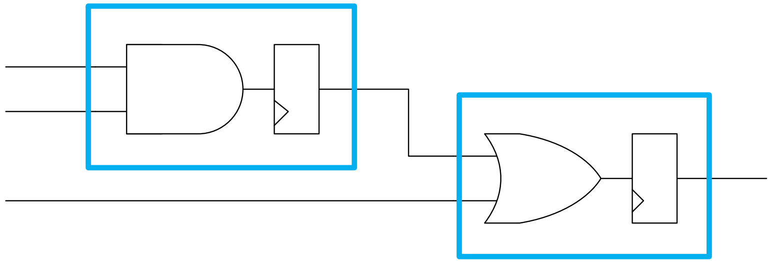
Magnetologic

[Lee et al. 2008]

Conventional gates

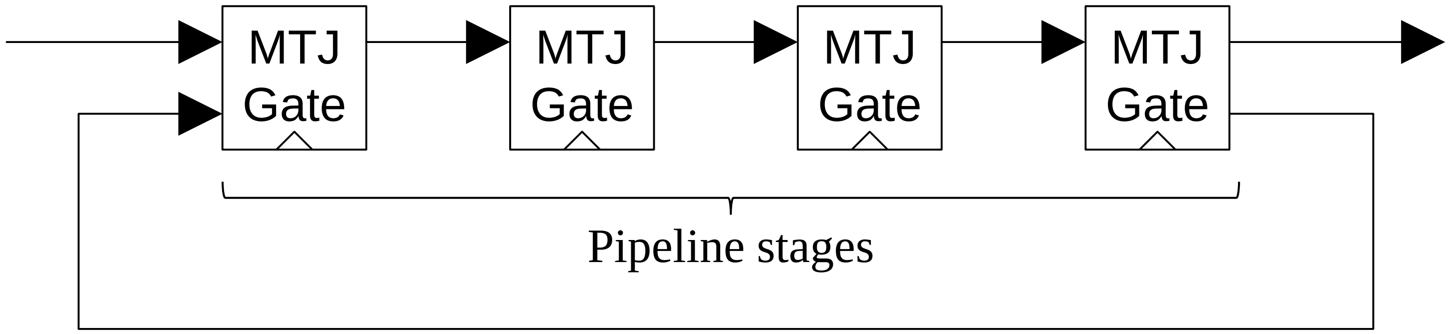


Magnetologic gates



- Conventional gates propagate signals combinationally
- Magnetologic gates have state, meaning that each gate is a pipeline stage
- For a large circuit, this can become a deeply-pipelined circuit

Magnetologic gates (cont.)



- Most digital systems have a feedback path
- We want to exploit the deeply-pipelined nature of magnetologic circuits when feedback is present

Research questions

- How can we read an MTJ device continuously?
- What can we learn about the read circuit with switching resistance inputs?
- How can we go about designing digital systems that are deeply-pipelined with feedback?

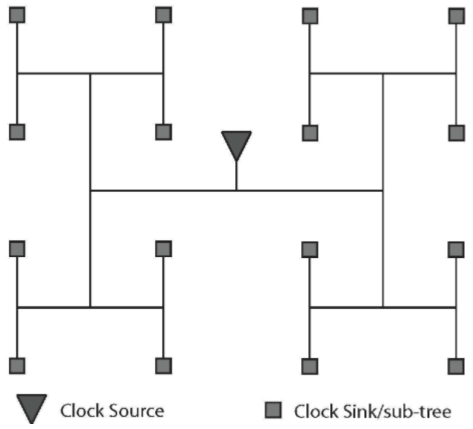
This work

- We explore two aspects of MTJ uses:
  - An experimental MTJ continuous read circuit
  - Hardware virtualization for utilizing deeply-pipelined logic circuits

Outline

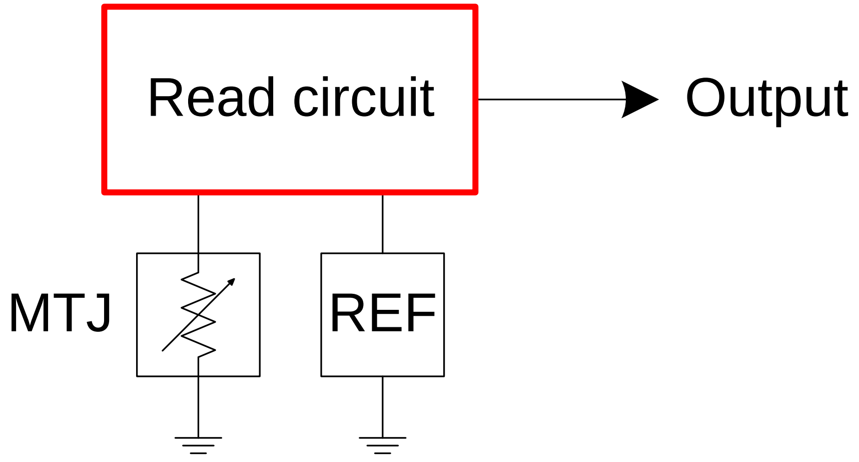
Read circuit design issues

- We target a magnetic clocking application
  - Requires continuous read operation
  - Needs to produce a logic voltage output
- Need to build a resistance-to-voltage read circuit with a continuous read



- Expect large input capacitance due to MTJ connections
- Choose a current-mode read circuit design
  - Existing current-mode read circuits in the literature are current conveyors used in memories for reading MTJs but these are sampled, not continuous
  - Continuous-mode current read is new

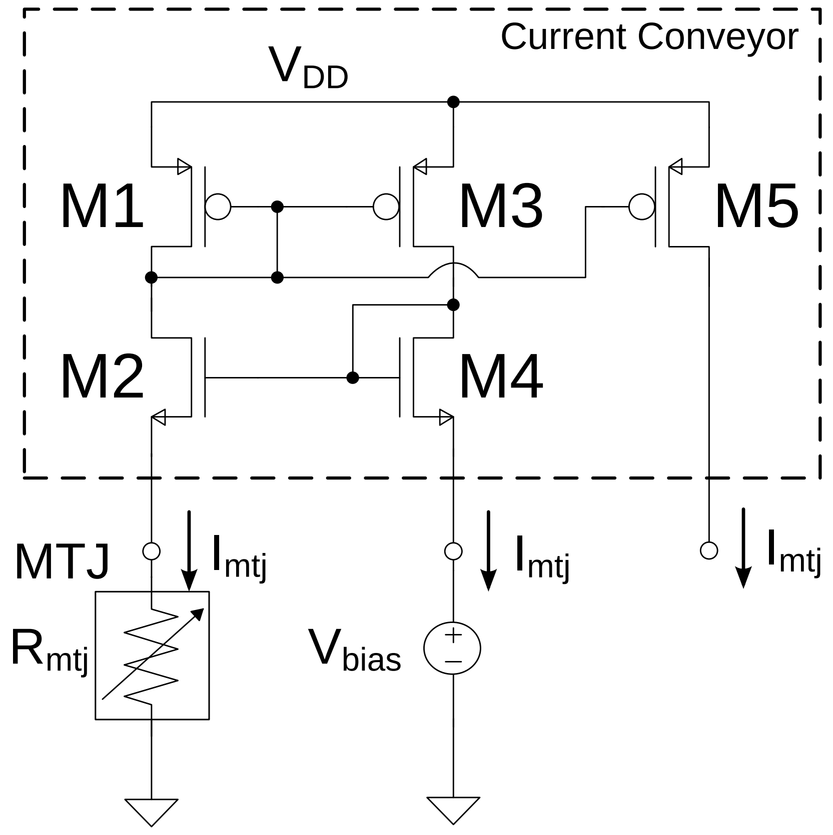
Overview of my approach



- Sense MTJ resistance
- Compare to reference
- Produce logic output

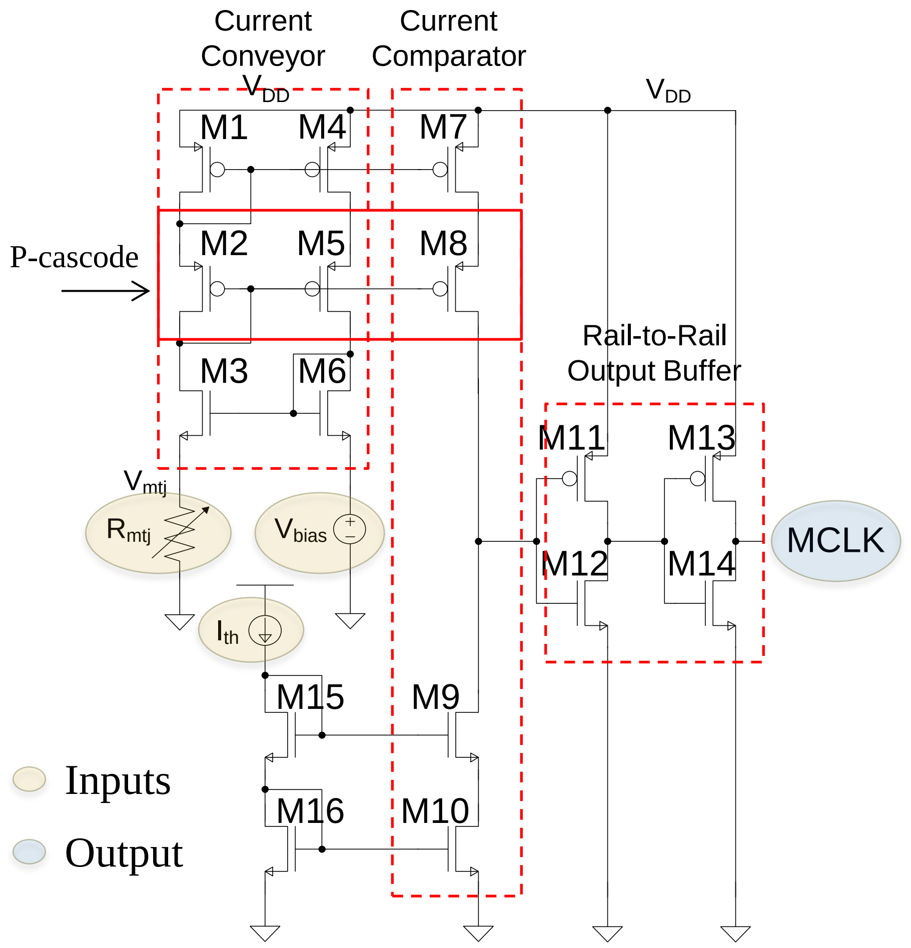
- Three parts to the resistance-to-voltage read circuit:
  - Current conveyor (MTJ sensing)
  - Current comparator
  - Output buffer

Basic current conveyor



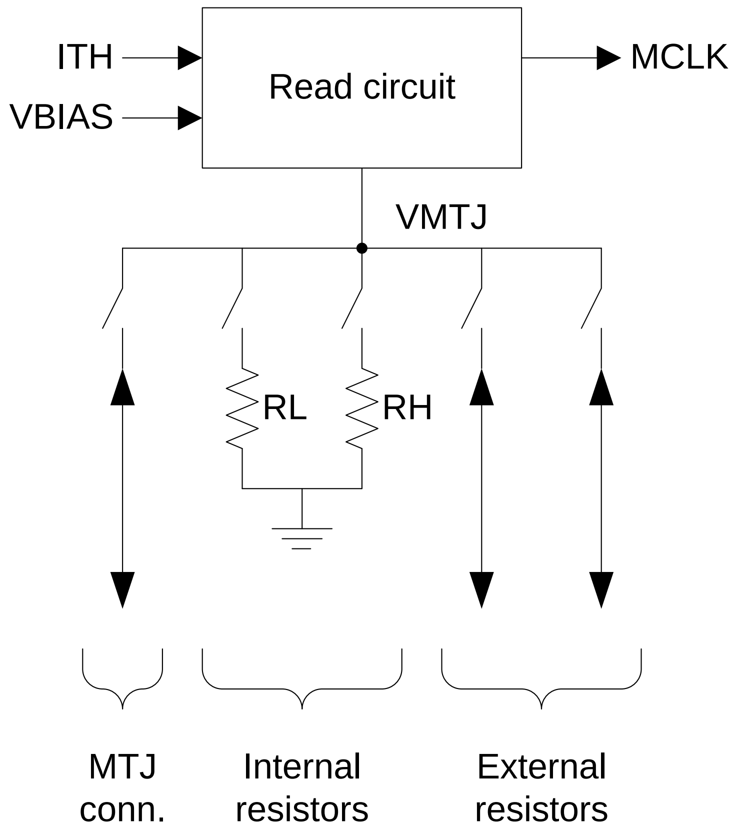
- Feedback path formed by current mirrors $M_{1,3}$ and $M_{2,4}$
- Output current mirror formed by $M_{1,5}$
- Basic operation
  - V_{bias} is clamped over R_{mtj}
  - This produces current $I_{mtj} = V_{bias} / R_{mtj}$
  - I_{mtj} is copied to the output

Resistance-to-voltage (R2V) read circuit



- Add P-cascode to improve linearity of output current and reduce 2nd order effects
- Compare output current to a threshold current I_{th}
- Amplify the comparator voltage output rail-to-rail

Instrumented read circuit for testing



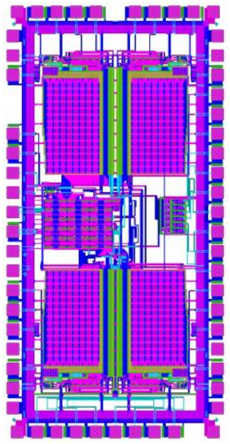
- Fabricated a prototype test chip in a 3M2P 0.5 μm process
- Read circuit testable with multiple input sources
- We test using resistors

Results

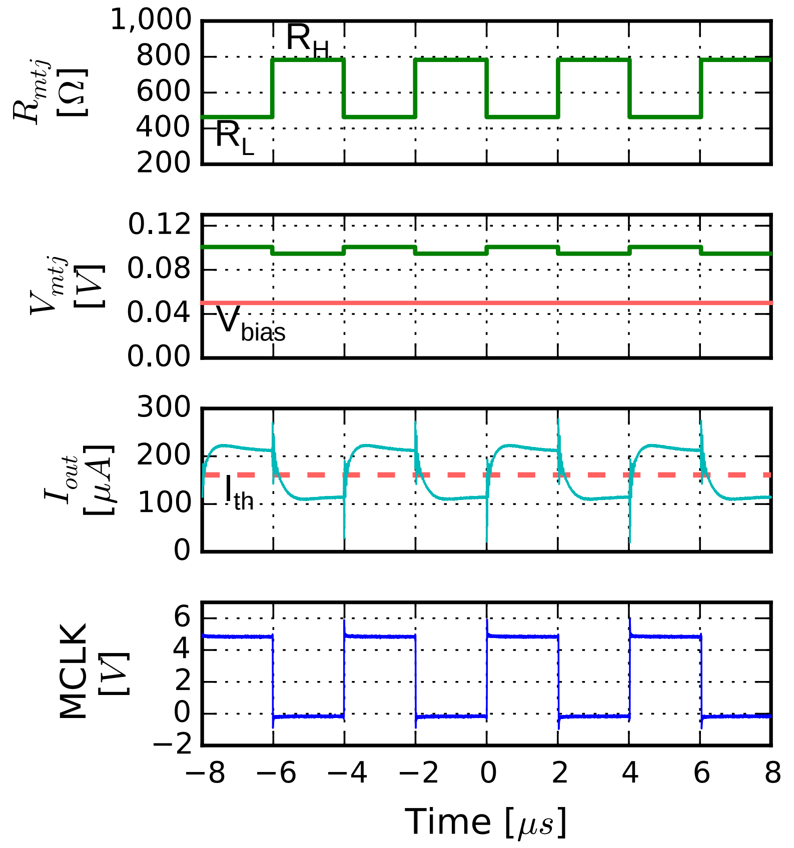
- Simulated in Cadence Design Environment using Spectre in 3M2P 0.5 µm process

- Measurements made experimentally from the fabricated prototype test chip

Printed circuit board with chip


Prototype test chip

Measured transient response

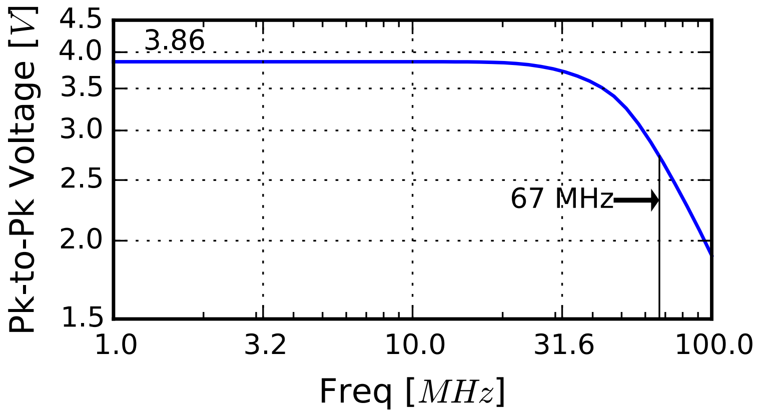


- Functionally working read circuit
- Nominal R_L = 500 Ω and R_H = 1 kΩ
- V_{mtj} tracks V_{bias} with an offset
- Setting an appropriate I_{th}, the I_{out} and MCLK outputs follow the R_{mtj} input

Simulated and measured performance

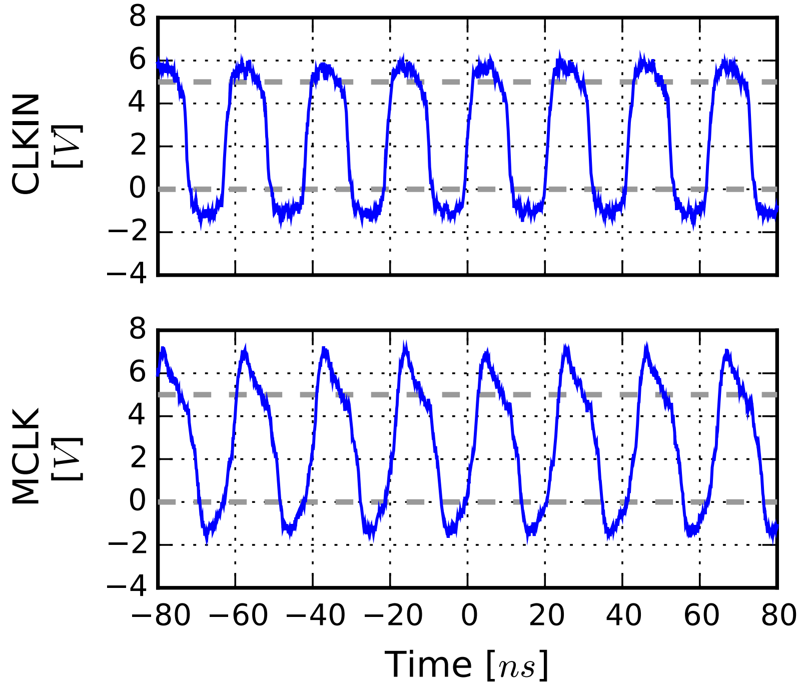
Simulation of bandwidth on the bottleneck node

Current comparator output node



Input node is relatively insensitive to node capacitance and can handle up to low 10s of pF

Measurement of f_{CLKIN} max



f_{CLKIN}= 48 MHz

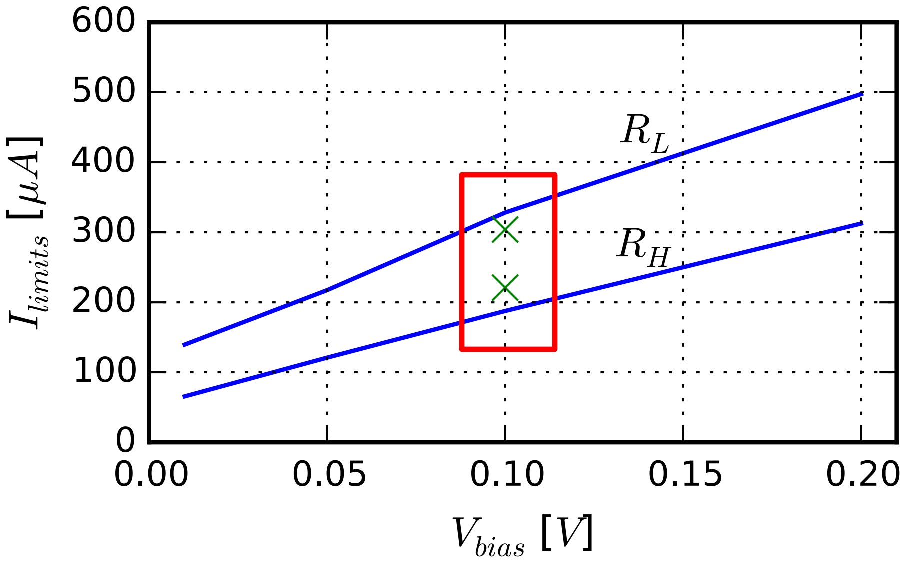
Measured range of V_{bias}

- Measured as low as V_{bias} = 50 mV
- Nominal RL is as high as V_{bias} = 0.2 V
- Nominal RH is as high as V_{bias} = 0.4 V

- V_{bias} range: ≈ 50 mV to 0.2 V

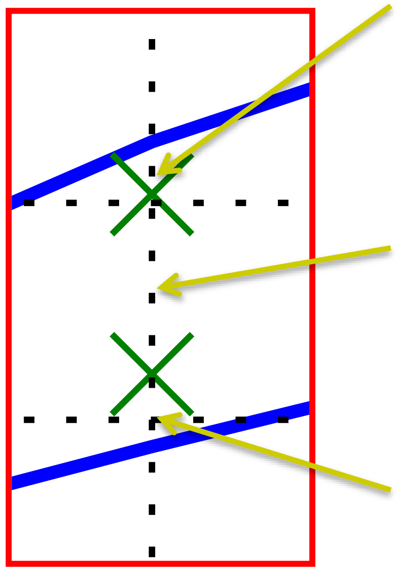
- This range covers the voltages that we would want to operate MTJs at.

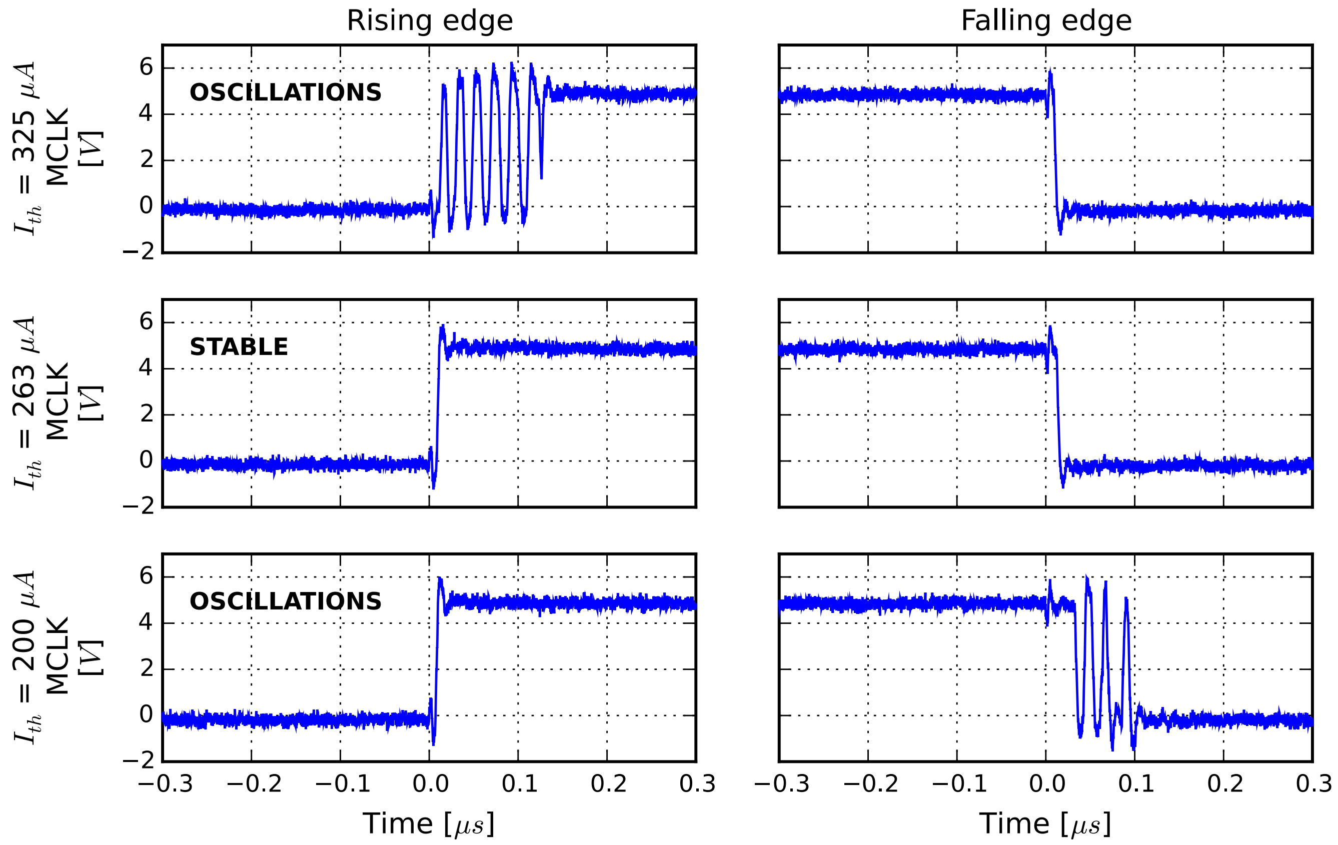
Measured dynamic stability range



- Blue lines are the I_{out} current for R_L and R_H resistances
- Green X's are the measured limits of I_{th} between which the output is stable

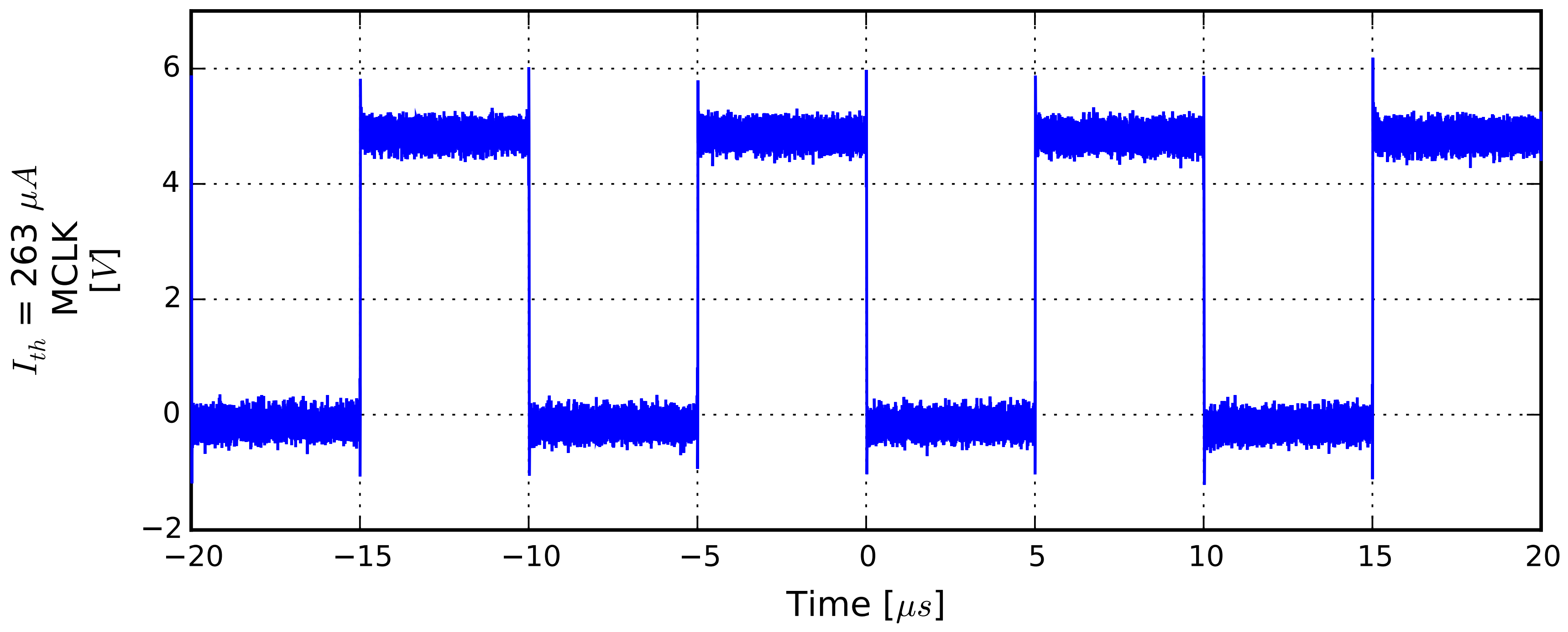
Measured output waveforms





Rising edge
Falling edge
OSCILLATIONS
STABLE
OSCILLATIONS
I_{th} = 325 μA MCLK $[V]$
I_{th} = 263 μA MCLK $[V]$
I_{th} = 200 μA MCLK $[V]$
Time $[\mu s]$
Time $[\mu s]$

Full stable output waveform



Summary for read circuit

- Designed, fabricated, and tested a resistance-to-voltage (R2V) read circuit
- Now, we can distribute a global clock using MTJs because we can read from them continuously
  - Still need to test with a real MTJ
  - For global clocking, still need to generate the magnetic field

Other thesis results

- [Sec. 3.1] Noise analysis of the current conveyor circuit
- [Sec. 3.1.4] Design guidance for tuning circuit parameters
- [Sec. 3.3] Simulation results of other properties of the read circuit
- [Sec. 4.3] Additional empirical measurements taken from the fabricated chip

Outline

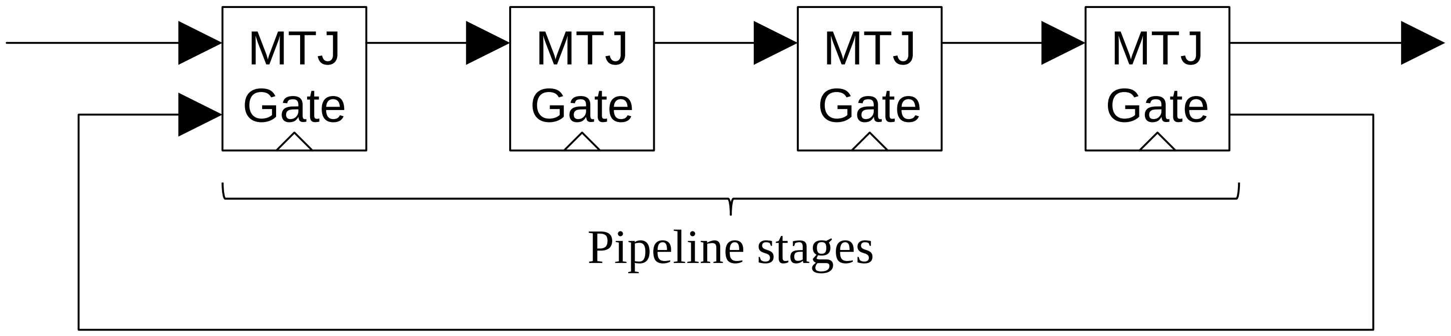
Introduction

MTJ Read Circuit

Hardware Virtualization [GLSVLSI’14] [ASAP’14]

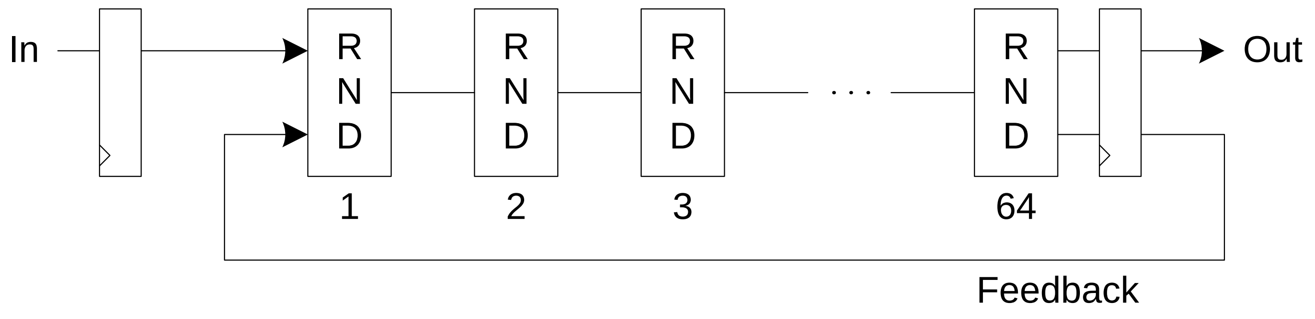
Conclusions & Future Work

Magnetologic gates



- To utilize these pipeline stages, we are going to virtualize this computation

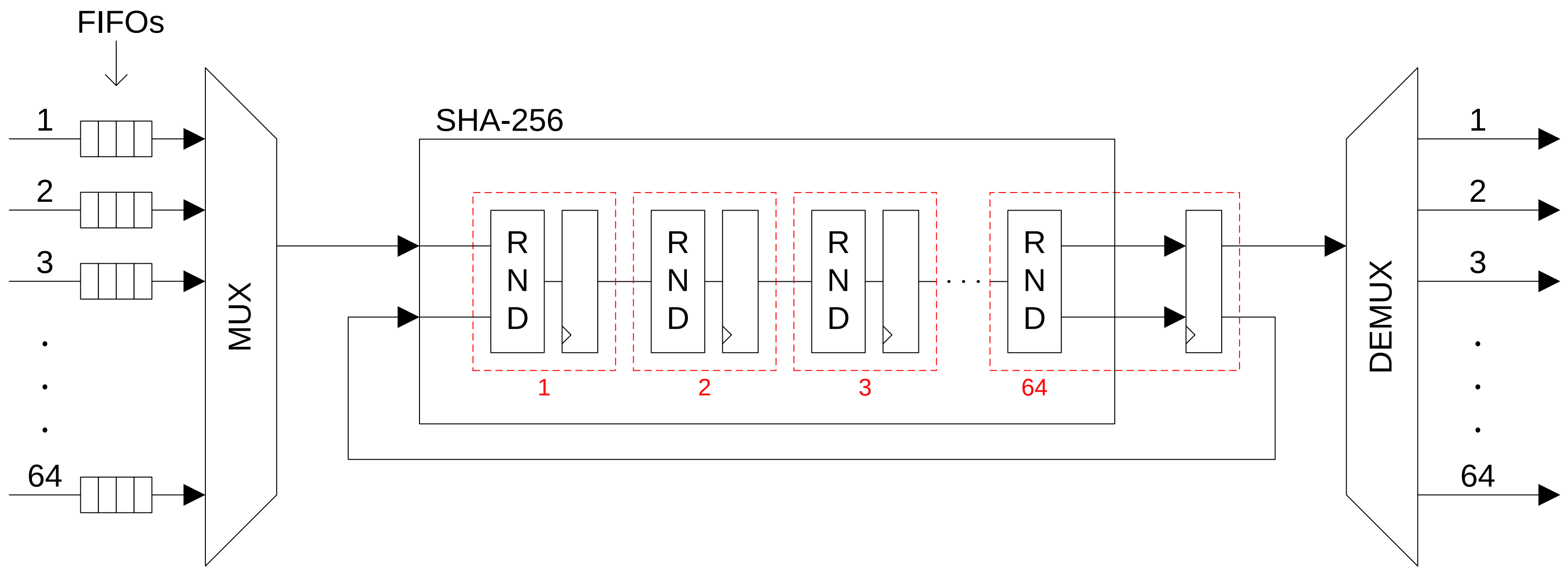
SHA-256 application



- Each round (RND) performs a series of operations on a block of data propagating through to the output
  - Rotations, logical operations, and additions
- Long propagation delay → Large clock period

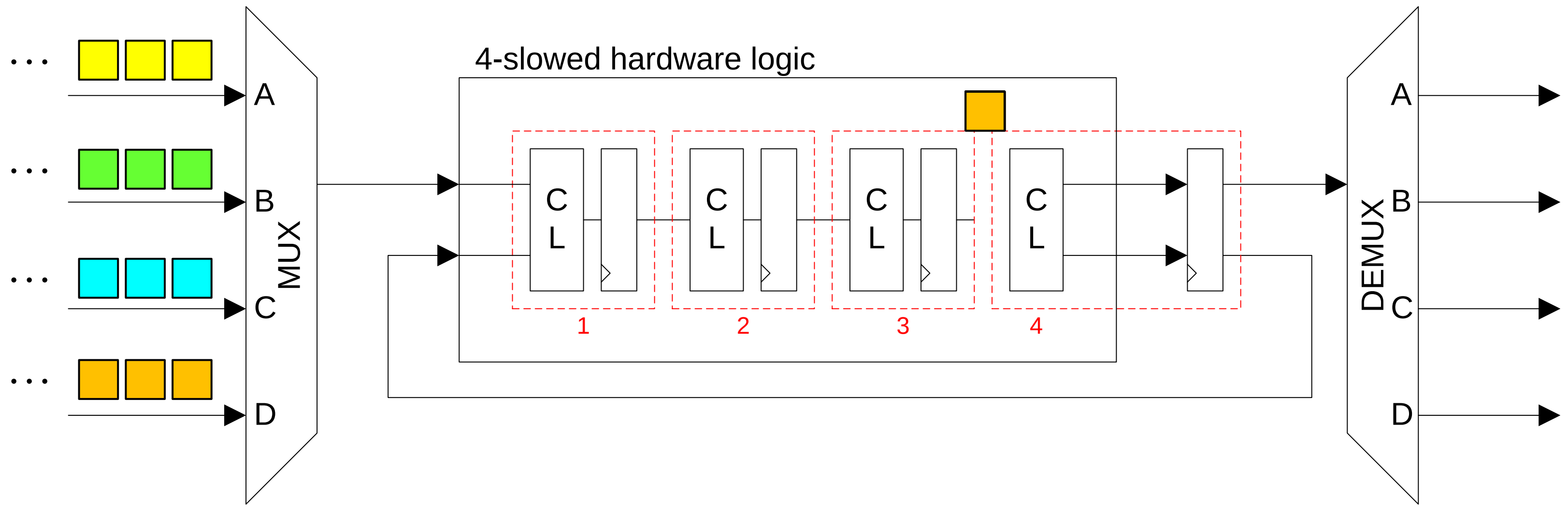
64-slow SHA-256

[Leiserson and Saxe 1991]



- Apply C-slow to this circuit with C = 64
- We now have 64 virtual copies

4-slow virtualization example

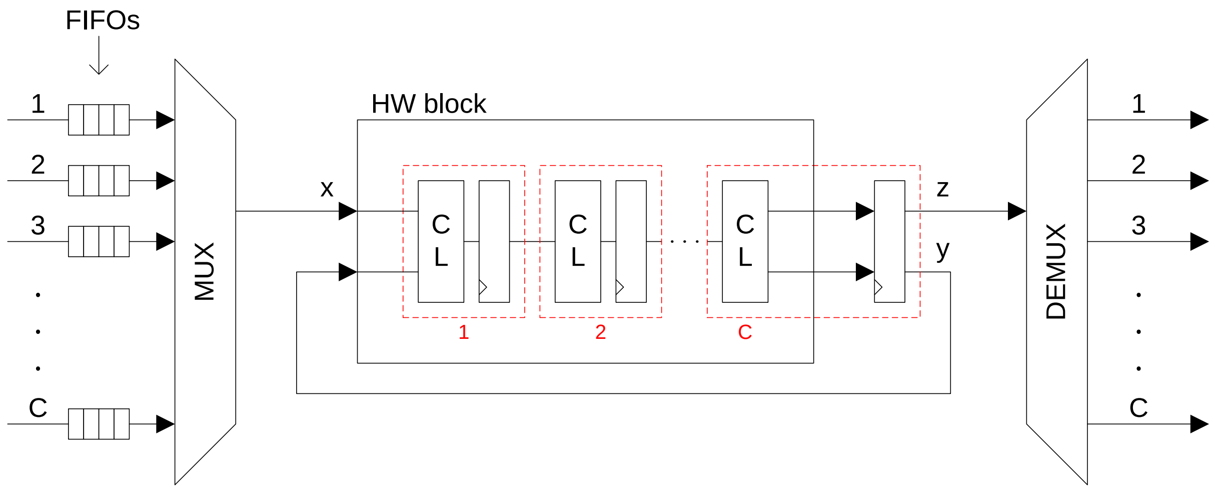


We will schedule the hardware using a *fixed*, *round-robin schedule*.

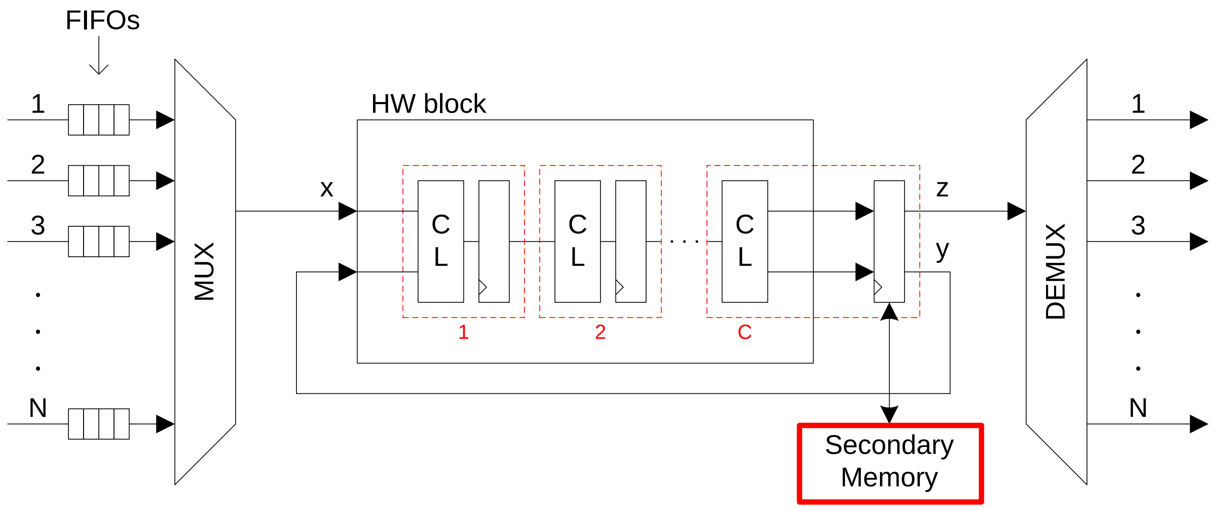
Time

Stream	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	🟨				🟨											
B		🟩				🟩										
C			🟦				🟦									
D				🟧												

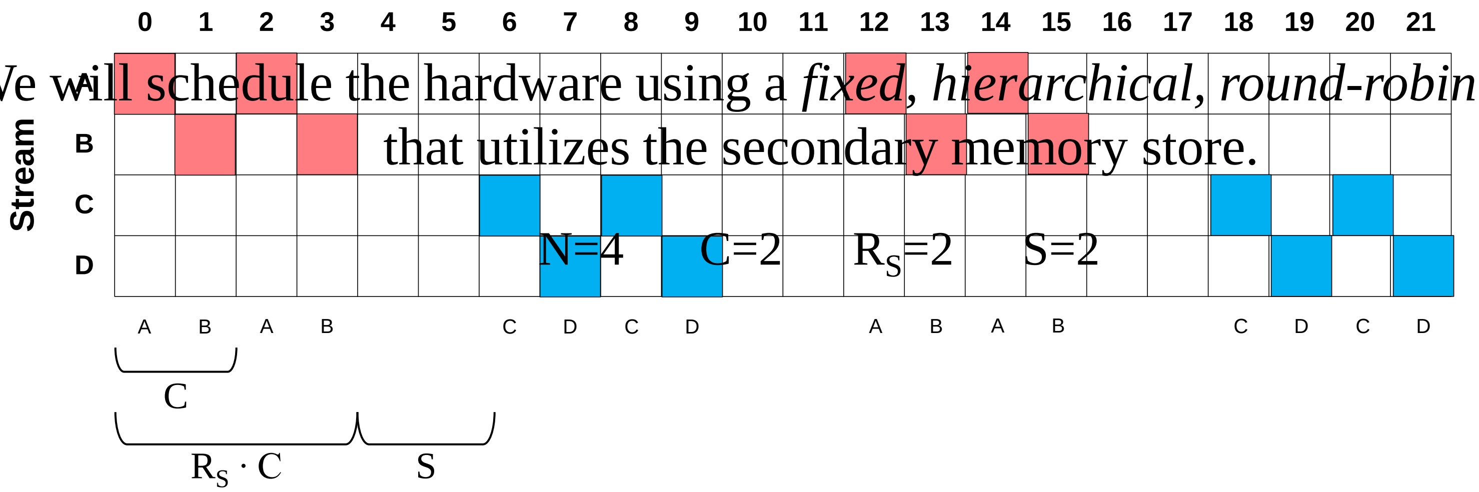
C-slow general virtualized hardware



C-slow general virtualized hardware



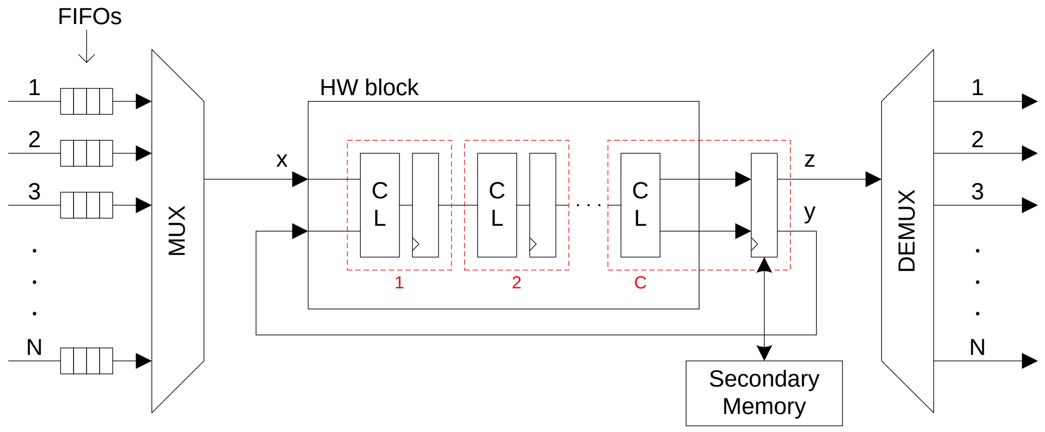
Var.	Definition
N	Total contexts
C	Pipeline depth
R_S	Scheduling period
S	Cost of context switch

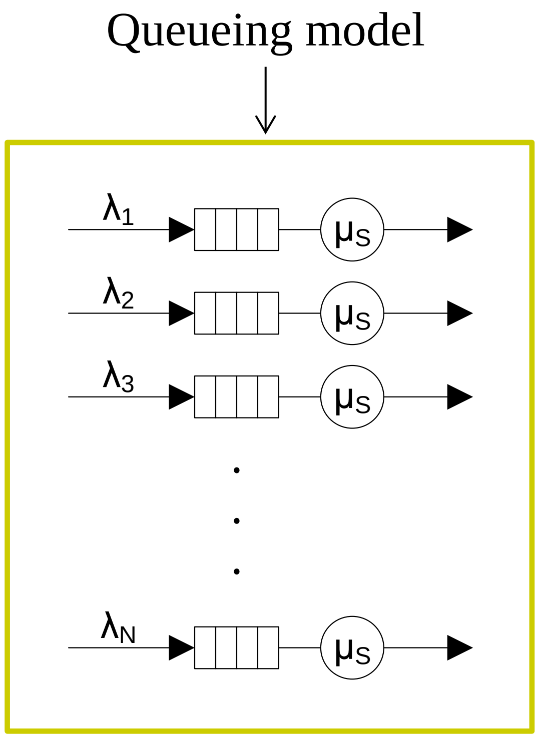


We will schedule the hardware using a *fixed*, *hierarchical*, *round-robin schedule* that utilizes the secondary memory store.

N=4 C=2 R_S=2 S=2

Queueing model





- Each queueing station is modeled as an M/G/1 queueing model with vacations
- M/G/1 is *M*arkovian, or memoryless, arrival process; *G*eneral service process; and *1* server

Model definition

$$\text{Tput, Latency, Occupancy} = f\,(\text{Circuit, Tech}, C, N, S, R_S, \lambda)$$

Variable	Definition
Circuit	Logical circuit description (e.g. SHA-256)
Tech	Target technology (e.g. MTJ, FPGA, or ASIC)
C	Pipeline depth (also represents the number of fine-grain contexts)
N	Total number of contexts (requires secondary memory if N > C)
S	Cost of a context switch (to/from secondary memory)
R_S	Scheduling period (number of rounds of C contexts that execute before doing a context switch to secondary memory)
λ	Arrival rate (e.g. data elements per second)

Performance model

Total achievable throughput:

$$T_{TOT} = \frac{R_S}{(R_S + S/C) \cdot t_{CLK}}$$

Total wait time (latency):

$$W_T = \frac{\lambda \overline{X^2}}{2(1-\rho)} + \frac{\overline{V}}{1-\rho} + \overline{X}$$

Number in queue:

$$N_q = \frac{\lambda^2 \overline{X^2}}{2(1-\rho)} + \frac{\lambda \overline{V}}{1-\rho}$$

Variable	Definition
C	Pipeline depth
N	Number of streams
S	Context switch cost
R_S	Scheduling period
λ	Mean arrival rate
ρ	Utilization
$\overline{X}$	Mean service time
$\overline{X^2}$	Service time second moment
$\overline{V}$	Mean vacation waiting

Ways to use the model in design

Model definition:

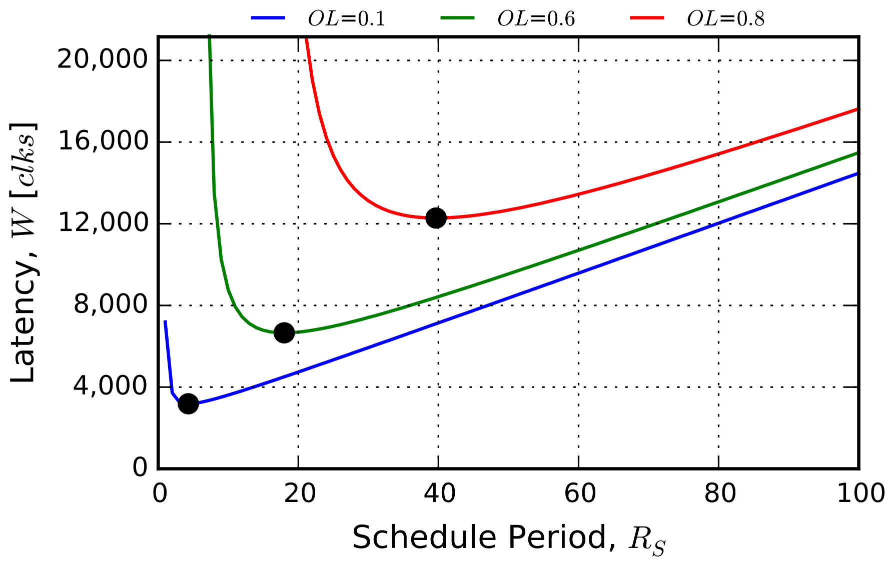
$$\text{Tput, Latency, Occupancy} = f\,(\text{Circuit, Tech}, C, N, S, R_S, \lambda)$$

- Subset of parameters are given
  - Eg. Circuit, Tech, N, C, S
- Remainder under control of designer
  - Eg. R_S, λ
- Design goal
  - Eg. Latency

Variable	Definition
C	Pipeline depth
N	Total contexts
S	Cost of a context switch
R_S	Scheduling period
λ	Arrival rate

Example Design Case 1

- Givens:
  - Circuit=SHA-256, Tech=MTJ, N=2C, C=491, S=2000, λ varies
- Design params:
  - R_S
- Optimize:
  - Latency

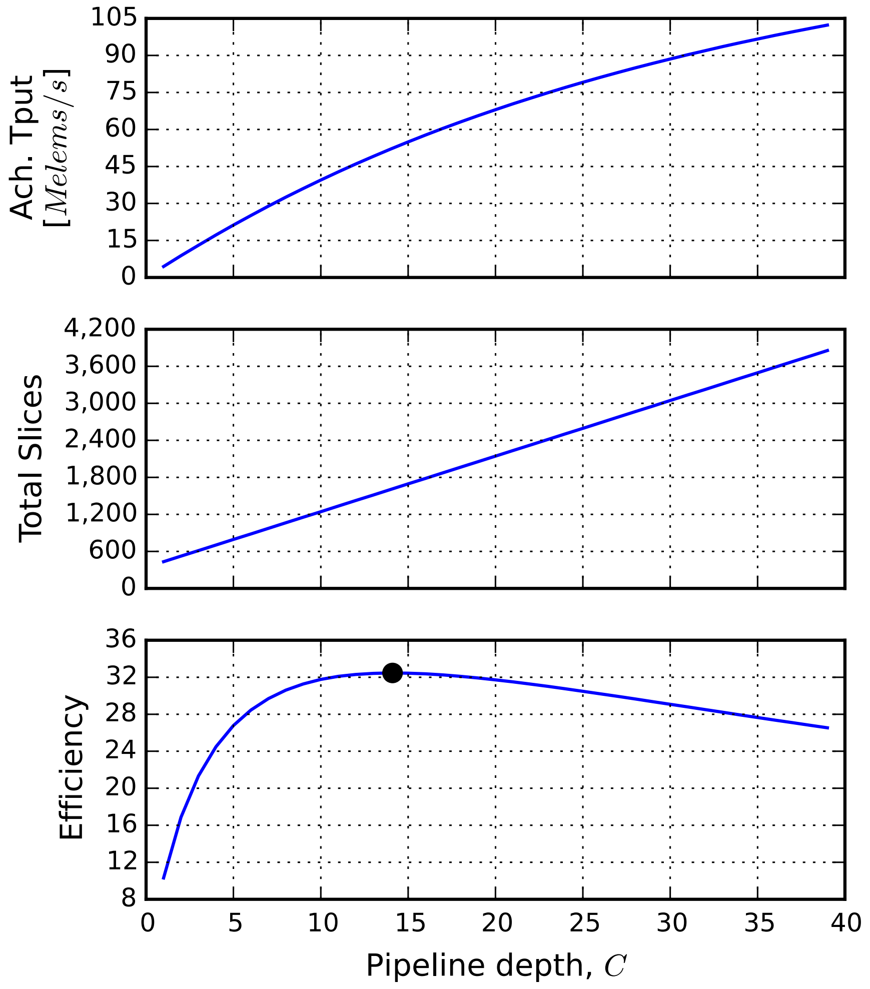


Note, Offered Load (OL) is λ normalized

C	Pipeline depth	S	Cost of a context switch
N	Total contexts	R_S	Scheduling period

Example Design Case 2

- Givens:
  - Circuit=COS, Tech=FPGA, N=C, S=0
- Design params:
  - C
- Optimize:
  - Efficiency = Tput/Slices



Summary of virtualized hardware

- Designed C-slow virtualized hardware
- We can now utilize the pipeline stages of deeply-pipelined logic circuits using hardware virtualization

- Developed an M/G/1 queueing model of the virtualized hardware with a fixed, hierarchical, round-robin schedule
- We can now optimize the performance of virtualized hardware and provide design guidance
  - For MTJ technology, optimized for minimum latency
  - When C is a design parameter, co-optimized for high throughput and low resource usage

Other thesis results

- [Sec. 5.1] Clock period model
- [Sec. 5.2] M/D/1 queueing model that preceded the M/G/1 model
- [Sec. 5.2.2 and 5.3.4] Validated the queueing models via a discrete-event simulation
- [Sec. 5.4] Calibration of three C-slowed applications to the clock period model and a resource model
- [Sec. 5.5] Additional results showing ways to use the model for the three applications across MTJ, FPGA, and ASIC technologies

Outline

- Introduction
- MTJ Read Circuit
- Hardware Virtualization
- Conclusions & Future Work

Conclusions

- Designed, fabricated, and tested a resistance-to-voltage read circuit with a continuous read
  - Resilient to high input capacitance
  - This now allows us to sense MTJs with magnetic clocking to distribute the clock signal

- Applied C-slow to virtualize hardware
  - This now allows us to effectively utilize magnetologic
  - This is effective across MTJ, FPGA, and ASIC technologies

- Developed a queueing model for virtualized, deeply-pipelined hardware
  - Useful for design guidance
  - This now allows us to predict and optimize the performance of virtualized hardware

Future work

- MTJ read circuit
  - Test with actual MTJs
  - Investigate oscillations observed near threshold

- Hardware virtualization
  - Use a general arrival process
  - Expand the queueing model to use dynamic schedules

Acknowledgments

- Thanks to Dr. Chamberlain for advising me during my PhD
- Thanks to Dr. Gruev for co-advising me
- Thanks to our collaborators at Oregon State University for introducing us to MTJs
- Thanks to committee members, fellow students, and the rest of the CSE department

References

- Lin et al., “45nm low power CMOS logic compatible embedded STT MRAM utilizing a reverse-connection 1T/1MTJ cell,” in IEEE Int’l Electron. Devices Mtg., 2009.
- Lee et al., “A full adder design using serially connected single-layer magnetic tunnel junction elements,” *IEEE Trans. on Electron. Devices*, 55(3), 2008.
- Leiserson and Saxe, “Retiming synchronous circuitry,” *Algorithmica*, 6, 1991.
- Engelbrecht, Jander, Dhagat, and **Hall**, “A toggle MRAM bit modeled in Verilog-A,” *Solid-State Electron.*, 54(10), 2010.
- **Hall**, Gruev, and Chamberlain, “Noise analysis of a current-mode read circuit for sensing magnetic tunnel junction resistance,” in IEEE Int’l Symp. on Circuits and Syst., 2011.
- **Hall**, Gruev, and Chamberlain, “Performance of a resistance-to-voltage read circuit for sensing magnetic tunnel junctions,” in IEEE Int’l Midwest Symp. on Circuits and Syst., 2012.
- **Hall** and Chamberlain, “Performance modeling of virtualized custom logic computations,” in 24th Great Lakes Symp. on VLSI, 2014.
- **Hall** and Chamberlain, “Performance modeling of virtualized custom logic computations,” in IEEE 25th Int’l Conf. on Application-specific Systems, Architectures and Processors, 2014.