A DECOMPOSITION-BASED CROSSBAR SOLVER AND FPGA-ACCELERATED IMPLEMENTATION

Suyash Vardhan Singh*

Anzhelika Kolinko*

Md. Hasibul Amin*

Ramtin Zand

Jason D. Bakos

UNIVERSITY OF South Carolina

Heterogeneous and Reconfigurable Computing Group

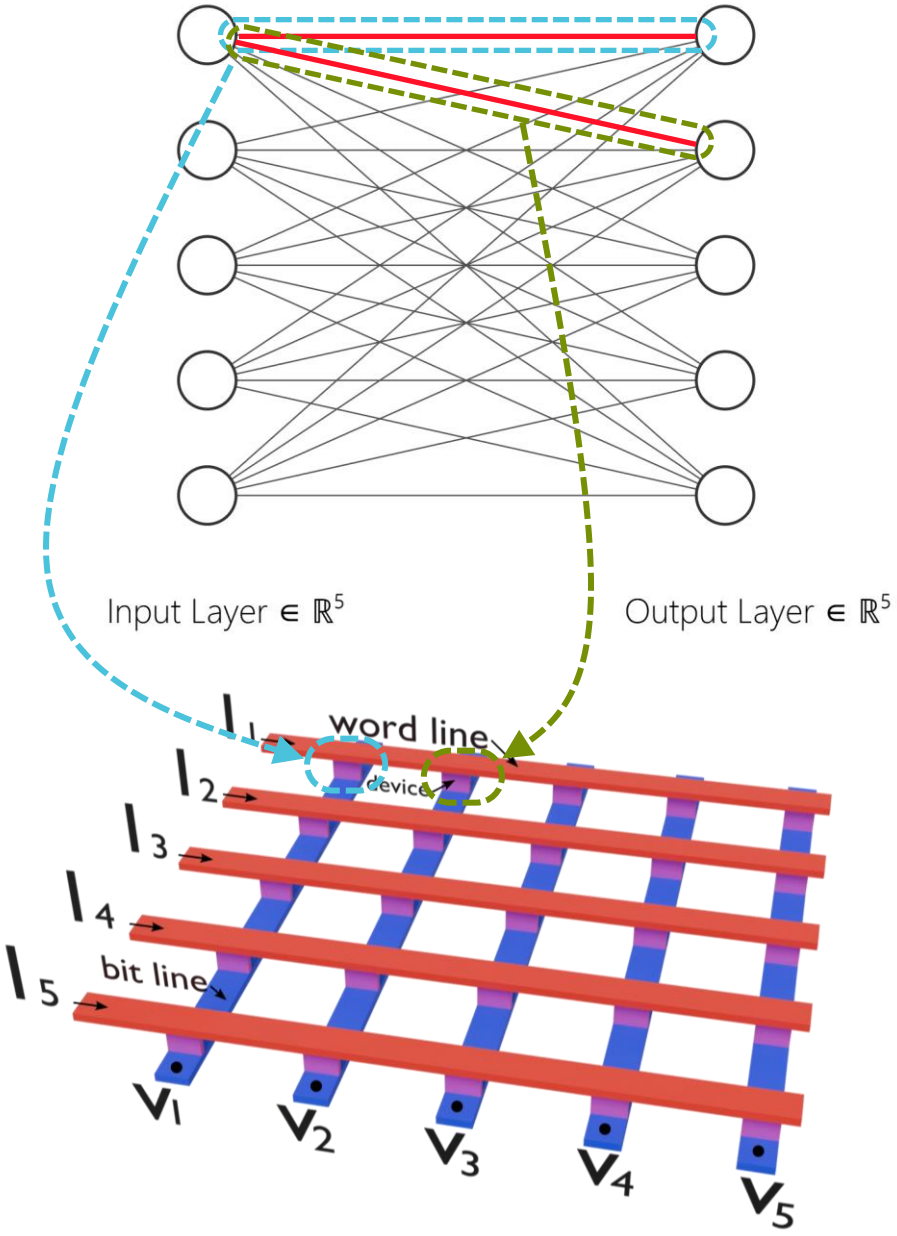
Intelligent Circuits, Architectures, and Systems Lab

*student authors

This material is based upon work supported by the National Science Foundation under Grant No. 2409697.

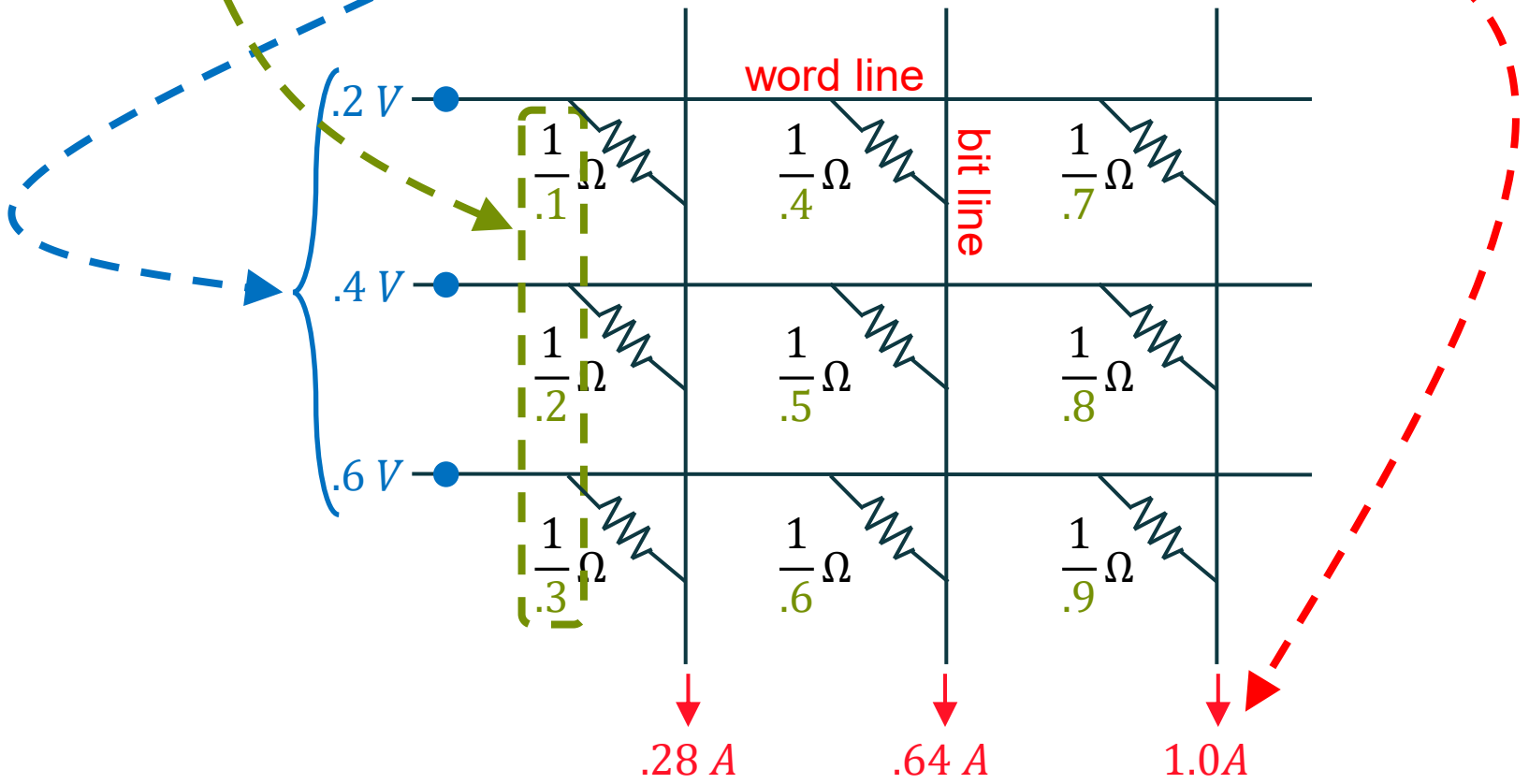
MEMRISTIVE CROSSBAR

- Explore using memristive crossbars to perform linear operator (MVM) in the analog domain
  - Can deliver transformative energy efficiency vs systolic array
- In practice, requires careful crossbar sizing and mapping
- Need fast simulation tools to evaluate impact of electrical and device effects



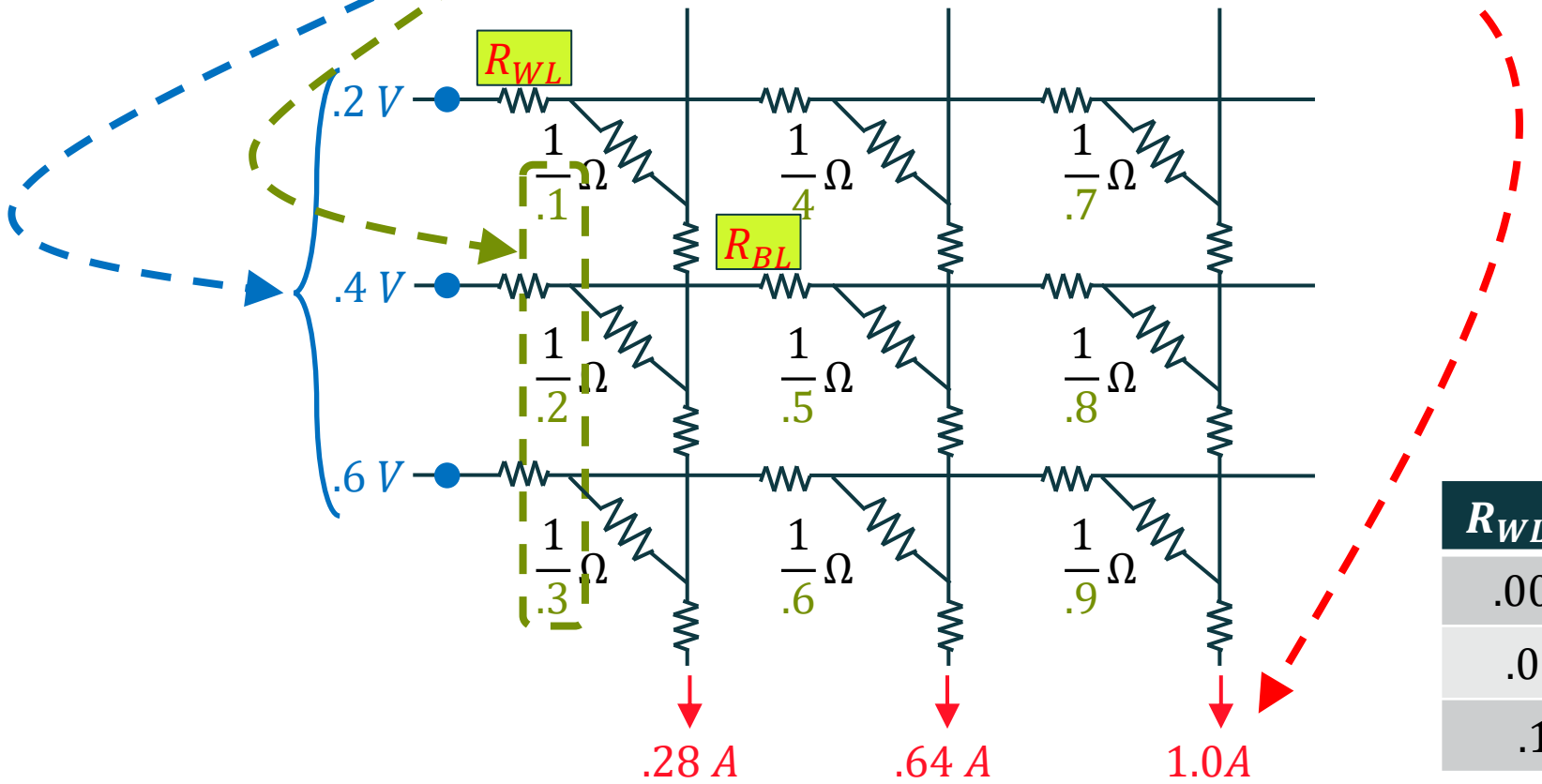
MEMRISTIVE CROSSBAR

$$\begin{bmatrix} .1 & .2 & .3 \\ .4 & .5 & .6 \\ .7 & .8 & .9 \end{bmatrix} \begin{bmatrix} .2 \\ .4 \\ .6 \end{bmatrix} = \begin{bmatrix} .28 \\ .64 \\ 1.0 \end{bmatrix}$$



MEMRISTIVE CROSSBAR

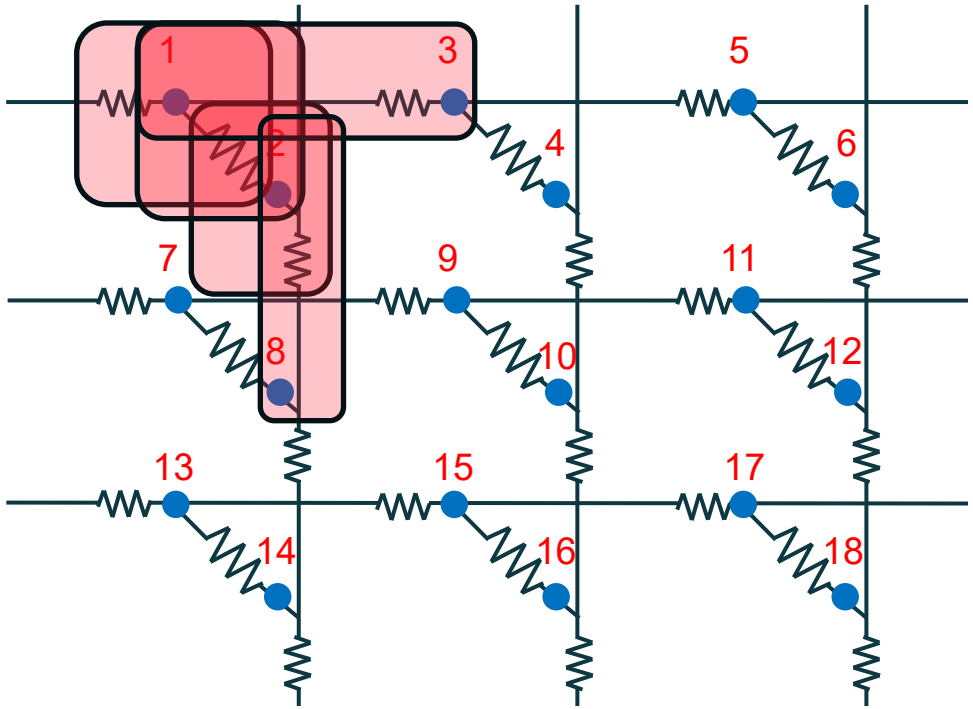
$$\begin{bmatrix} .1 & .2 & .3 \\ .4 & .5 & .6 \\ .7 & .8 & .9 \end{bmatrix} \begin{bmatrix} .2 \\ .4 \\ .6 \end{bmatrix} = \begin{bmatrix} .28 \\ .64 \\ 1.0 \end{bmatrix}$$



R_{WL}/R_{BL}	Mean % error
.001 Ω	.62%
.01 Ω	5.8%
.1 Ω	36.1%

SOLVER

- Construct Laplacian matrix:



- Crossbar size: m rows and n columns
- Laplacian matrix size = $2mn \times 2mn$

$$L_{i,j} = \begin{cases} \sum_{k \neq i} G_{i,k}, & if\ i = j \\ -G_{i,k}, & if\ i \neq j\ and\ resistor\ between\ i\ and\ j \\ 0 & otherwise \end{cases}$$

$$L_{1,1} = G_{WL} + G_{MR(1,1)}$$

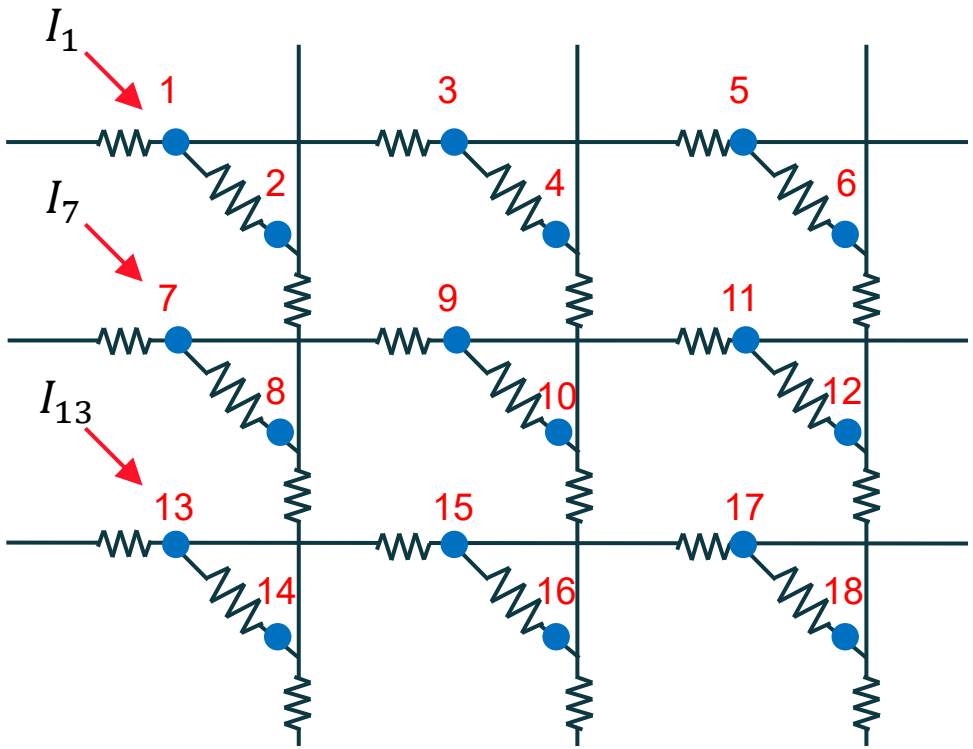
$$L_{2,2} = G_{MR(1,1)} + G_{BL}$$

$$L_{1,2} = L_{2,1} = -G_{MR(1,1)}$$

$$L_{1,3} = L_{3,1} = -G_{WL}$$

$$L_{2,8} = L_{8,2} = -G_{BL}$$

SOLVER



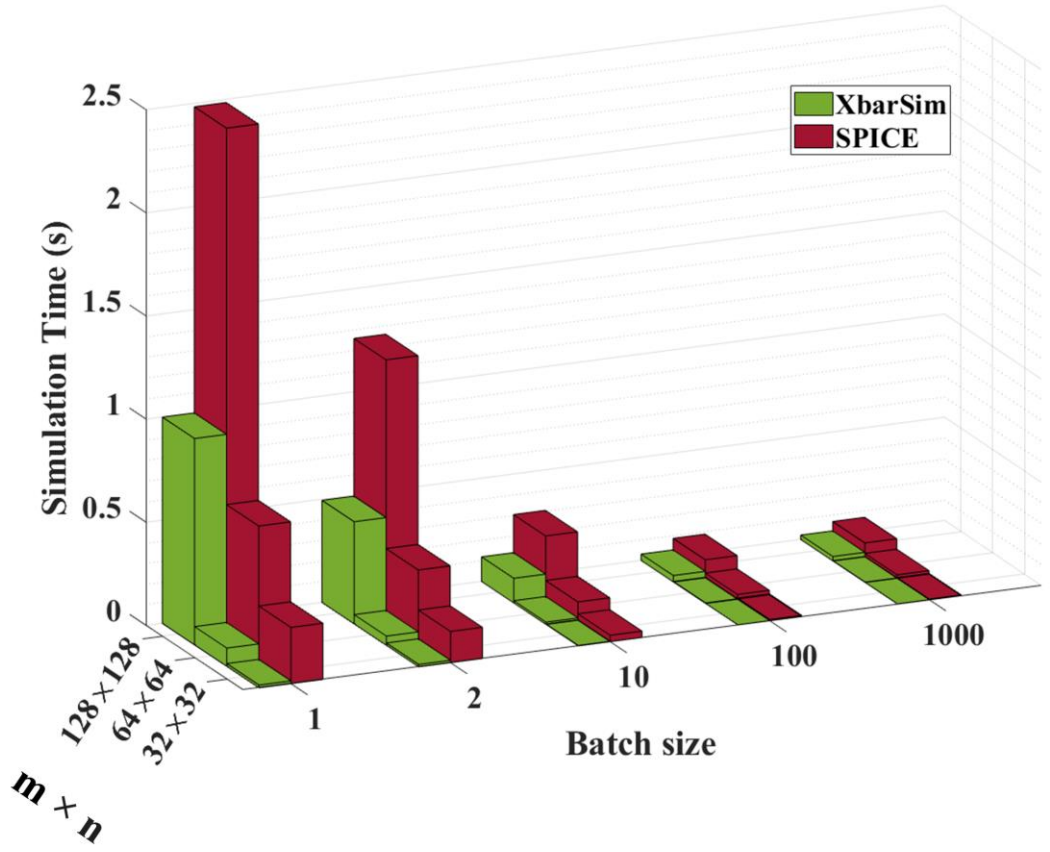
$$I_{2,\ldots,6} = I_{7,\ldots,12} = I_{14,\ldots,18} = 0$$

- Solve node voltages:
  - $GV = I$
- Decompose: $G = LU$
  - $LUV = I$
  - Set $y = UV$
- Forward substitution: solve $Ly = I$
- Backsubstitution: solve $UV = y$

BATCH SOLVING

- LU decompositive is expensive compared with forward/back substitution

- Neuroarchitecture search requires solving in batches
  - LU decomposion performed once
  - Amoratize LU decomposition cost over batch of solves



FORWARD AND BACK- SUBSTITUTION

Forward substitution

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ c_1 & 1 & 0 & 0 \\ c_2 & c_3 & 1 & 0 \\ c_4 & c_5 & c_6 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \end{bmatrix}$$

$$y_1 = I_1$$

$$y_2 = I_2 - c_1 y_1$$

$$y_3 = I_3 - c_2 y_1 - c_3 y_2$$

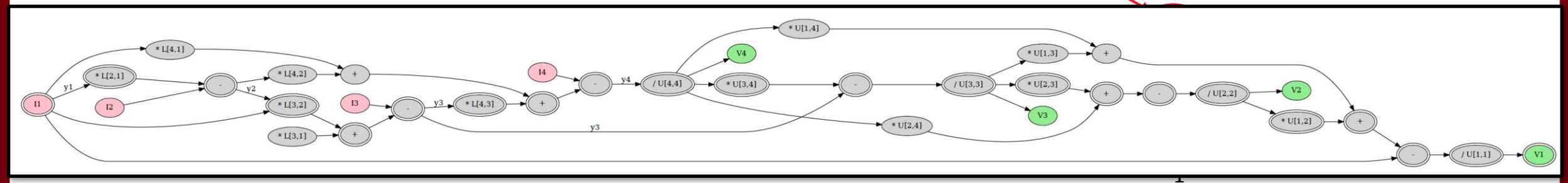
$$y_4 = I_4 - c_4 y_1 - c_5 y_2 - c_6 y_3$$

Backsubstitution

$$\begin{bmatrix} d_1 & d_2 & d_3 & d_4 \\ 0 & d_5 & d_6 & d_7 \\ 0 & 0 & d_8 & d_9 \\ 0 & 0 & 0 & d_{10} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

$$V_4 = \frac{y_4}{d_{10}}$$

$$V_3 = \frac{y_3 - d_9 V_4}{d_8}$$



PARTITIONING

- Pipeline forward and backsubstitution to process multiple solves in parallel
  - **Pipeline latency:** depends on the critical path
  - **Pipeline throughput:** can initiate pipeline every $2mn$ cycles
    - Required for reading I vector and writing V vector to off-chip memory
  - L and U matrices are stored on-chip

Xbar Size	II	IL	DP mults	DP subs/ adds	DP divides	Batch size 1000			Batch size 10000		
						cycles	time(us)	ops/cycle	cycles	time(us)	ops/cycle
2×2	8	1041	26	29	8	9041	25.40	6.97	81041	227.64	7.77
4×4	32	5502	296	235	32	37502	105.34	15.01	325502	914.33	17.30
8×8	128	32889	1934	1943	128	160889	451.94	24.89	1312889	3687.89	30.51

HARDWARE COST

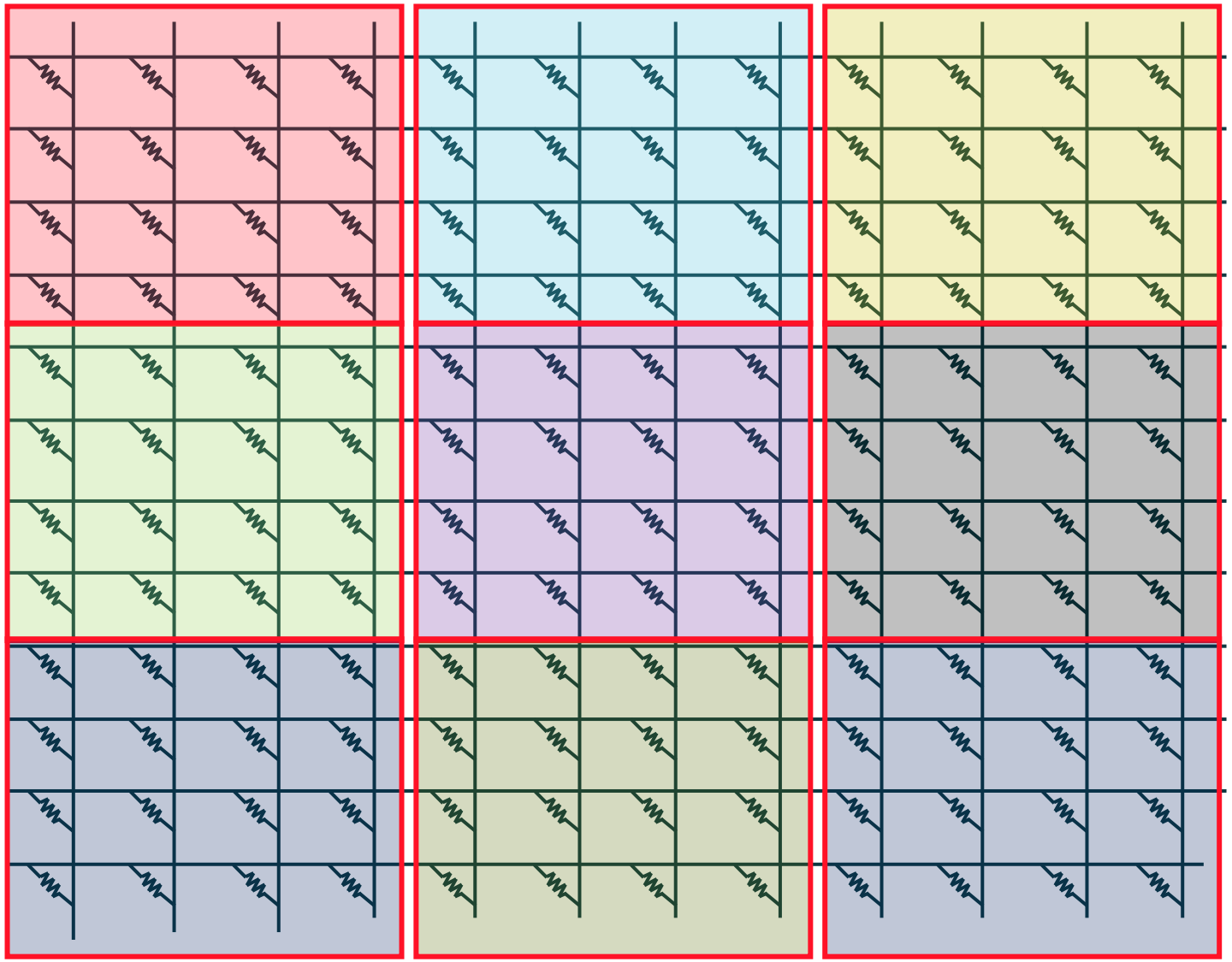
Xbar Size	LUTs	FFs	DSPs	Fmax (MHz)
2×2	9997	18234	40	356
4×4	24790	62218	80	356
8×8	98222	243221	160	356

Xbar Size	Matrix Size	L Matrix nnz	U Matrix nnz	HLS Compile Time (sec)
2×2	8×8	21	21	16.87
4×4	32×32	147	147	141.76
8×8	128×128	1095	1095	19507.3

- For pipelined solvers, HLS compile time and resource usage scales poorly
- Largest practical pipelined solver is 8x8 (32x32 matrix)

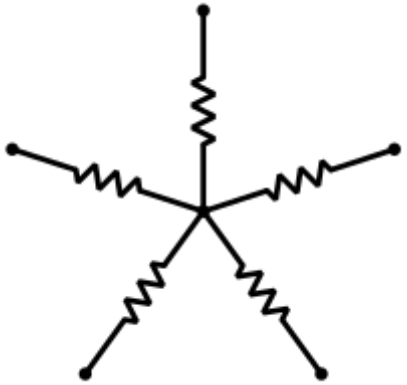
PARTITIONING

- Partitioning needed to solve larger crossbars
- Store all partition matrices on chip
- Solve sequentially

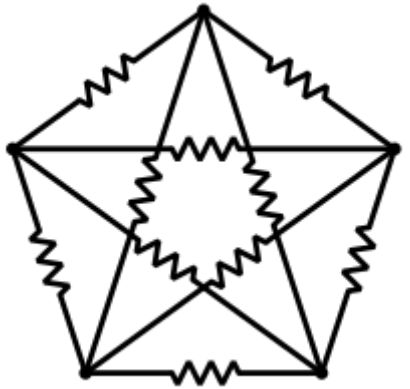


PARTITIONING

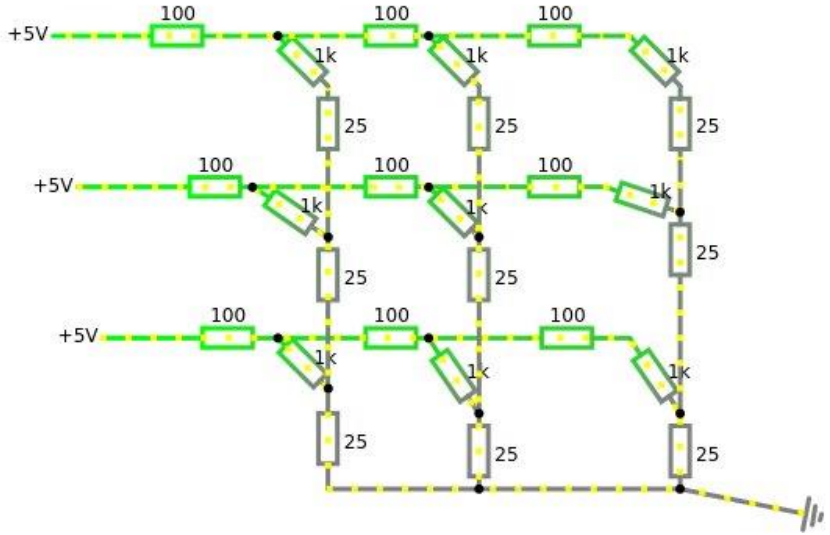
- Star-mesh transform:



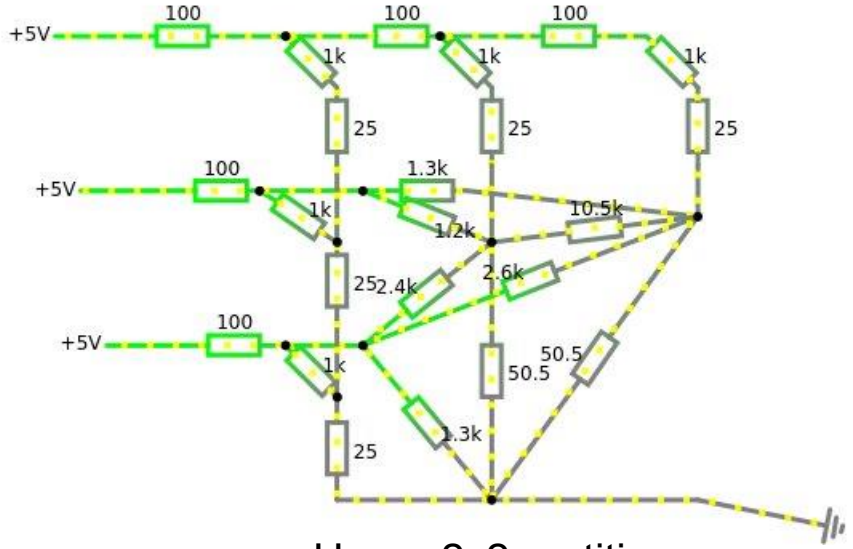
6 nodes
5 resistors



5 nodes
10 resistors

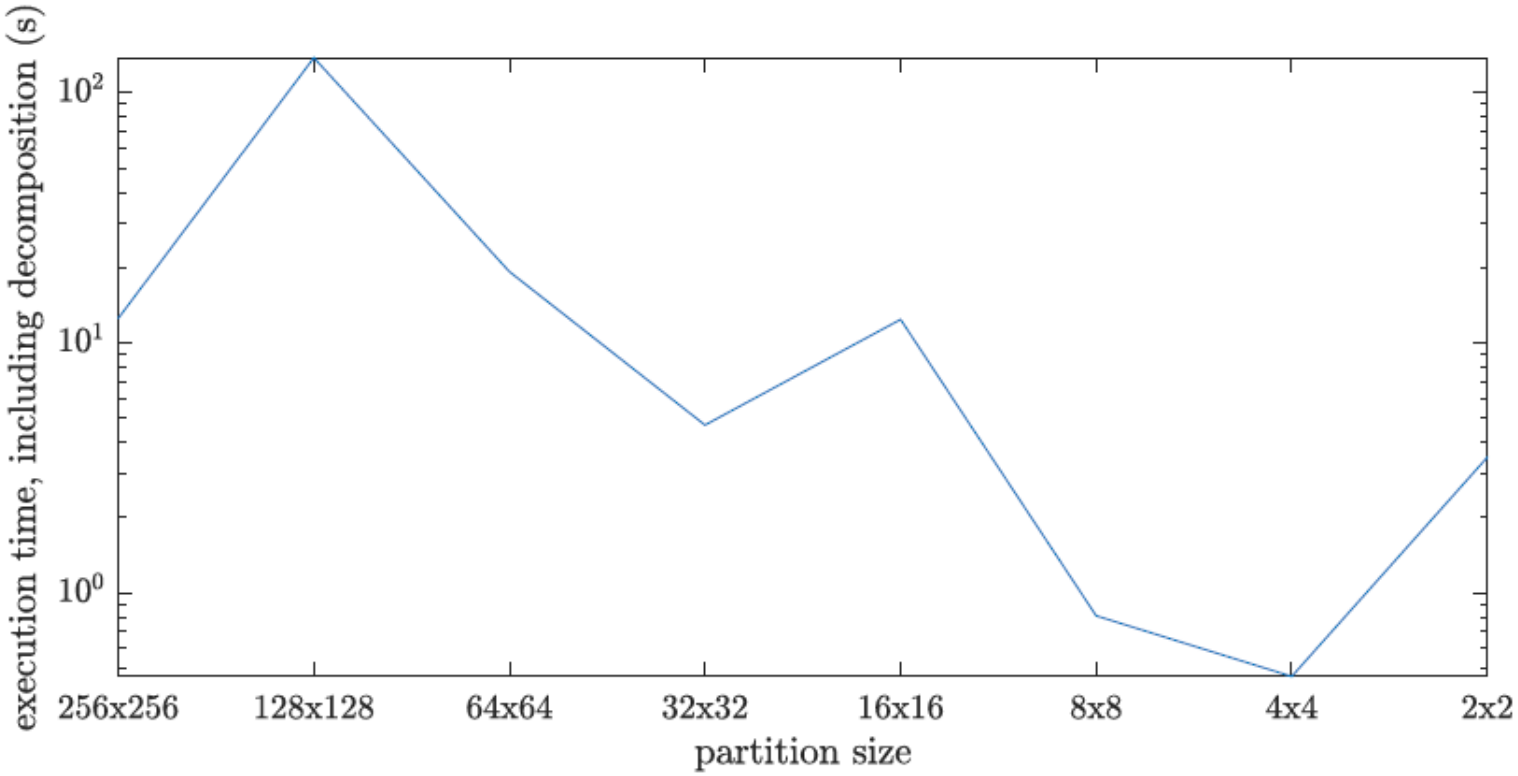


3x3 crossbar
18x18 Laplacian



Upper 2x2 partition
8x8 Laplacian
Provides $V_1, \ldots, V_8$

PARTITIONING PERFORMANCE



PERFORMANCE RESULTS

Xbar size	Matlab Intel Xeon Gold 5220R				FPGA * (2×2 partitions)					
	Batch size 1000		Batch size 10000		Batch size 1000		Batch size 10000		Number of partitions	BRAM requirement (fp64 values)
	Best partition size	Solve time	Best partition size	Solve time	Solve time	FPGA vs Matlab speedup	Solve time	FPGA vs Matlab speedup		
4×4	4×4	54.5 ns	4×4	33.1 ns	10.2 ns	5.34	9.11 ns	3.64	4	256
8×8	4×4	109 ns	4×4	78.2 ns	40.6 ns	2.68	36.4 ns	2.15	16	1024
16×16	4×4	456 ns	8×8	306 ns	163 ns	2.80	146 ns	2.10	64	4096
32×32	8×8	1.41 µs	4×4	1.24 µs	650 ns	2.17	583 ns	2.13	256	16384
64×64	4×4	5.57 µs	8×8	4.97 µs	2.60 µs	2.14	2.33 µs	2.13	1024	65536
128×128	4×4	22.3 µs	8×8	19.8 µs	10.4 µs	2.14	9.32 µs	2.12	4096	262144

* Use 2x2 to minimize hardware requirement

CONCLUSIONS

- Decomposition-based solver for crossbar arrays
  - Support for batching and partitioning
  - FPGA-based implementation

- Objective: deploy as part of a FPGA-based whole-system emulator

THANK YOU!
Q&A

Molinaroli College of Engineering and Computing
UNIVERSITY OF SOUTH CAROLINA