The Cell Matrix Computing Architecture

Nicholas J. Macias
Cell Matrix Corporation

Cell Matrix Characteristics

Cell Matrix Characteristics

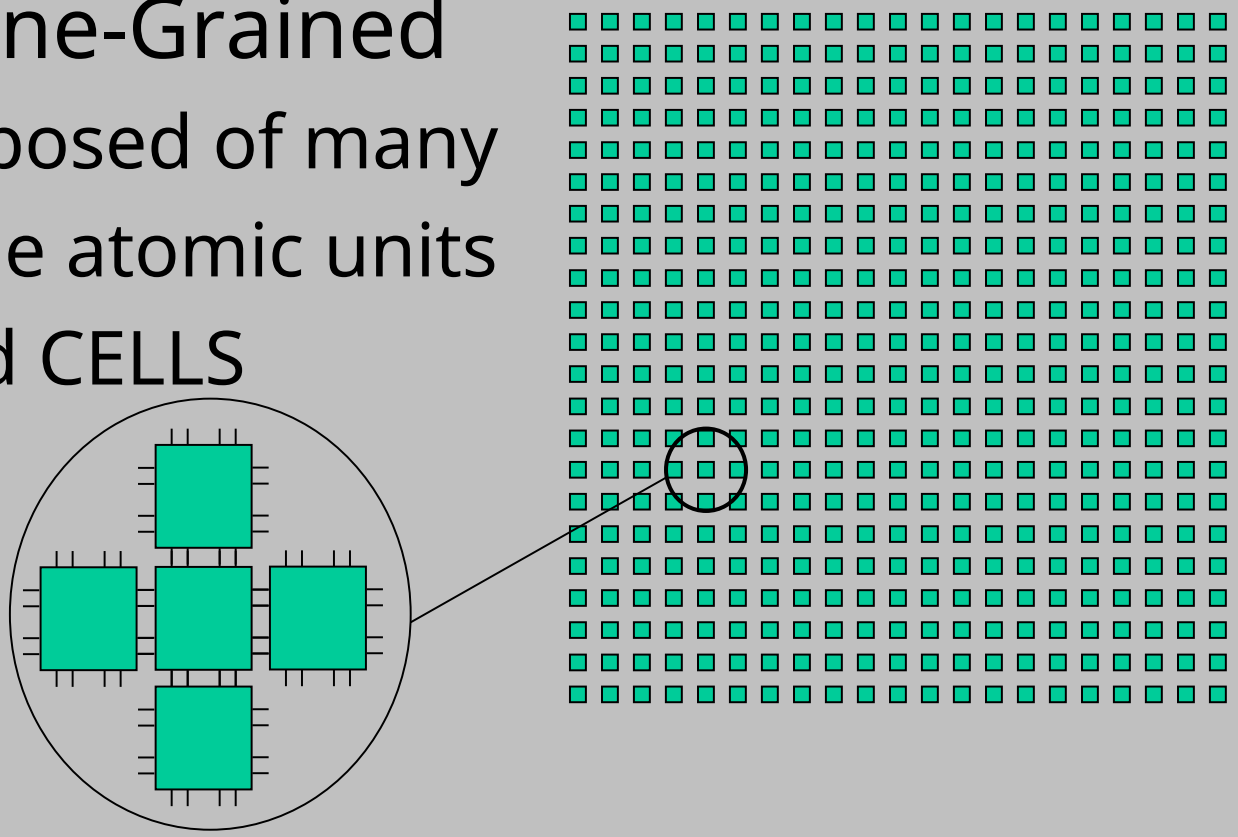
- Reconfigurable Device

Cell Matrix Characteristics

- Self-configurable Device

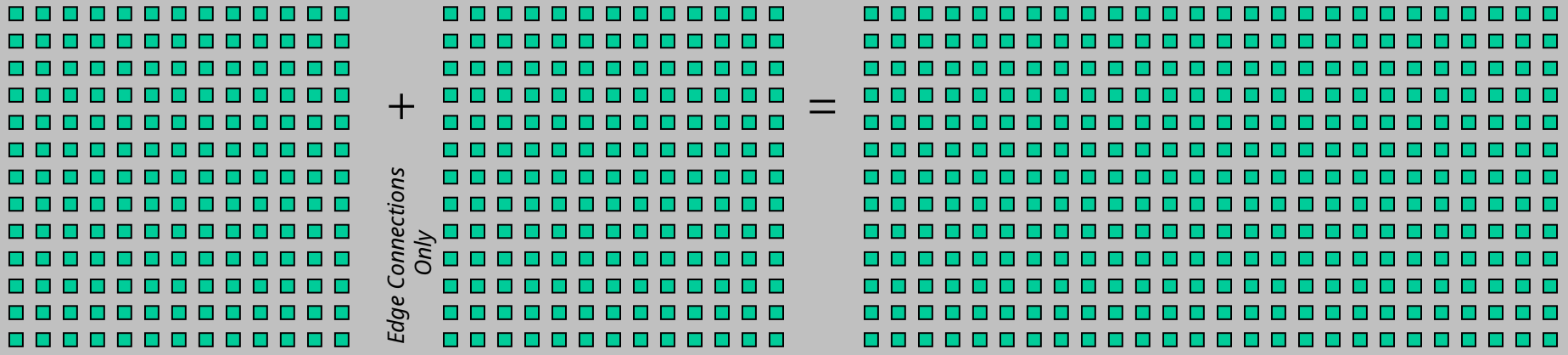
Cell Matrix Characteristics

- Self-configurable Device
- Very Fine-Grained
  - Composed of many simple atomic units called CELLS



Cell Matrix Characteristics

- Self-configurable Device
- Very Fine-Grained
- Perfectly Scalable

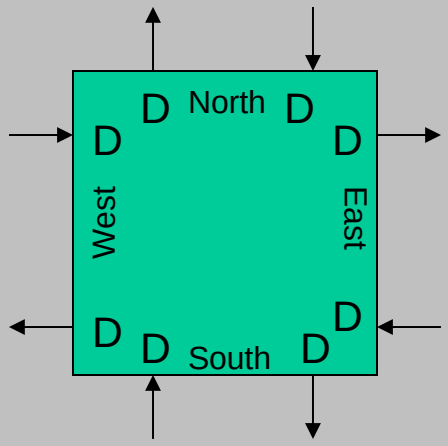


Cell Matrix Characteristics

- Self-configurable Device
- Very Fine-Grained
- Perfectly Scalable
- Extremely Parallel

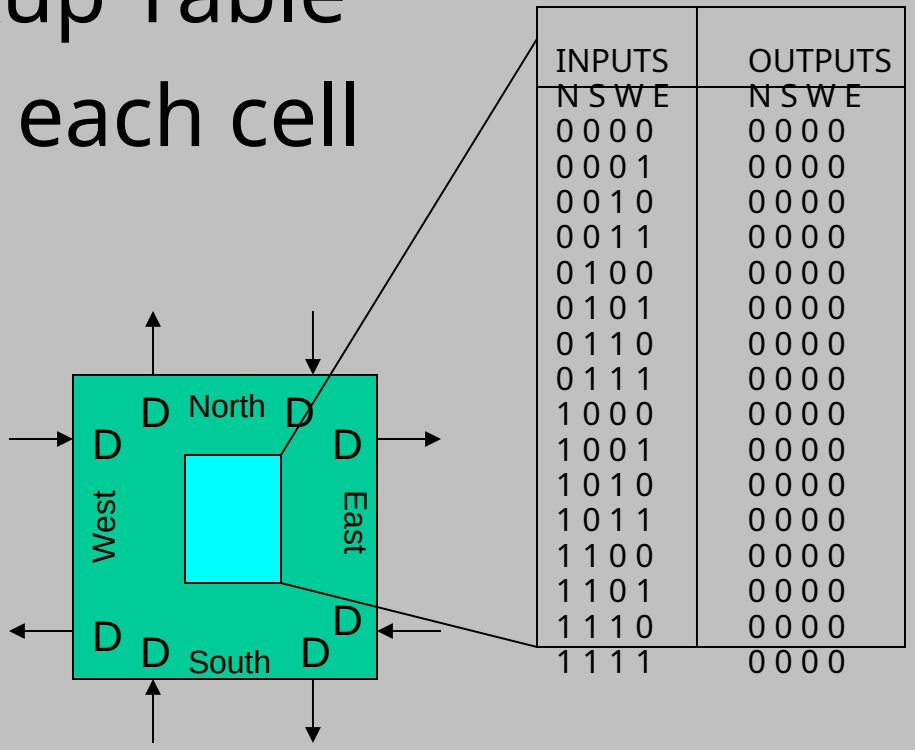
Cell-Level Behavior

- Basic processing element called a *Cell*
- Cell has 4 inputs and 4 outputs



Specifying Cell Behavior

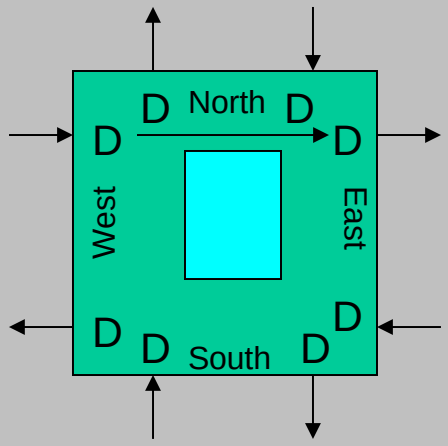
- Add a Lookup Table (LUT) inside each cell



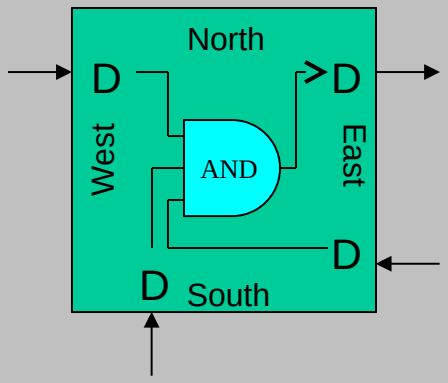
The LUT Fully Specifies the Cell's Behavior

- DW_{in} --> DE_{out}

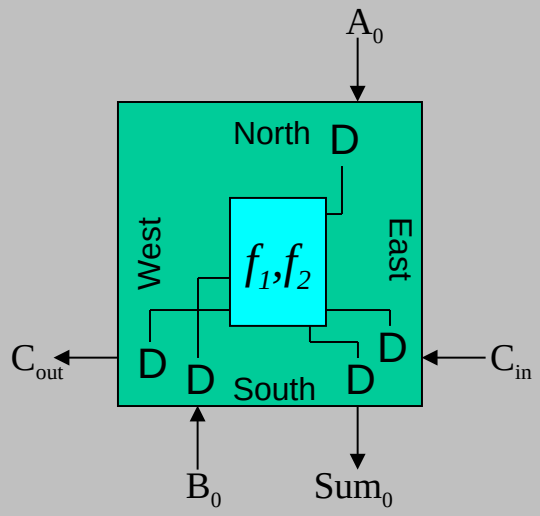
  This is a simple wire



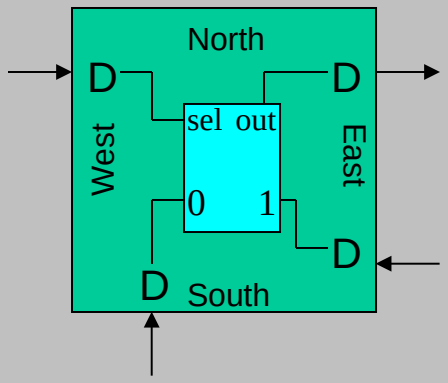
INPUTS N S W E	OUTPUTS N S W E
0 0 0 0	0 0 0 0
0 0 0 1	0 0 0 0
0 0 1 0	0 0 0 1
0 0 1 1	0 0 0 1
0 1 0 0	0 0 0 0
0 1 0 1	0 0 0 0
0 1 1 0	0 0 0 1
0 1 1 1	0 0 0 1
1 0 0 0	0 0 0 0
1 0 0 1	0 0 0 0
1 0 1 0	0 0 0 1
1 0 1 1	0 0 0 1
1 1 0 0	0 0 0 0
1 1 0 1	0 0 0 0
1 1 1 0	0 0 0 1
1 1 1 1	0 0 0 1



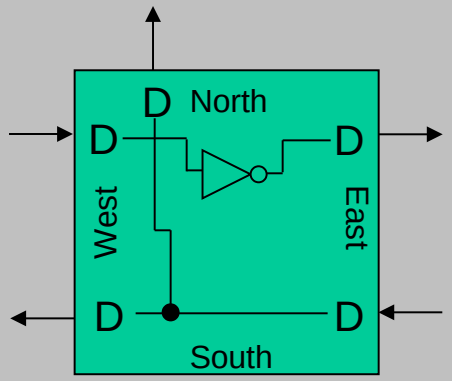
3-Input AND Gate



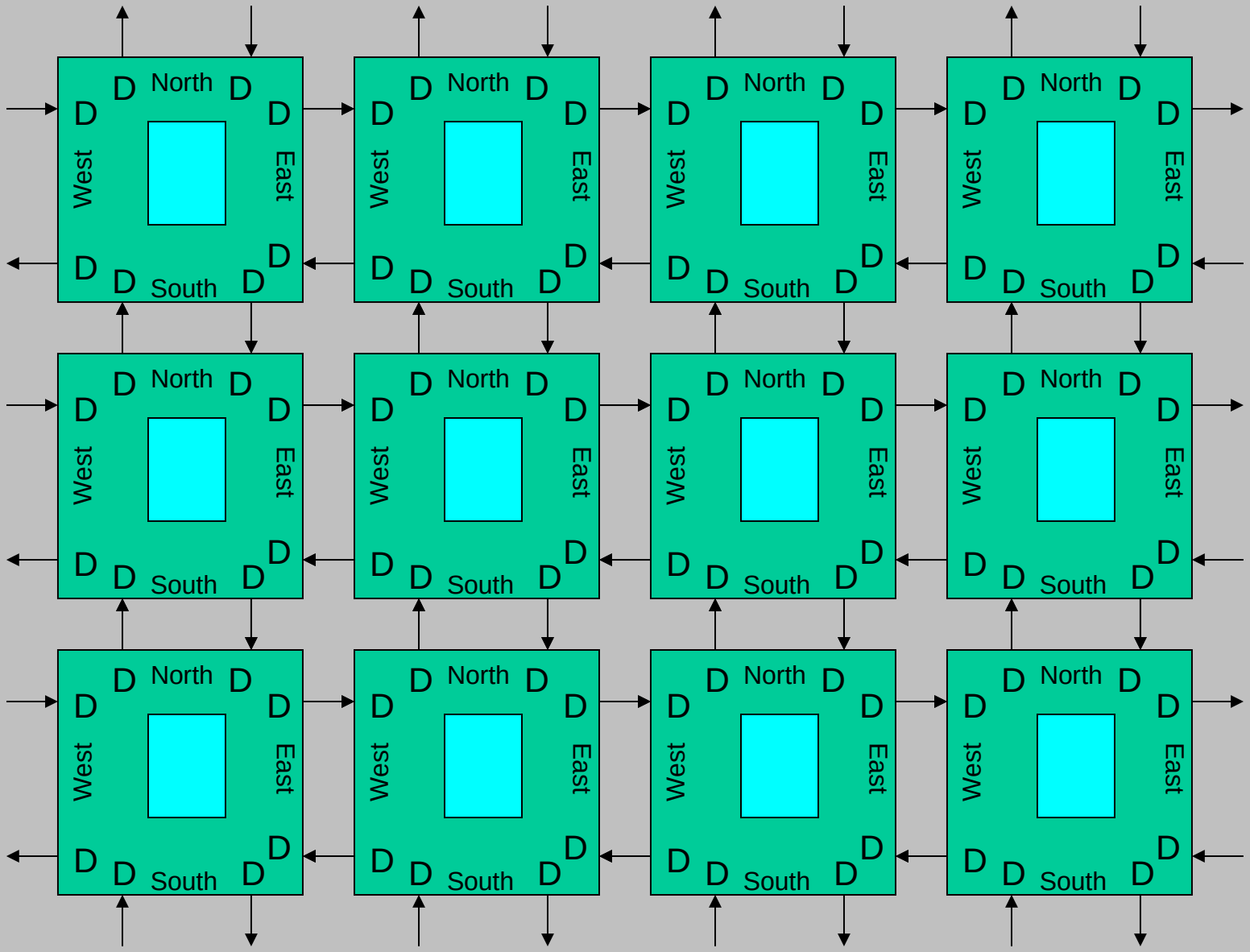
1-Bit Full Adder



2-1 Selector

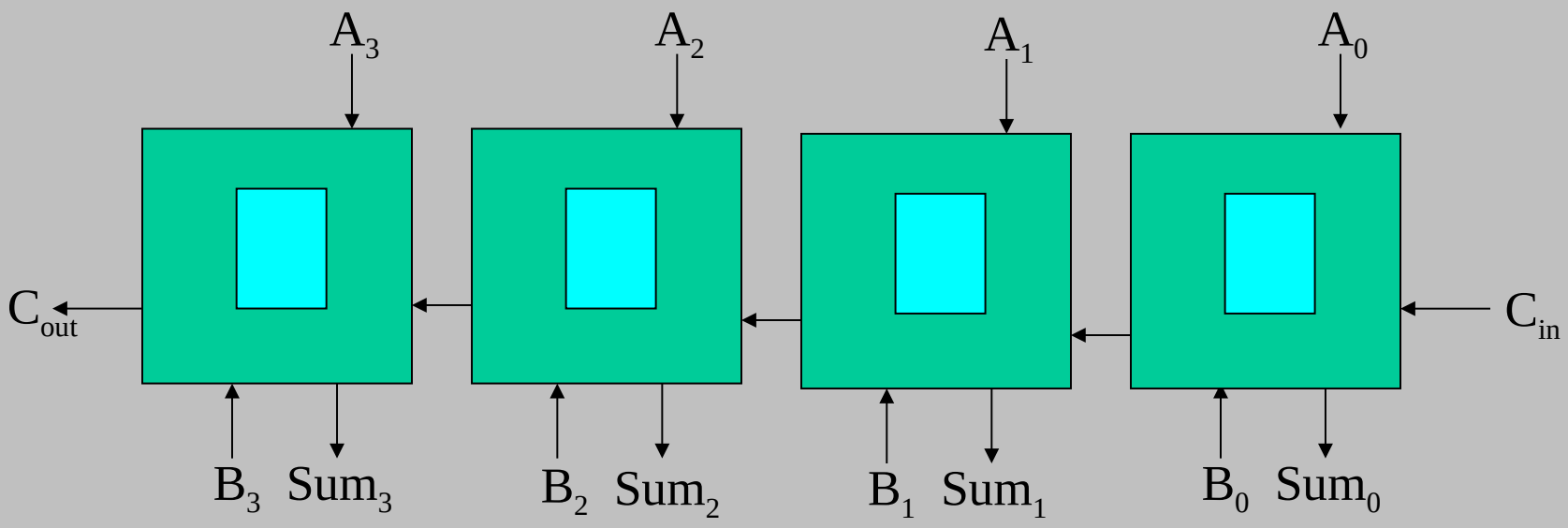


Inverter + T-Junction

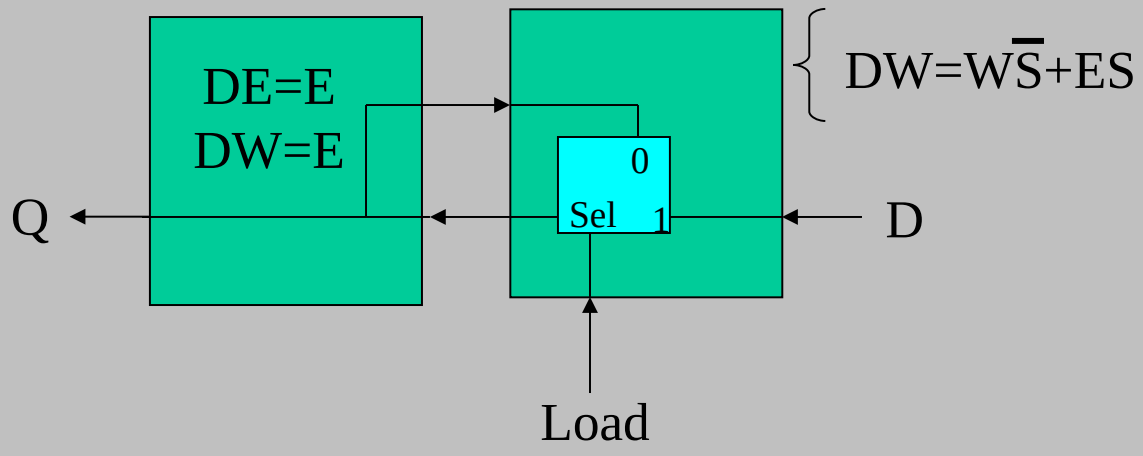


3x4 Grid of 4-Sided 2-D Cells

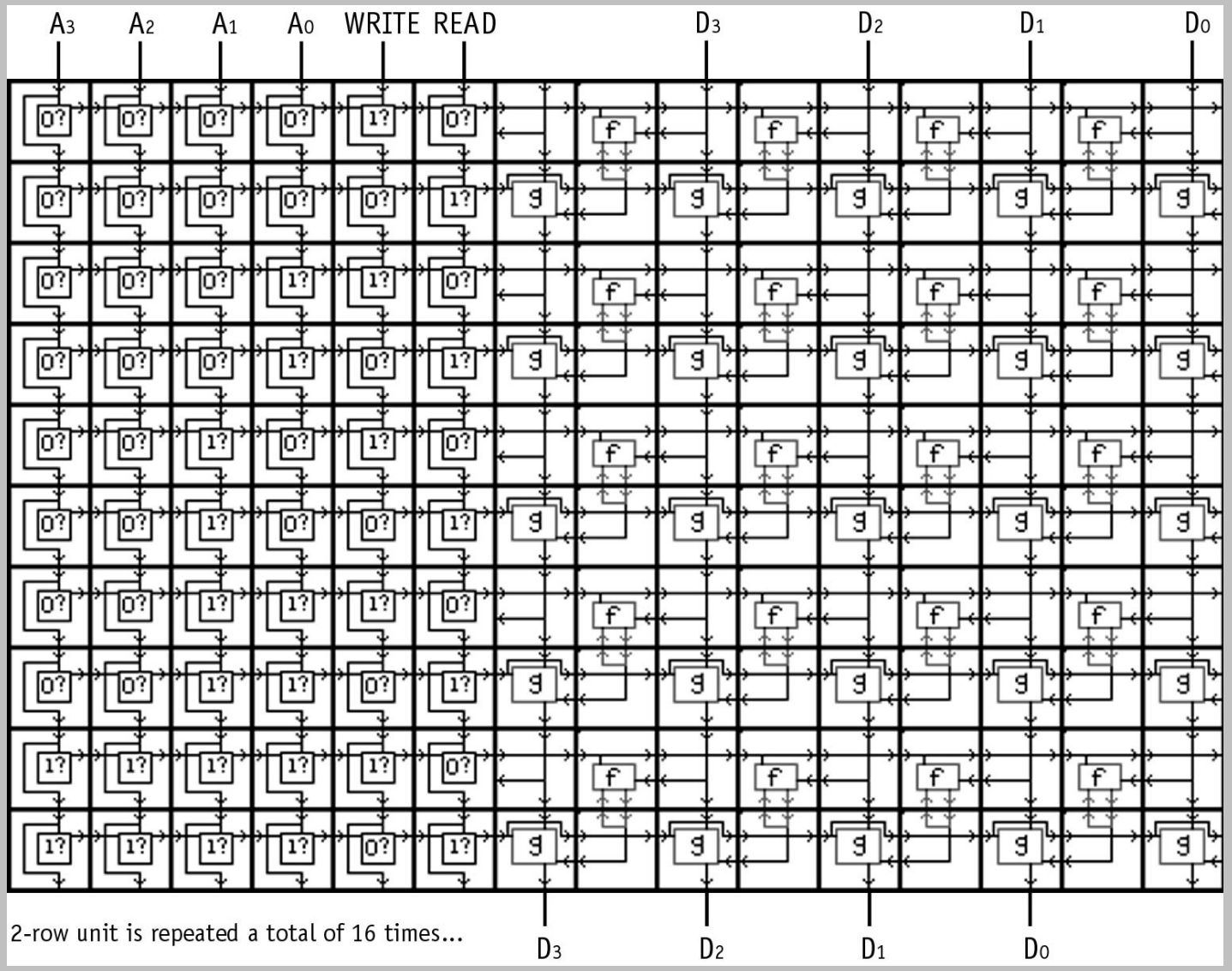
Example: Four-Bit Adder



Example: D-Flip Flop

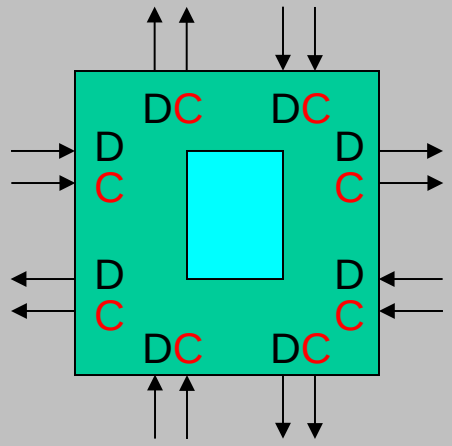


Example: Memory



How is a Cell Configured?

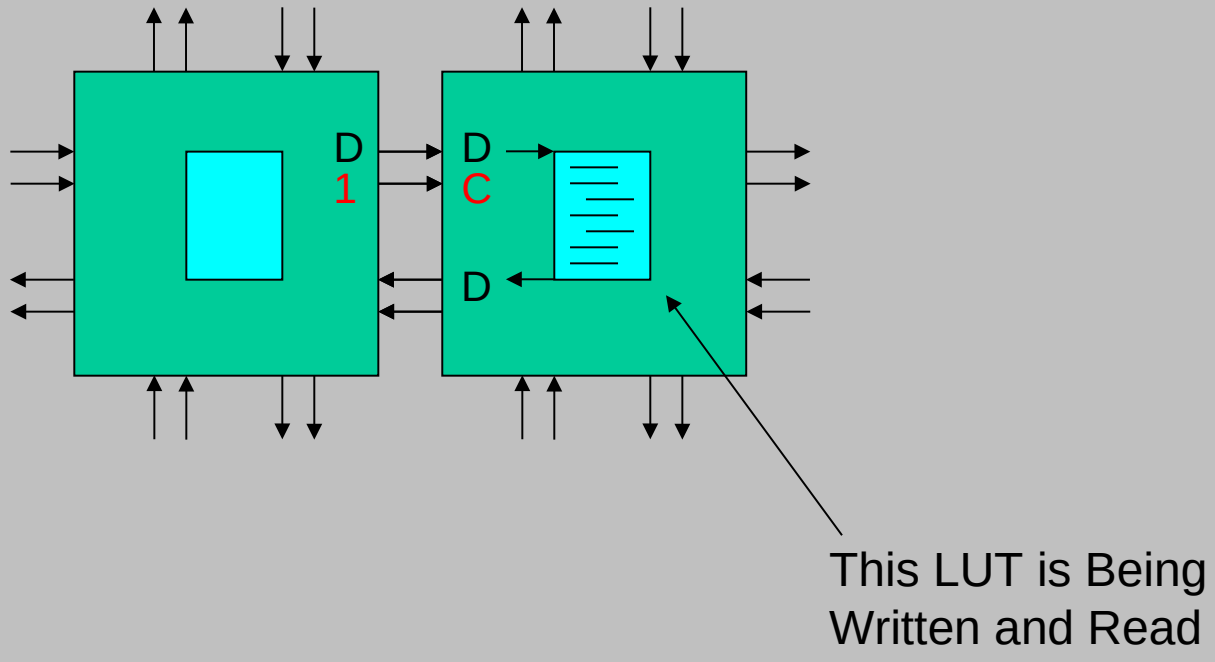
- If C=0, everything is as before



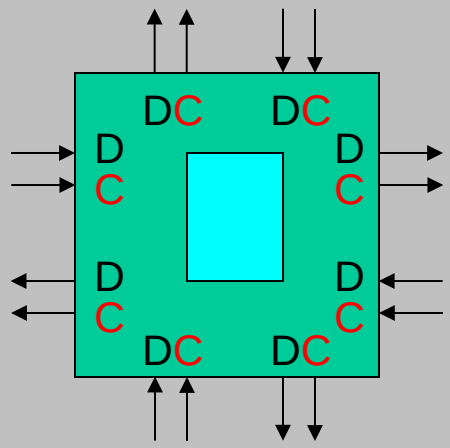
INPUTS N S W E	OUTPUTS N S W E
0 0 0 0	0 0 0 0
0 0 0 1	0 0 0 0
0 0 1 0	0 0 0 0
0 0 1 1	0 0 0 0
0 1 0 0	0 0 0 0
0 1 0 1	0 0 0 0
0 1 1 0	0 0 0 0
0 1 1 1	0 0 0 0
1 0 0 0	0 0 0 0
1 0 0 1	0 0 0 0
1 0 1 0	0 0 0 0
1 0 1 1	0 0 0 0
1 1 0 0	0 0 0 0
1 1 0 1	0 0 0 0
1 1 1 0	0 0 0 0
1 1 1 1	0 0 0 0

LUT Transfer

- If C=1, LUTs are transferred



Add C Outputs to LUT

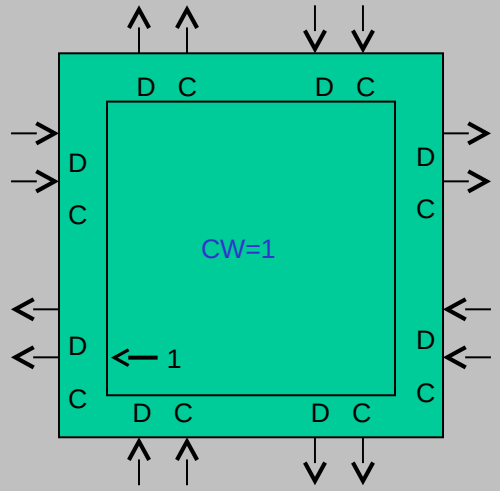


INPUTS	C-OUTPUTS	D-OUTPUTS
N S W E	N S W E	N S W E
0 0 0 0	0 0 0 0	0 0 0 0
0 0 0 1	0 0 0 0	0 0 0 0
0 0 1 0	0 0 0 0	0 0 0 0
0 0 1 1	0 0 0 0	0 0 0 0
0 1 0 0	0 0 0 0	0 0 0 0
0 1 0 1	0 0 0 0	0 0 0 0
0 1 1 0	0 0 0 0	0 0 0 0
0 1 1 1	0 0 0 0	0 0 0 0
1 0 0 0	0 0 0 0	0 0 0 0
1 0 0 1	0 0 0 0	0 0 0 0
1 0 1 0	0 0 0 0	0 0 0 0
1 0 1 1	0 0 0 0	0 0 0 0
1 1 0 0	0 0 0 0	0 0 0 0
1 1 0 1	0 0 0 0	0 0 0 0
1 1 1 0	0 0 0 0	0 0 0 0
1 1 1 1	0 0 0 0	0 0 0 0

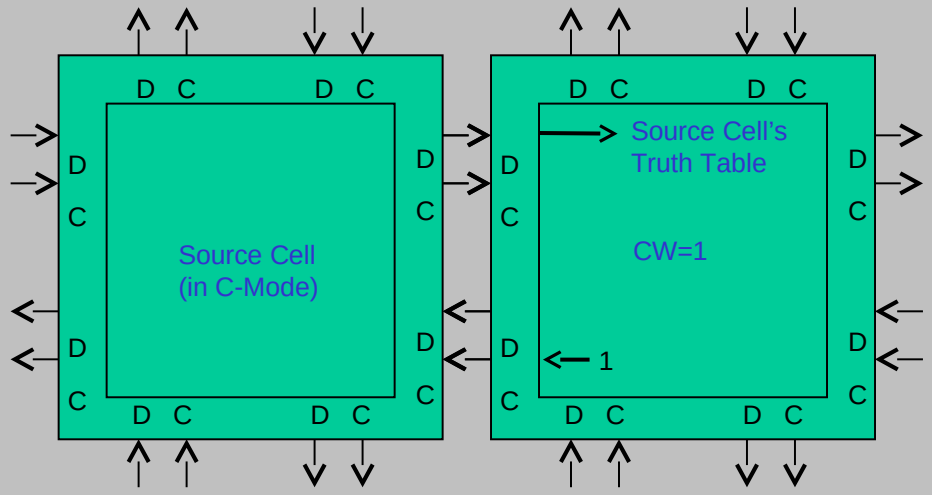
This is an Extremely Natural Construct

- Designing on a Cell Matrix is like standard digital logic design
- Added component: ability to run *circuit configuration information* through the circuits
- **Key Difference between FPGA and a Cell Matrix**

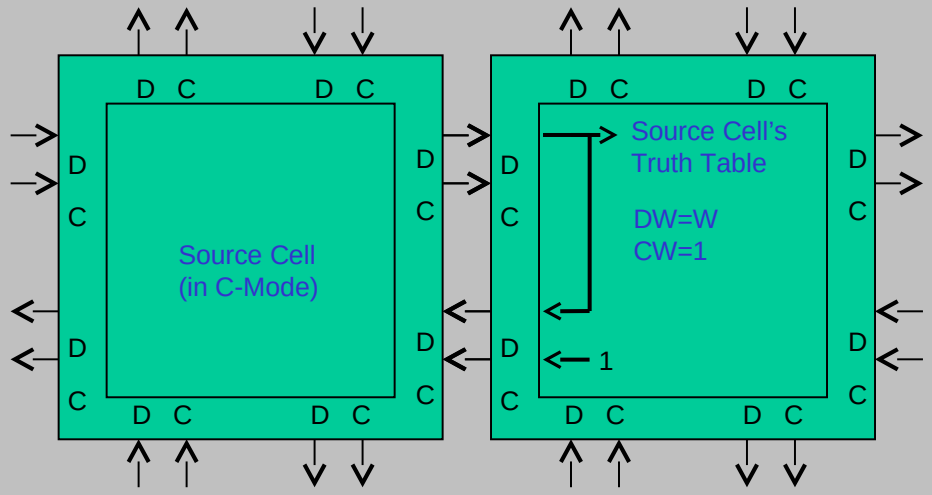
C-Mode Examples



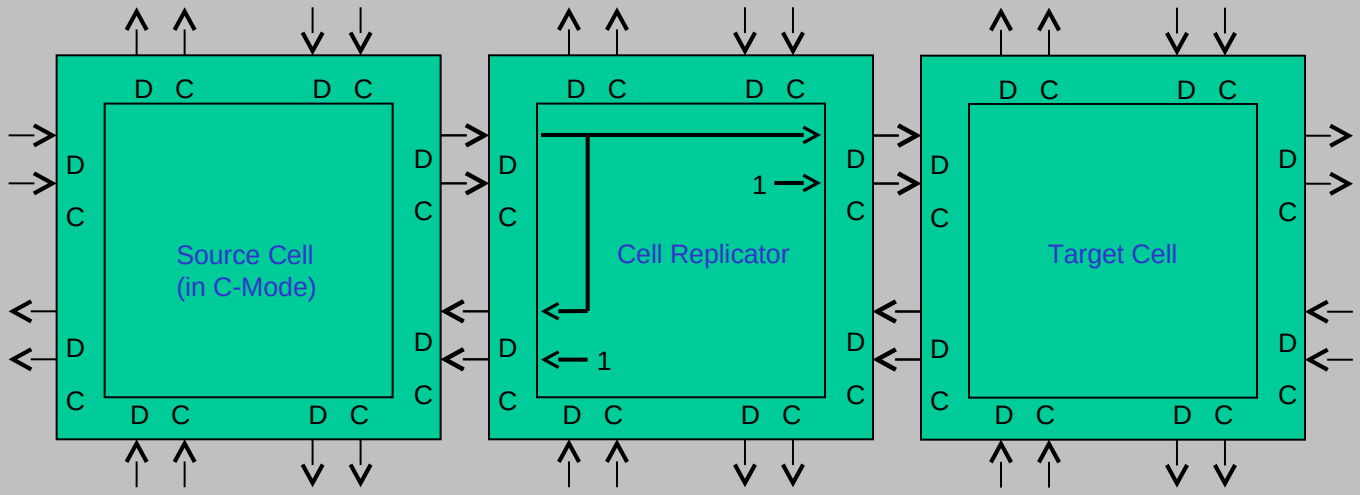
C-Mode Examples



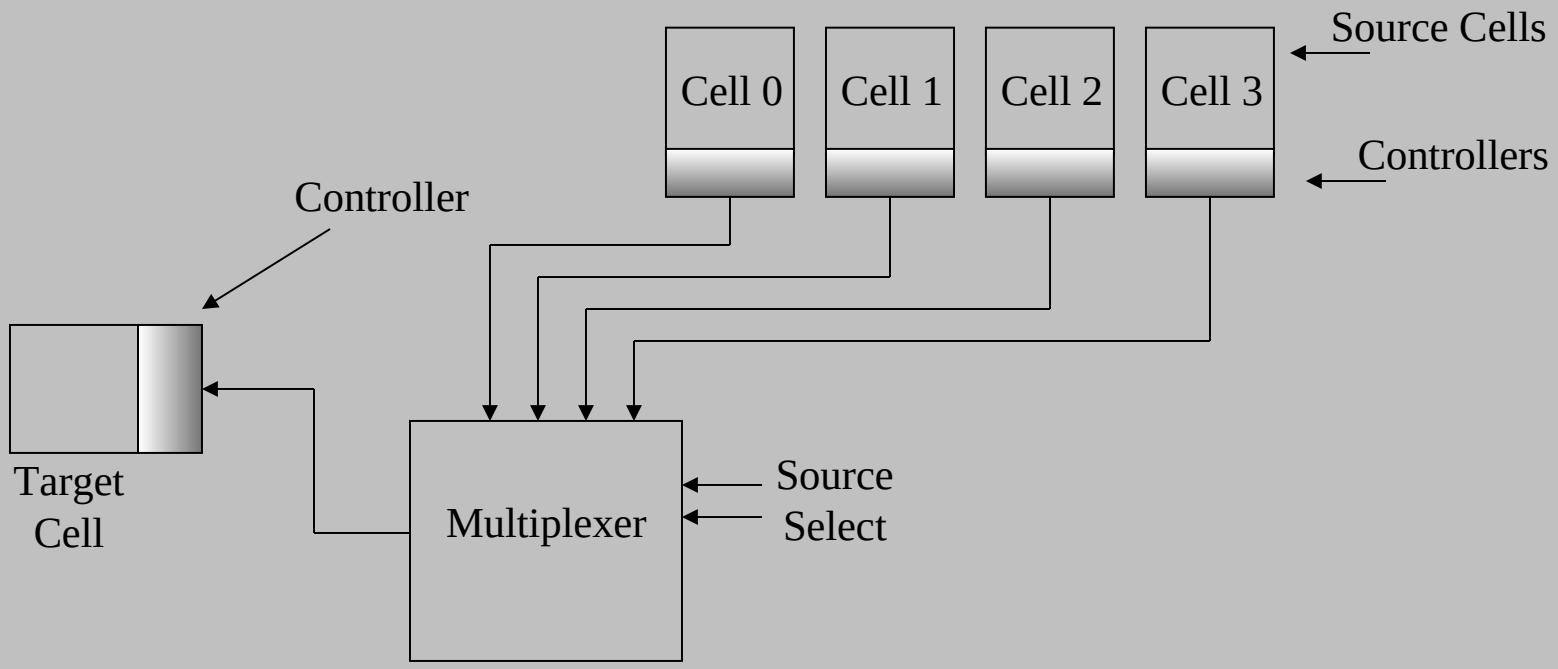
C-Mode Examples



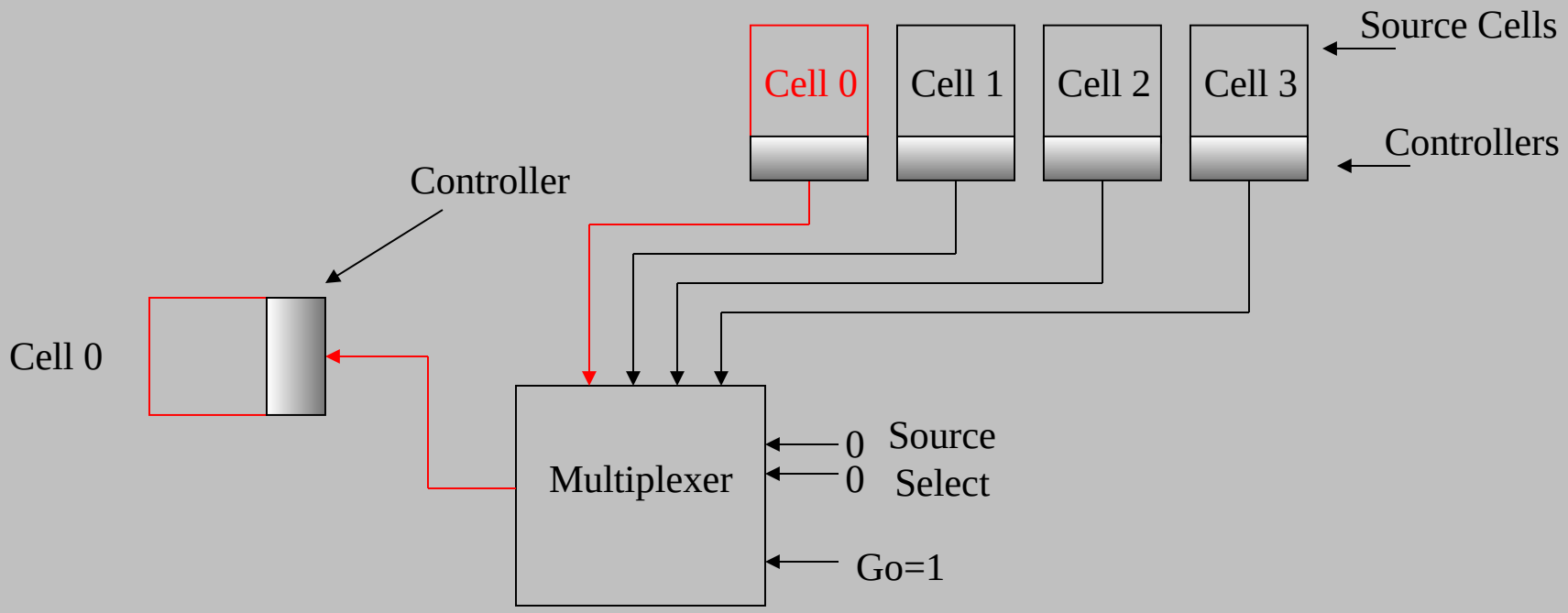
C-Mode Examples



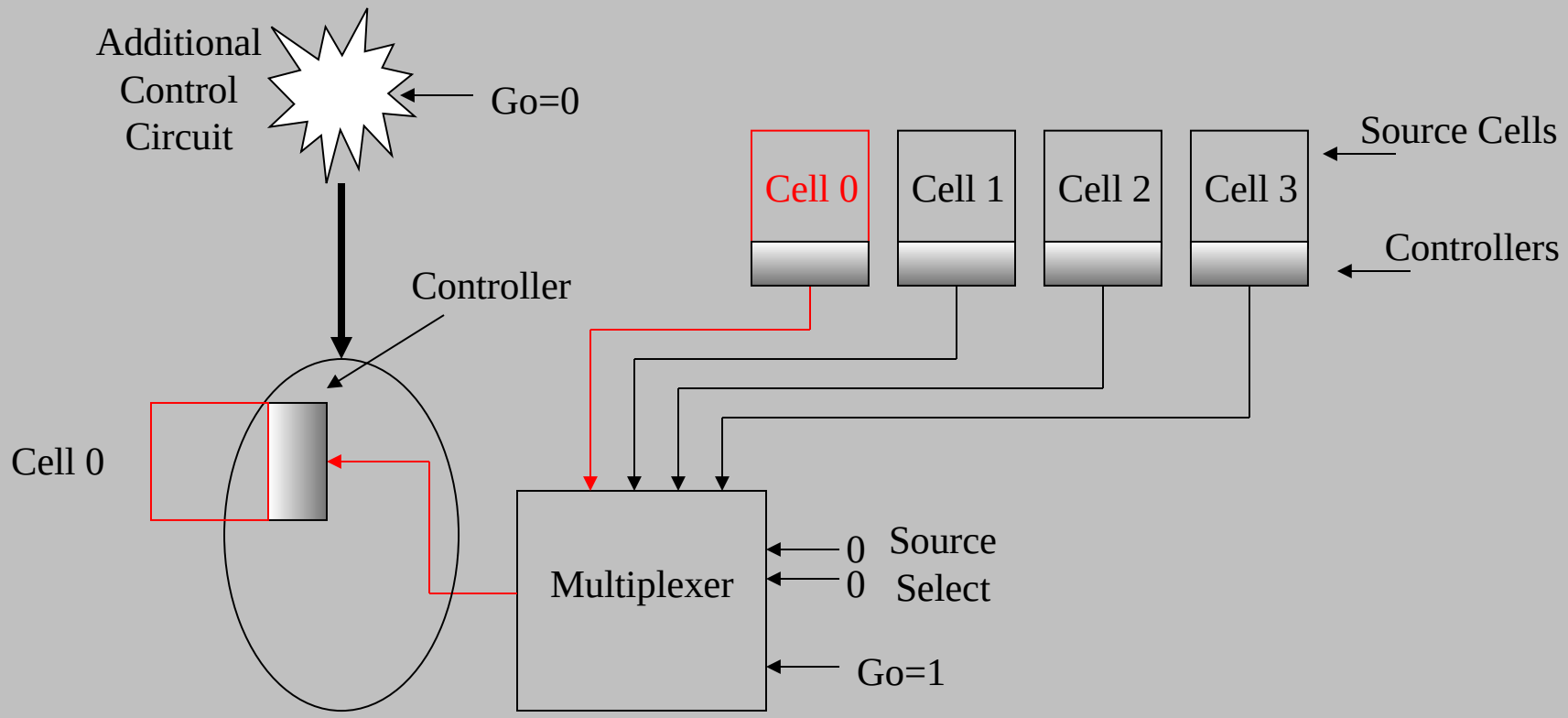
Cell Library



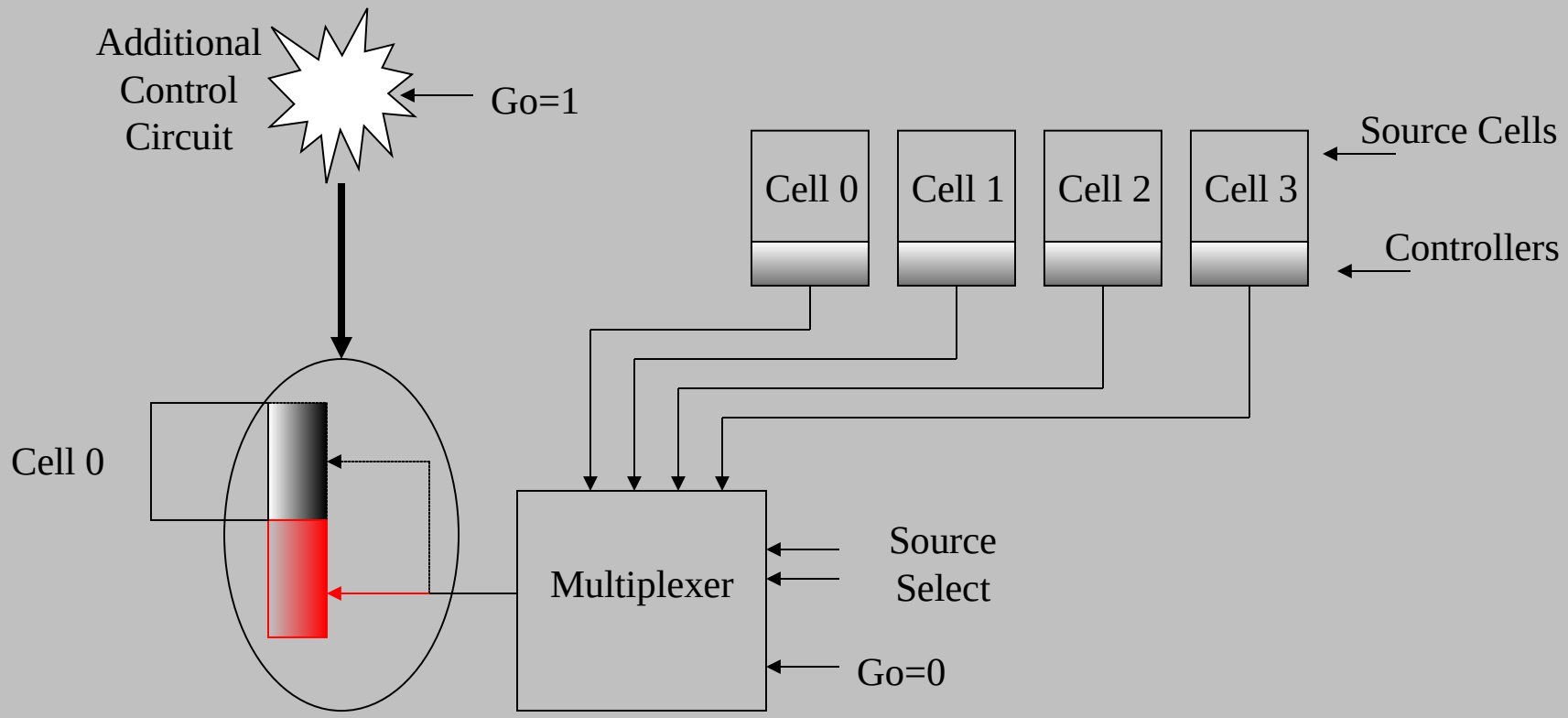
Circuit Synthesis



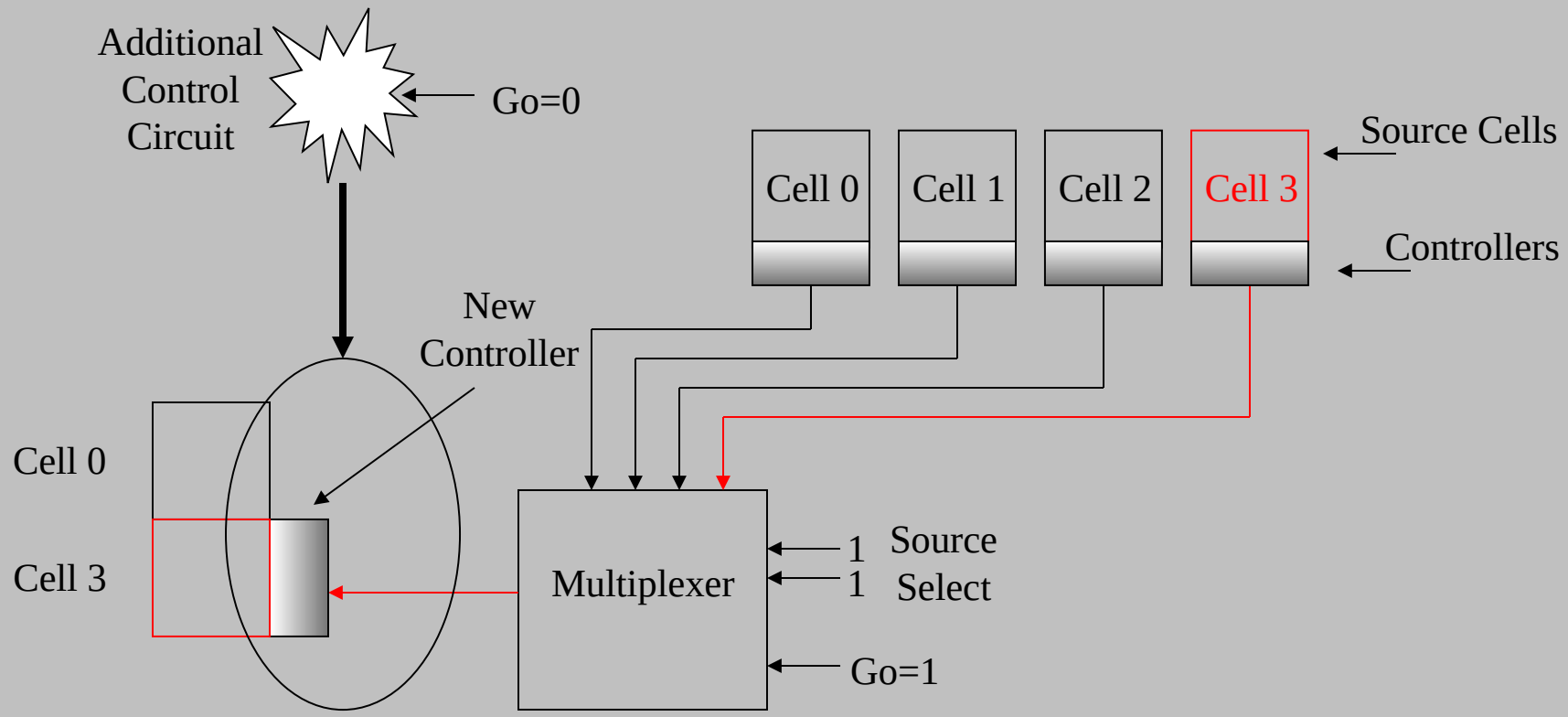
Circuit Synthesis



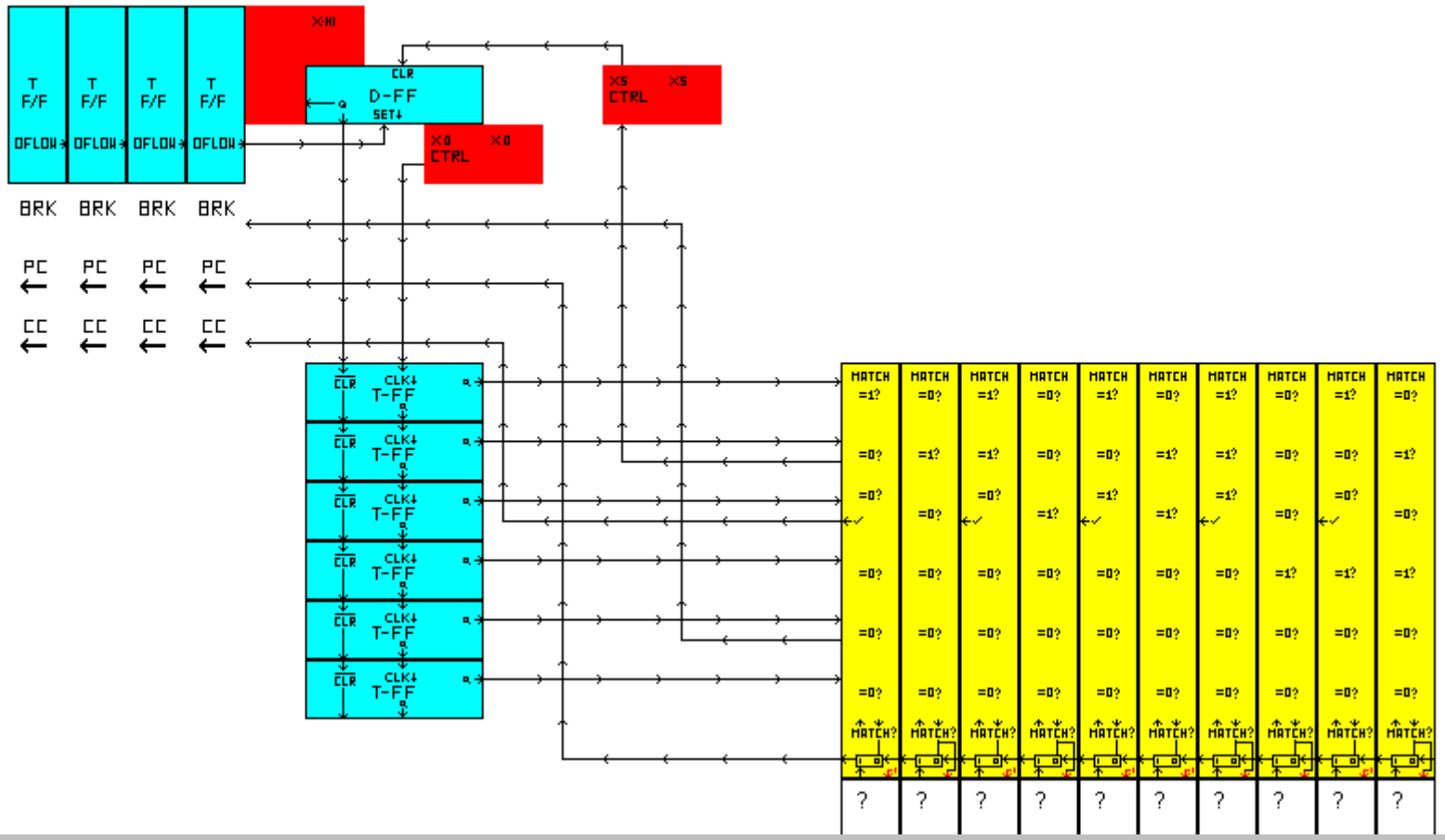
Circuit Synthesis



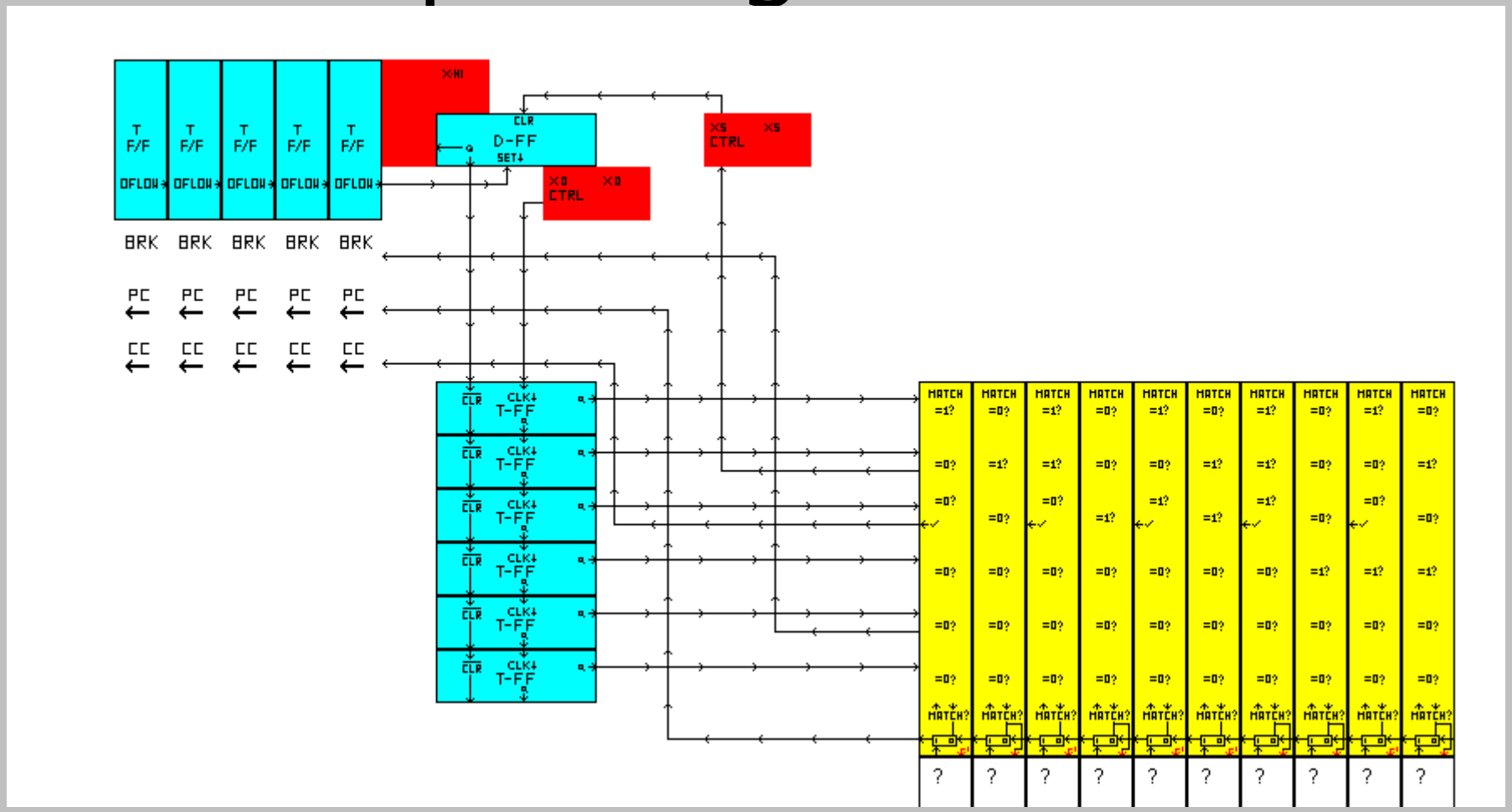
Circuit Synthesis



Expanding Counter



Expanding Counter



Expanding Counter

This is an autonomous, self-modifying hardware system!

Rapid Configuration

- Can configure in parallel
- FPGA-More gates means longer configuration time
- Cell Matrix-More cells means more configuration controllers

Fault Tolerance

- Architecture is inherently fault tolerant
- Can do local testing and isolation
- Example: Scandisk

Scalability

- No global address space
- Edge connections only

Leads to Unique Manufacturing Opportunities

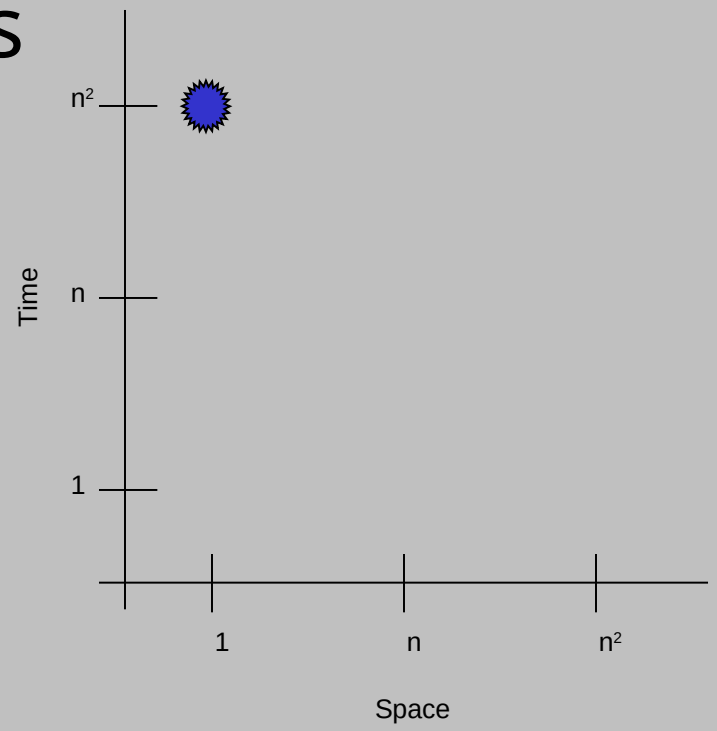
- Wafer Scale
- 3-D Fabrication
- Printable Circuits

Space Vs. Time

- Can we redesign a multiplier from the ground up to take advantage of massive parallelism?
- Interested in an n-bit multiplier, where n much larger than typical word size

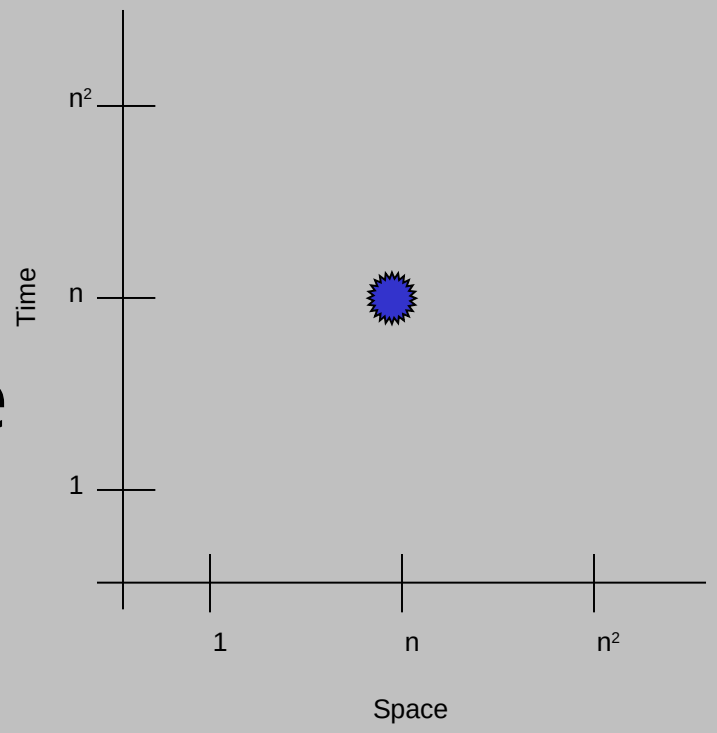
Space Vs. Time

- Fixed hardware uses
  O(1) space and
  O(n^2) time



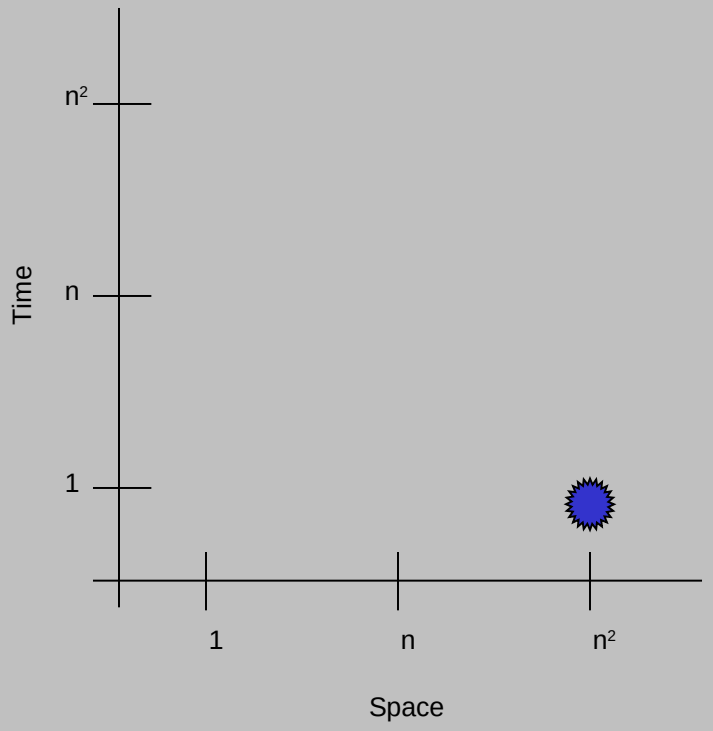
Space Vs. Time

- Reconfigurable HW used to build n-bit accumulators and logic uses O(n) space and O(n) time

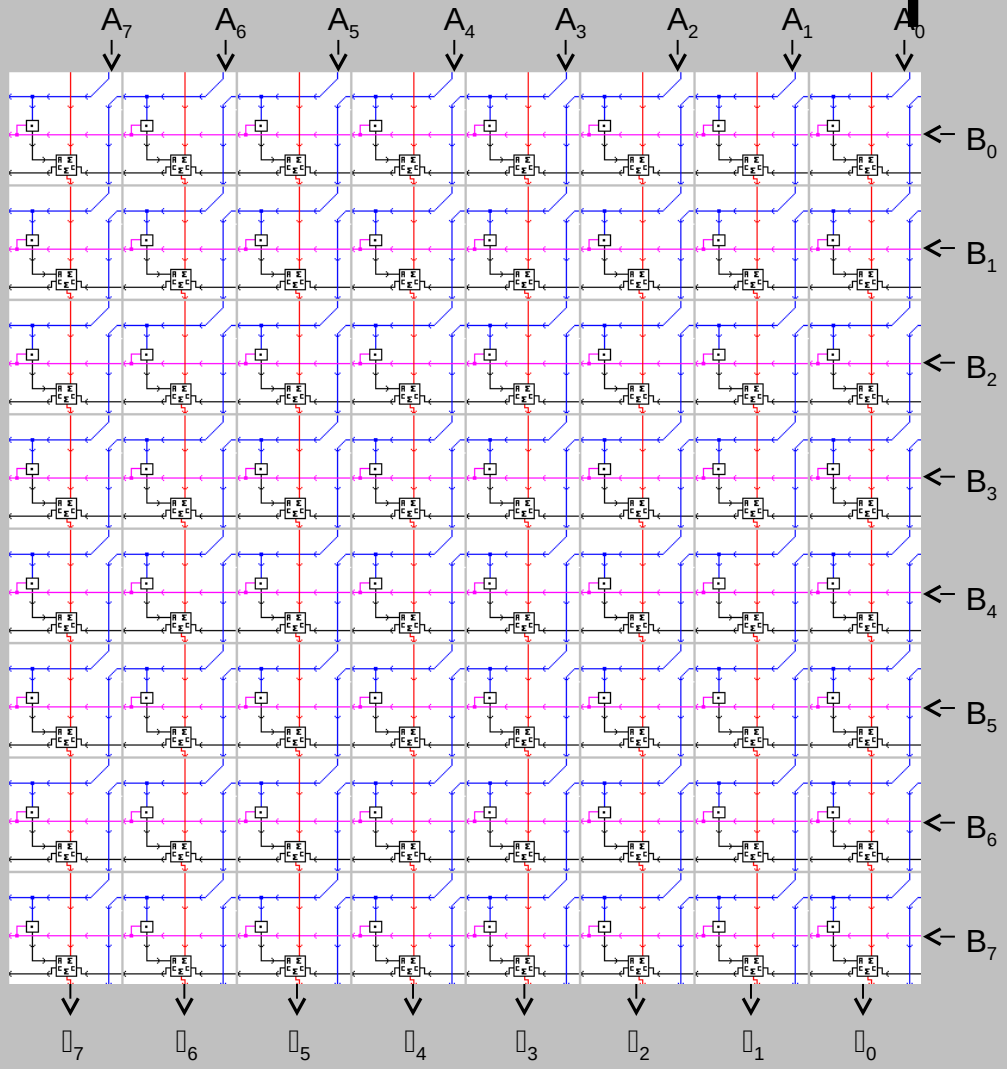


Space Vs. Time

- Reconfigurable HW used to build an nxn multiplication array uses $O(n^2)$ space and $O(1)$ time



Combinatorial Multiplier

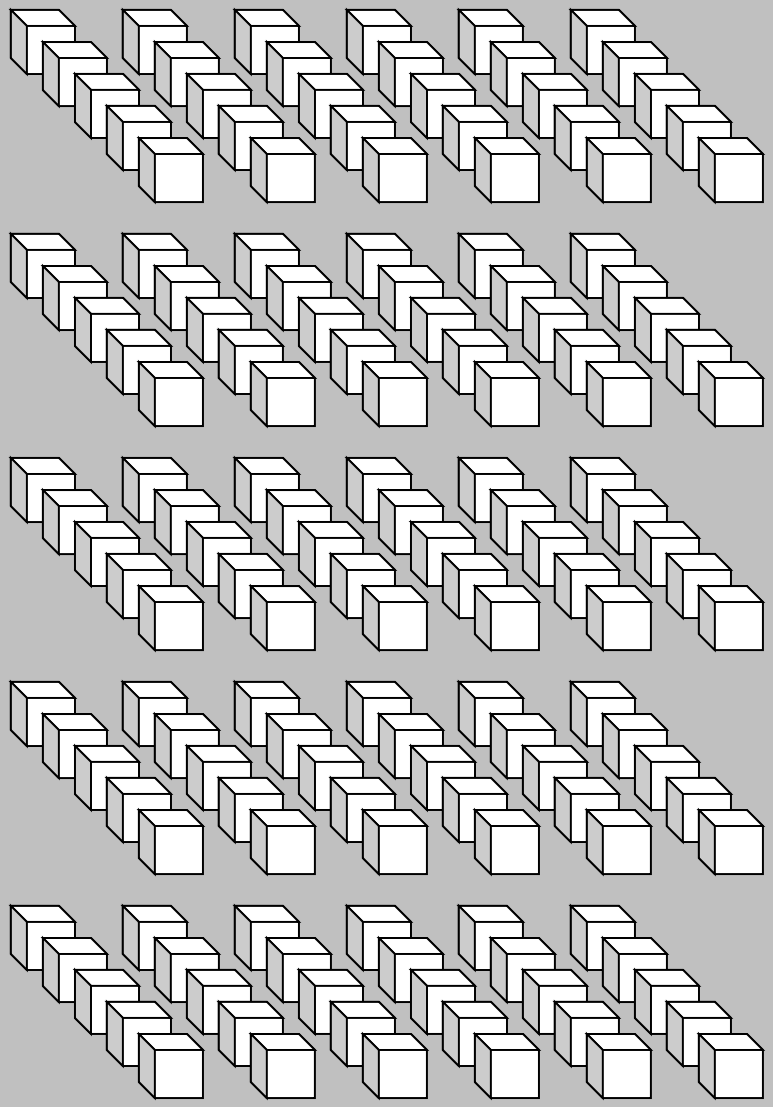


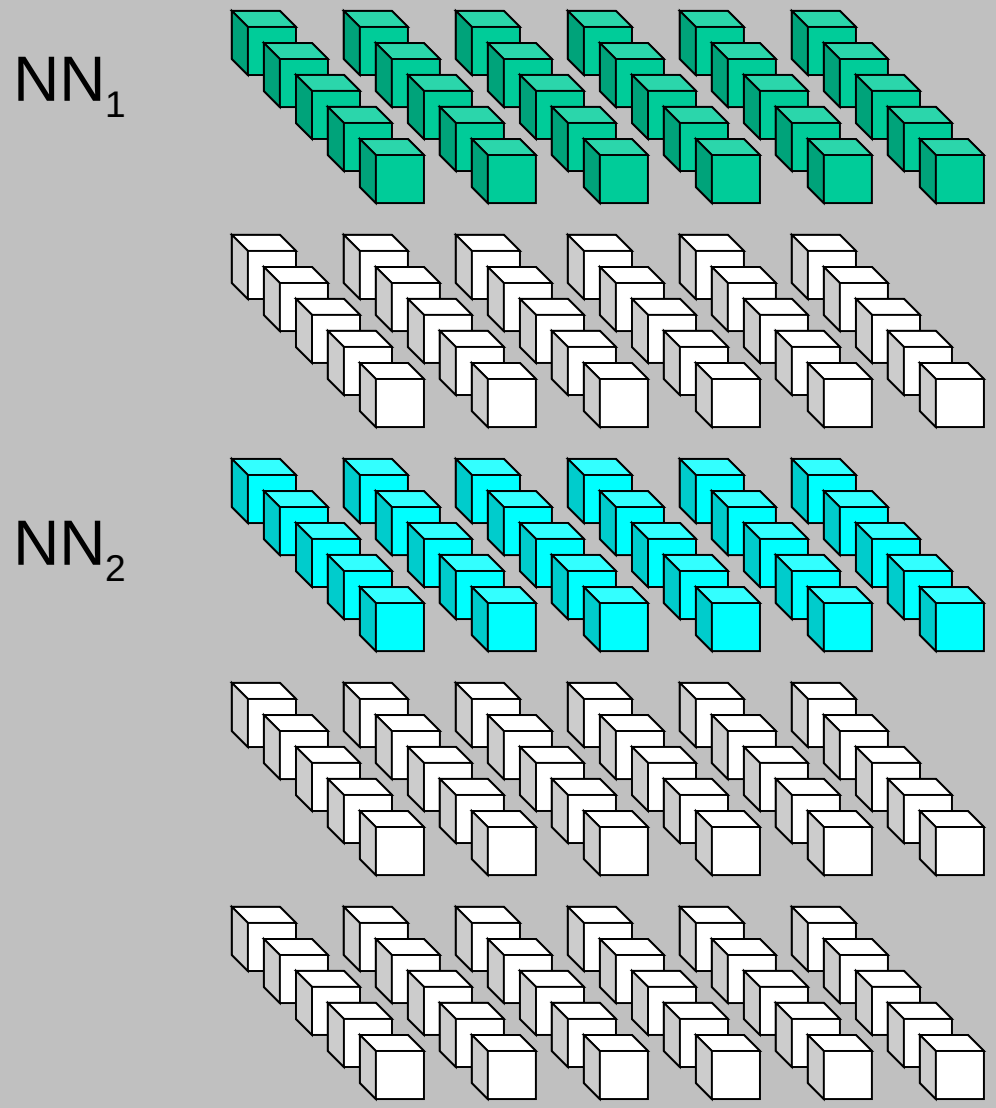
Massively Parallel GA

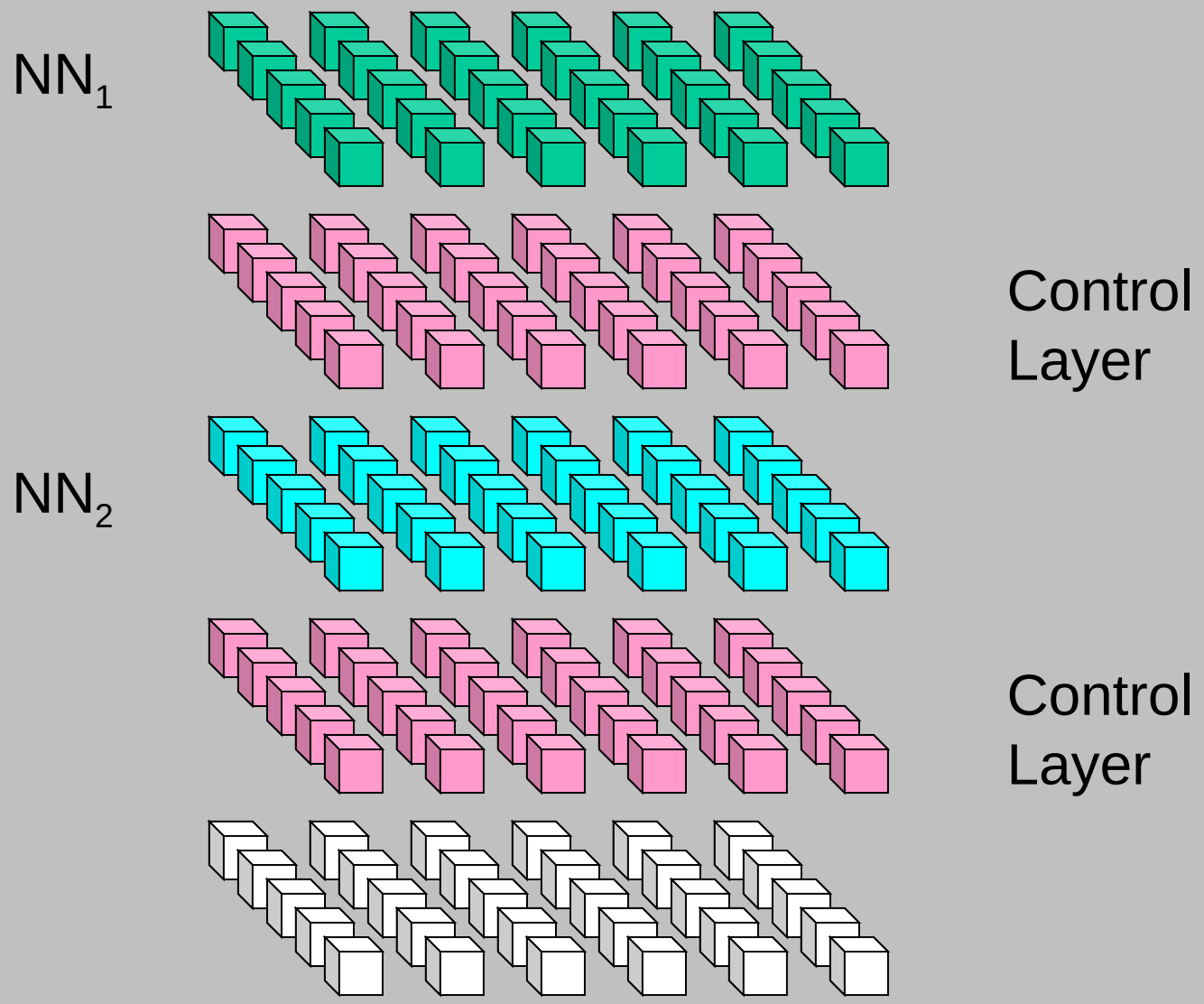
- Evolvable Hardware
- Distribute the GA among the population
- Autonomous evaluation, mating, and production of next generation
- O(1) GA loop

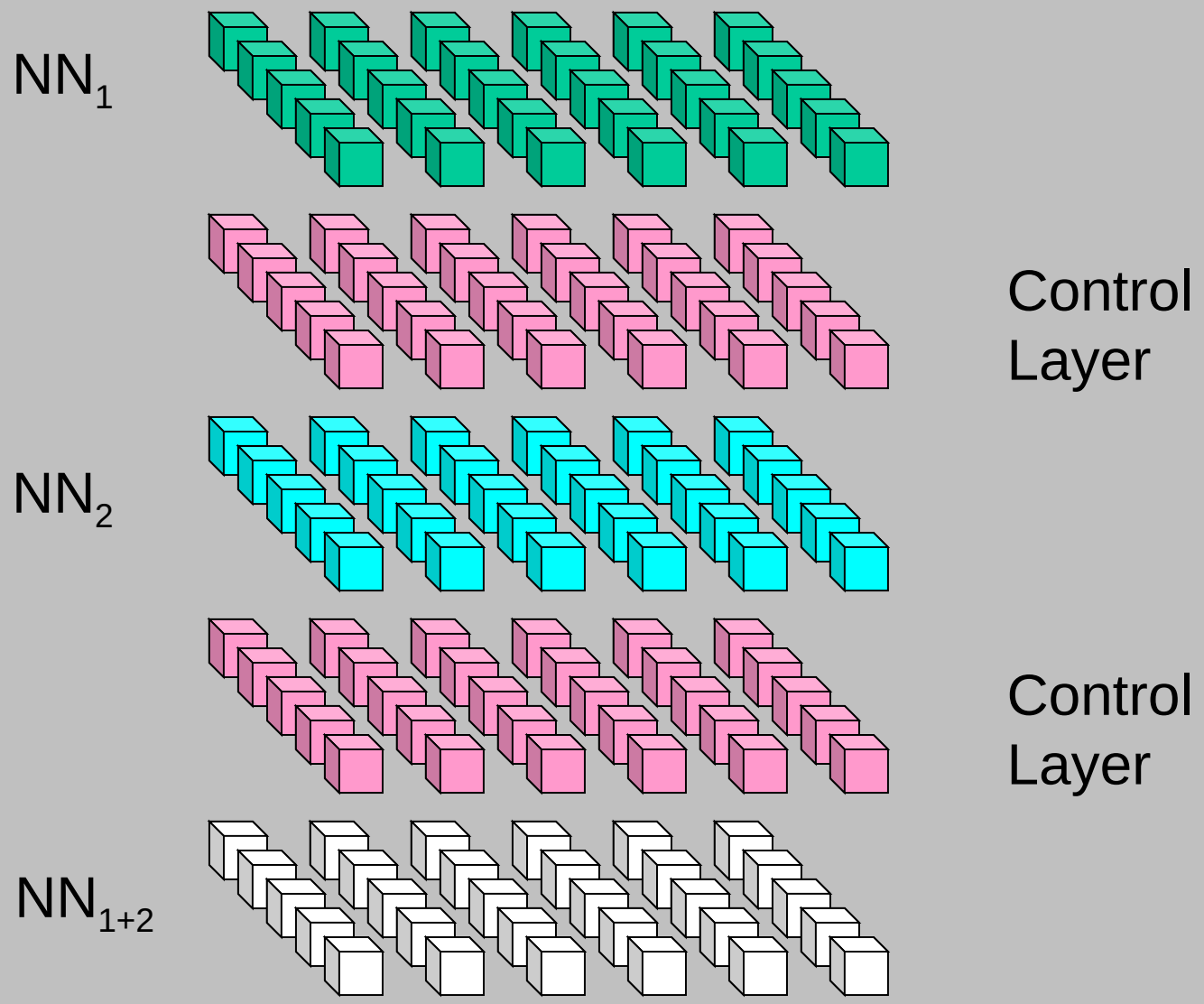
Autonomous NN Module

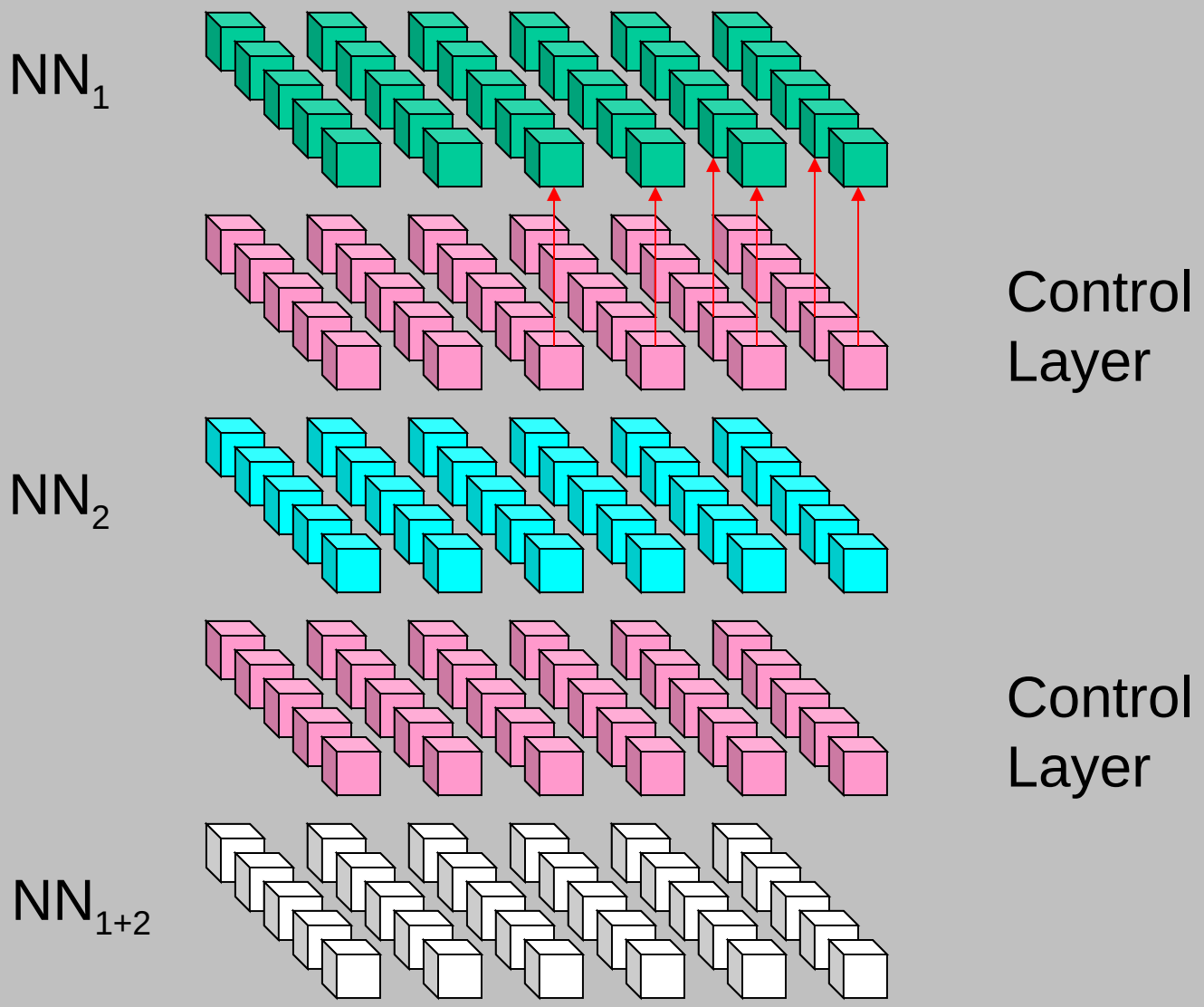
- “Smart” neurons
- Able to grow axons
- Intersections:Cross or Join
- Fast-parallel growth
- Fault Resistant

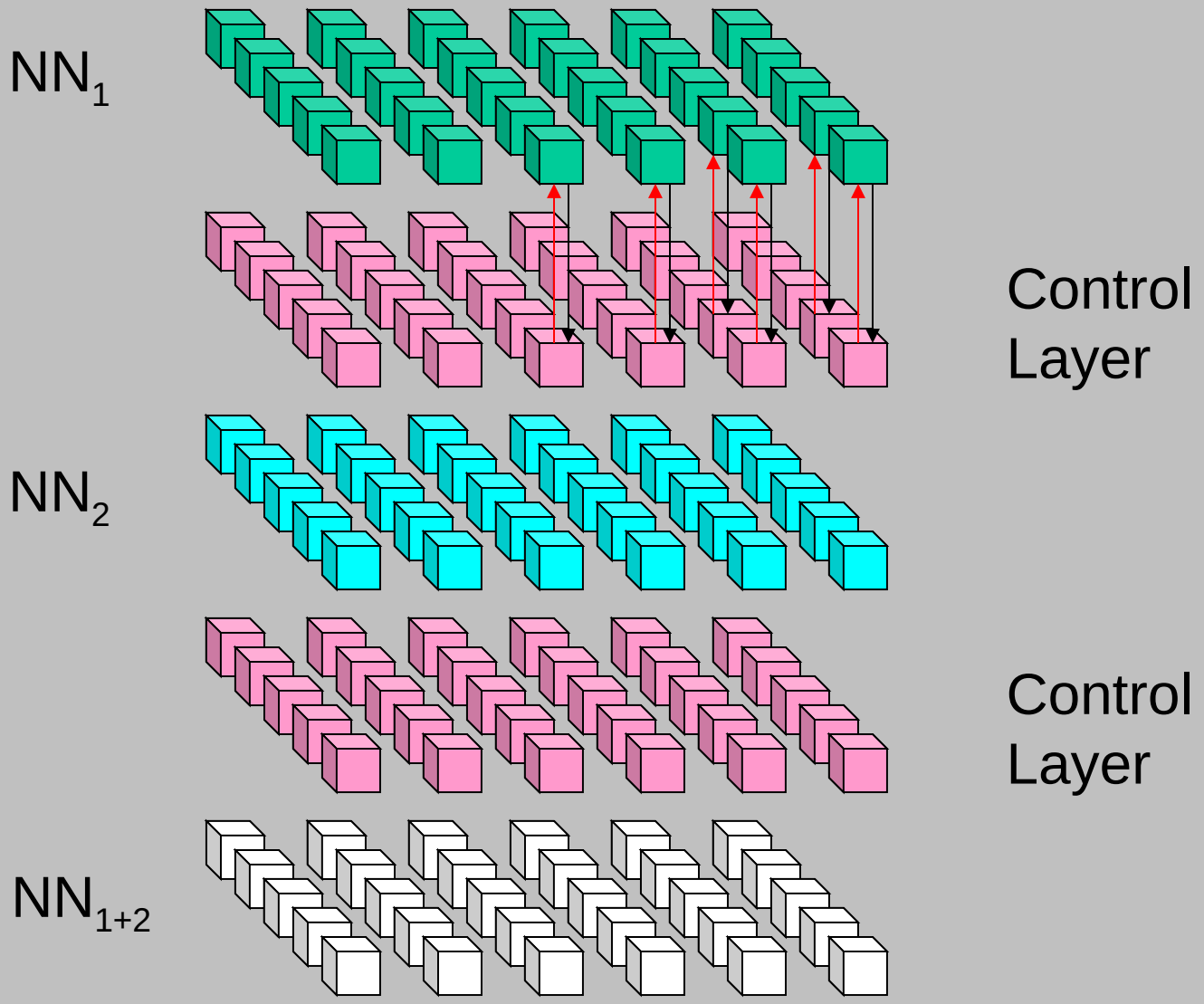


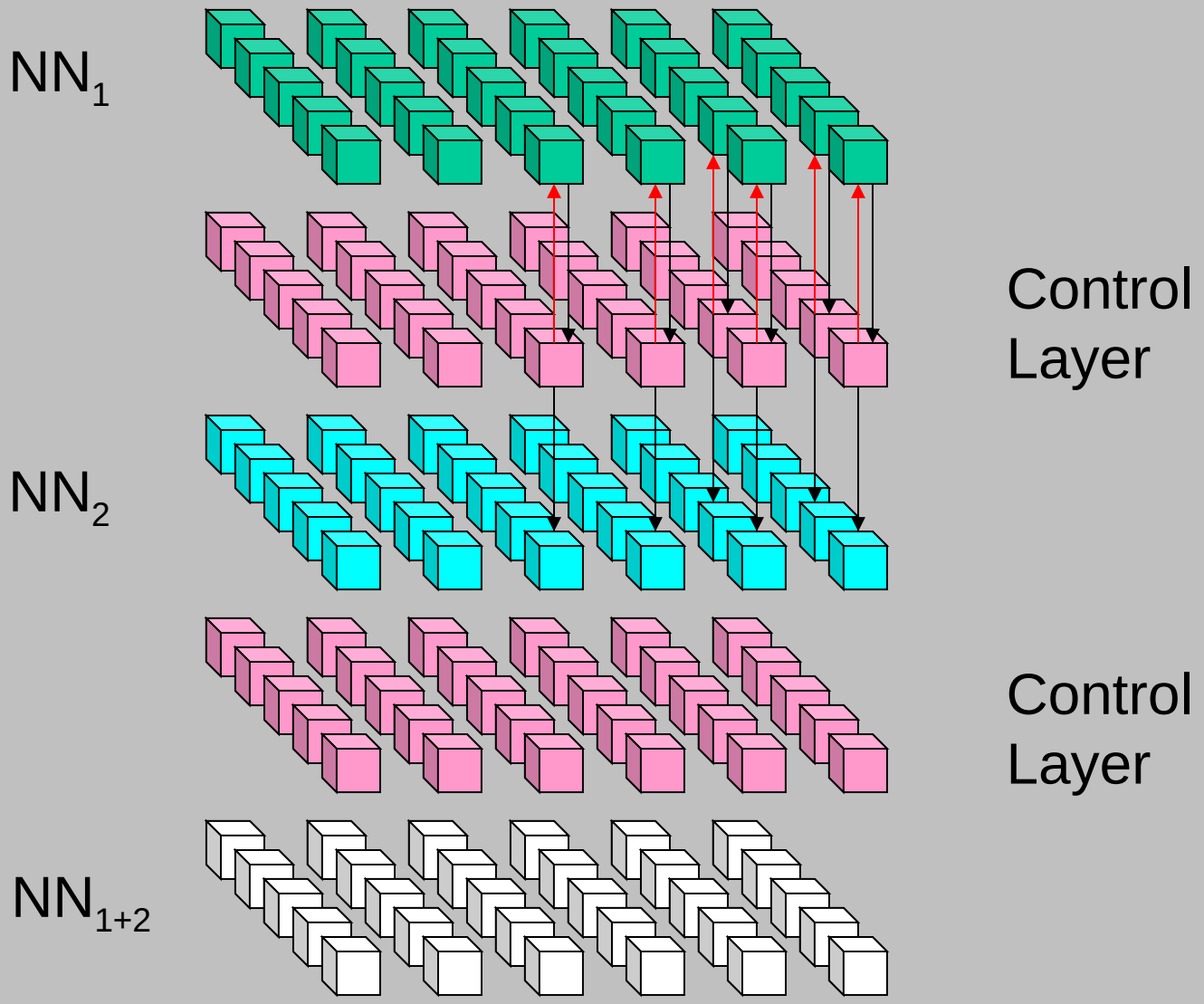

NN1
NN2

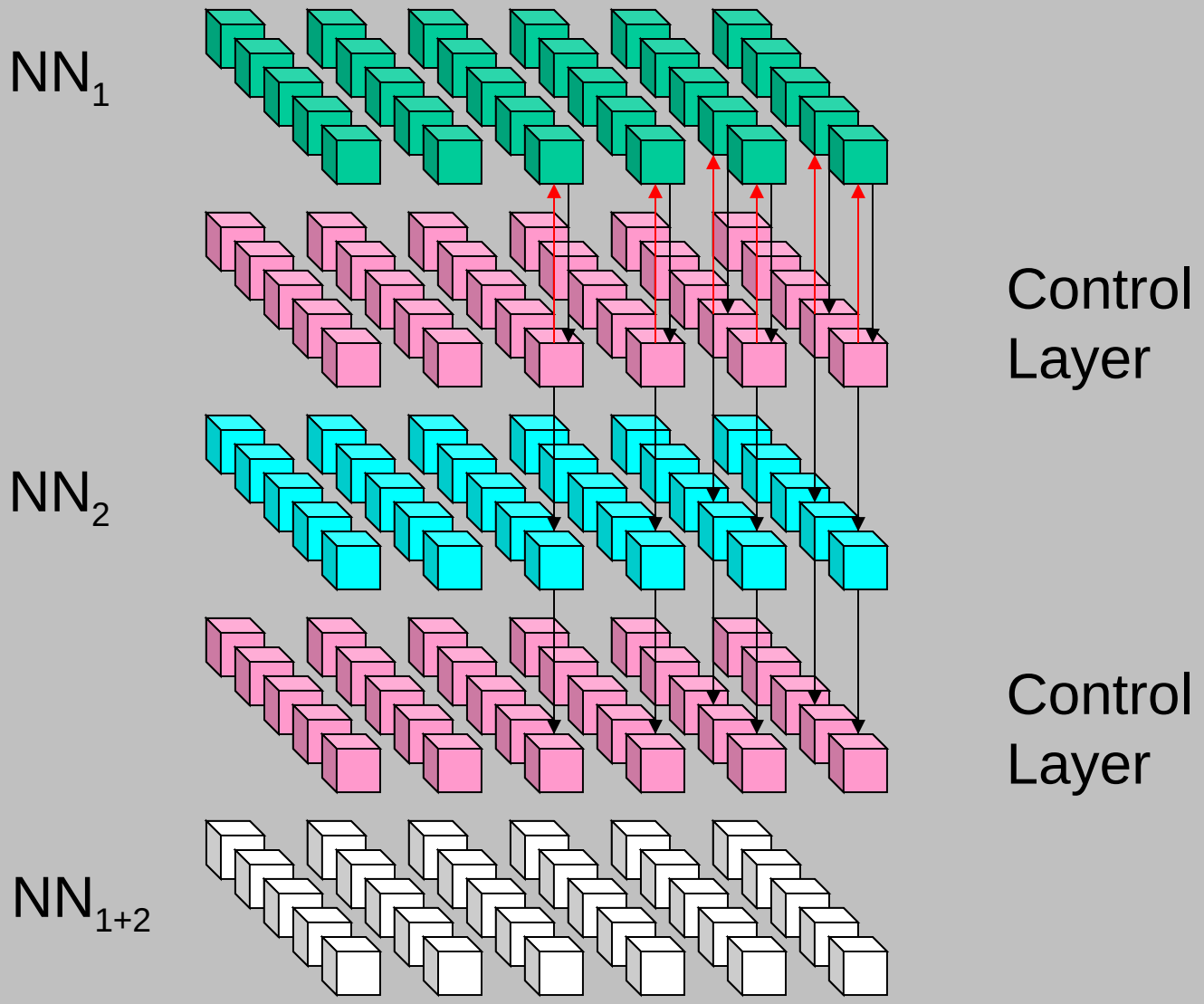

NN_1
Control Layer
NN_2
Control Layer

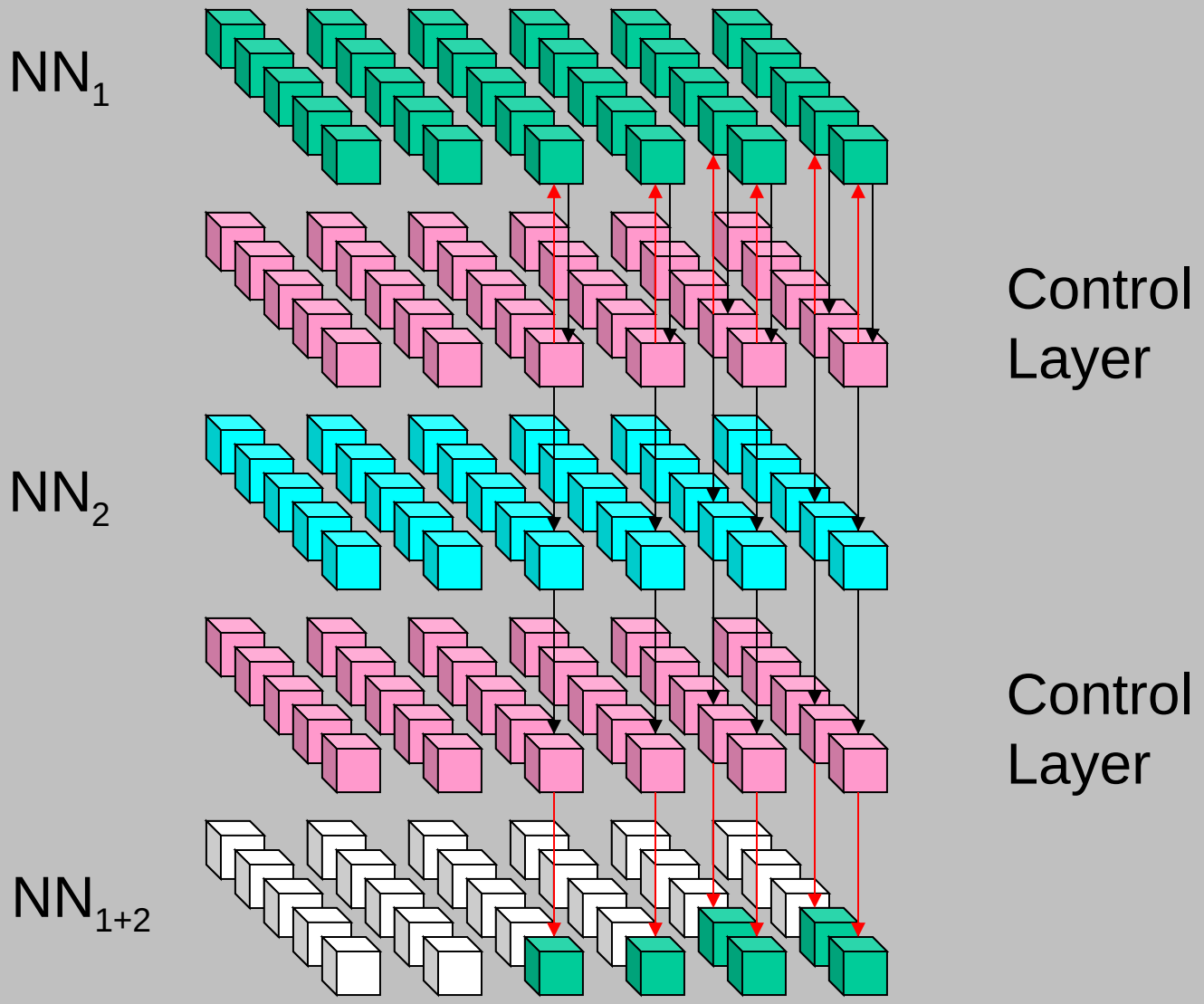

NN_1
Control Layer
NN_2
Control Layer
NN_{1+2}

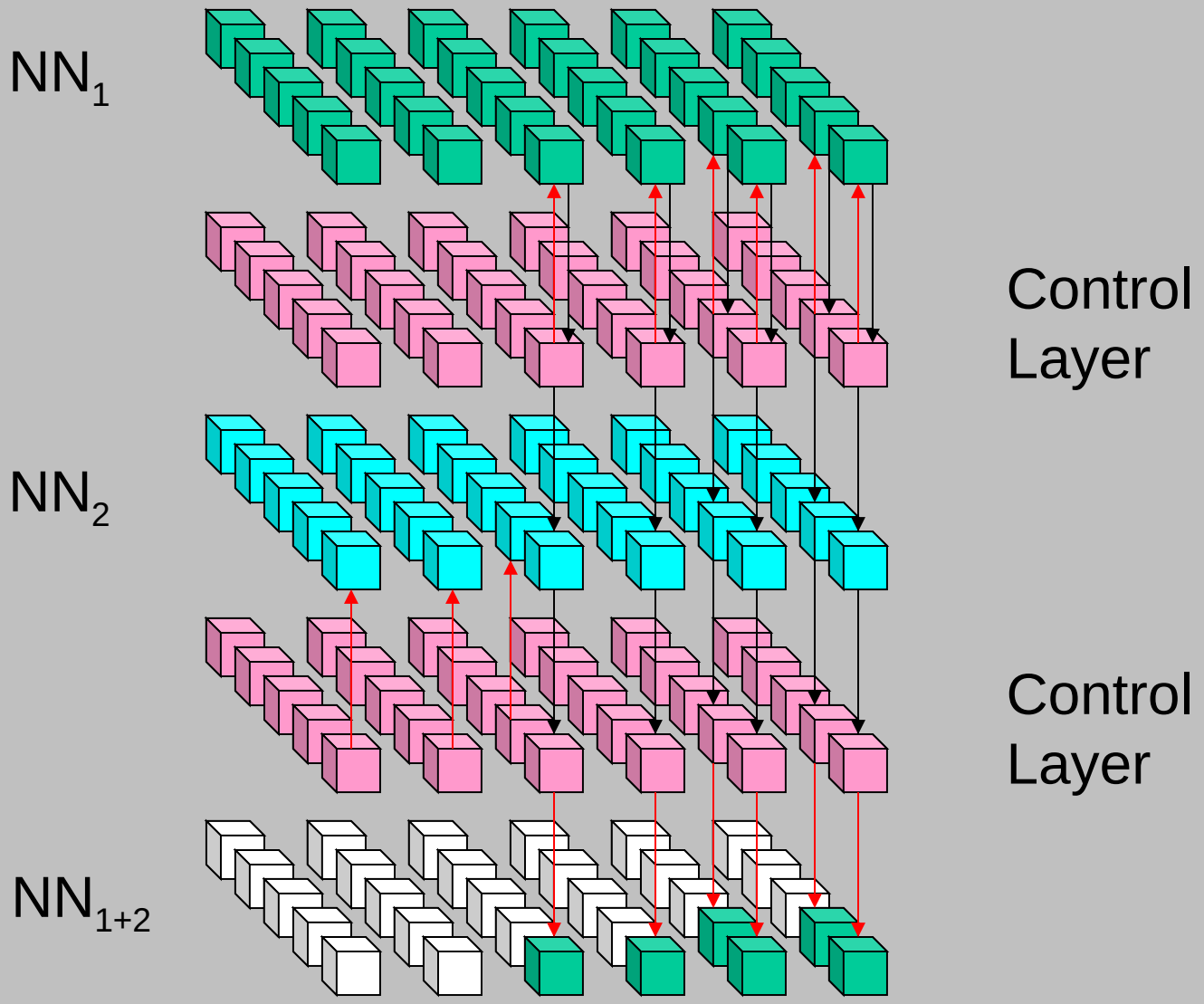

NN_1
NN_2
NN_{1+2}
Control Layer
Control Layer

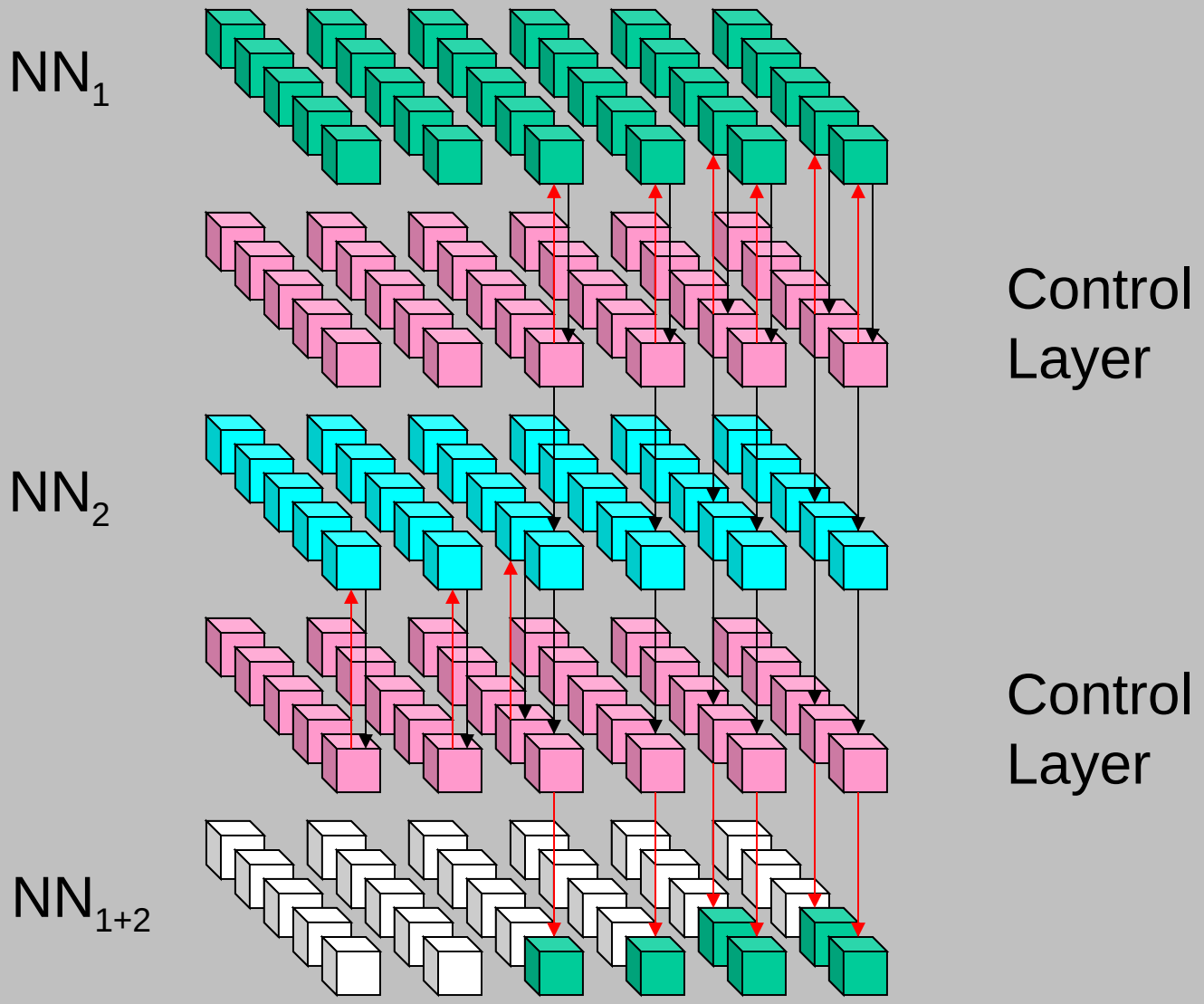

NN1
NN2
NN1+2
Control Layer
Control Layer

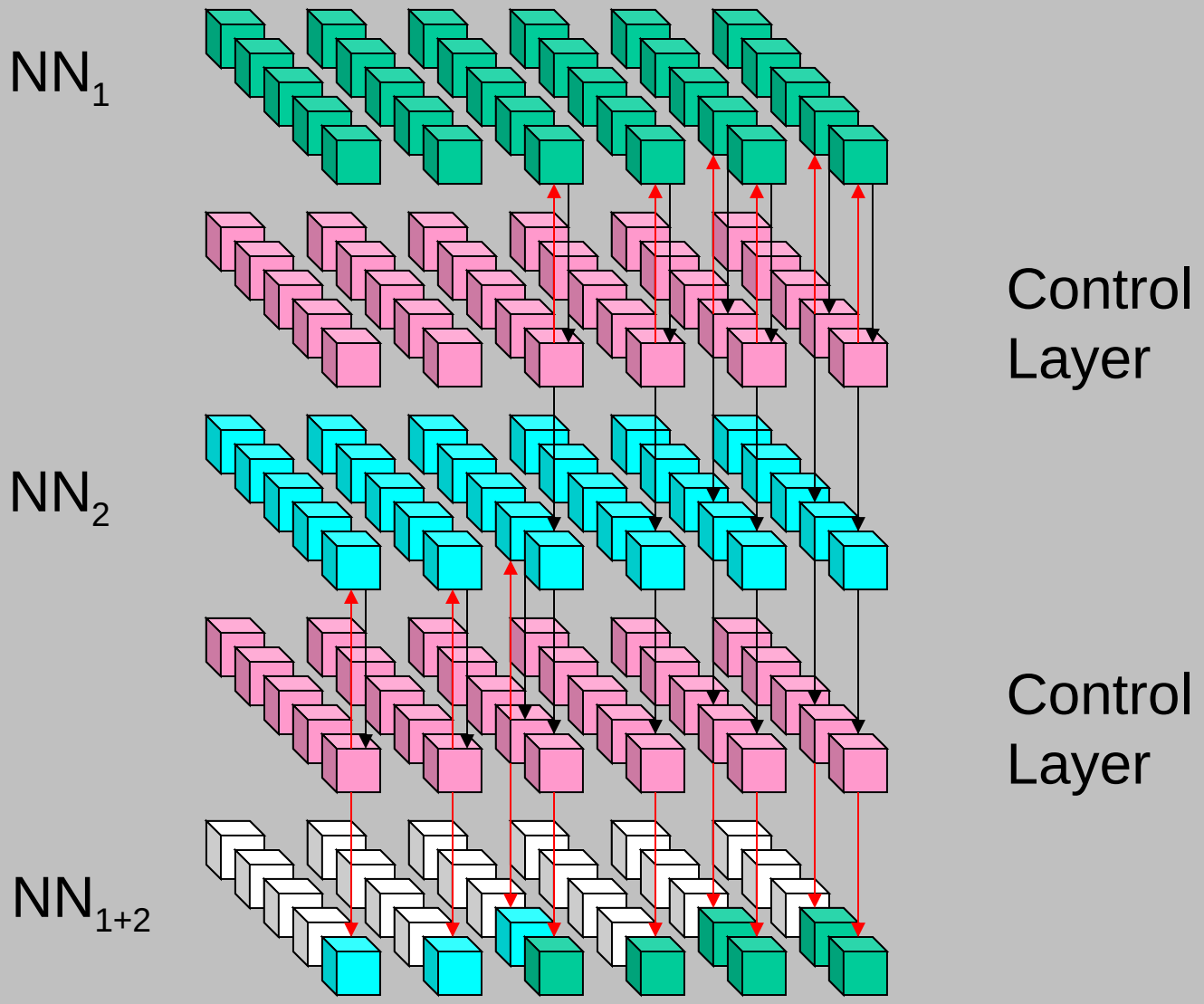

NN1
NN2
NN1+2
Control Layer
Control Layer


NN1
NN2
NN1+2
Control Layer
Control Layer


NN_1
NN_2
NN_{1+2}
Control Layer
Control Layer


NN_1
NN_2
NN_{1+2}
Control Layer
Control Layer

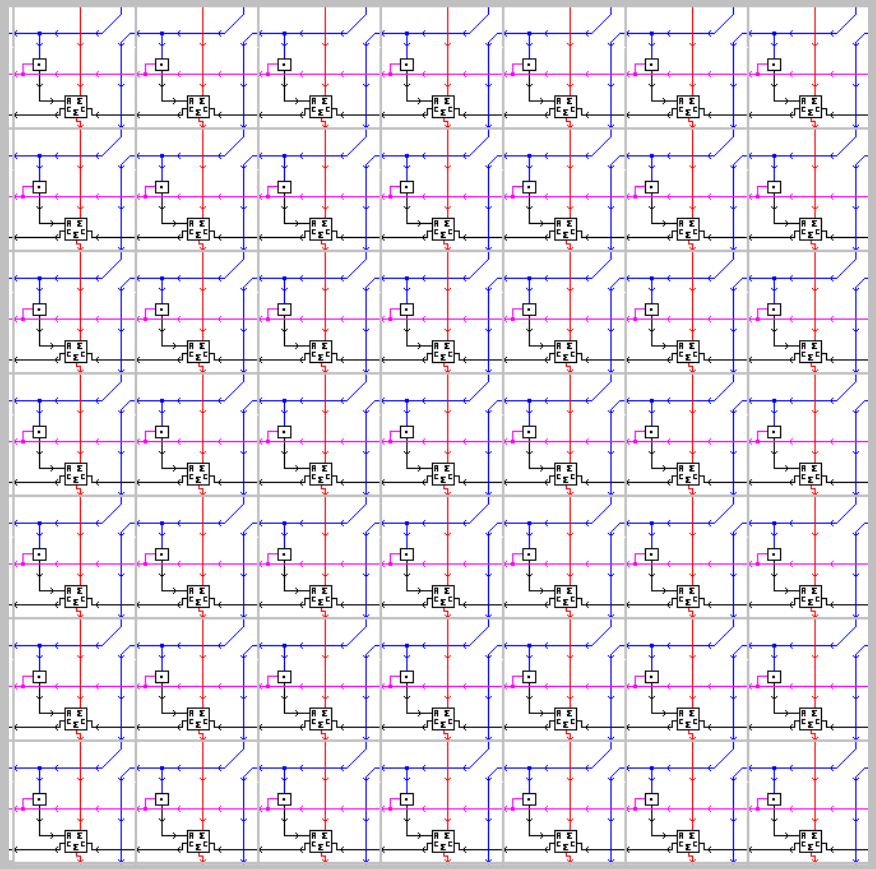

NN_1
NN_2
NN_{1+2}
Control Layer
Control Layer


NN_1
NN_2
NN_{1+2}
Control Layer
Control Layer

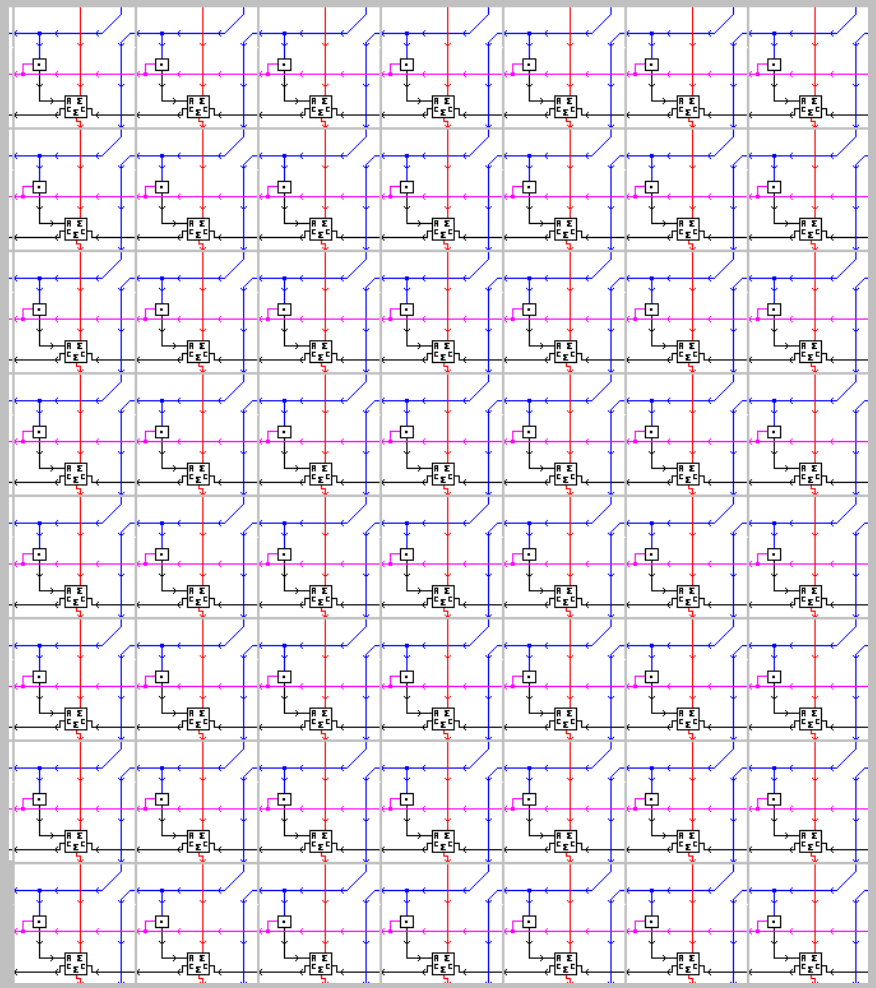
On-the-fly Compilation

- Compile often-used SW loops into HW
- Adapt HW to specifics of problem and history
  - Example: Variable bitsize arithmetic

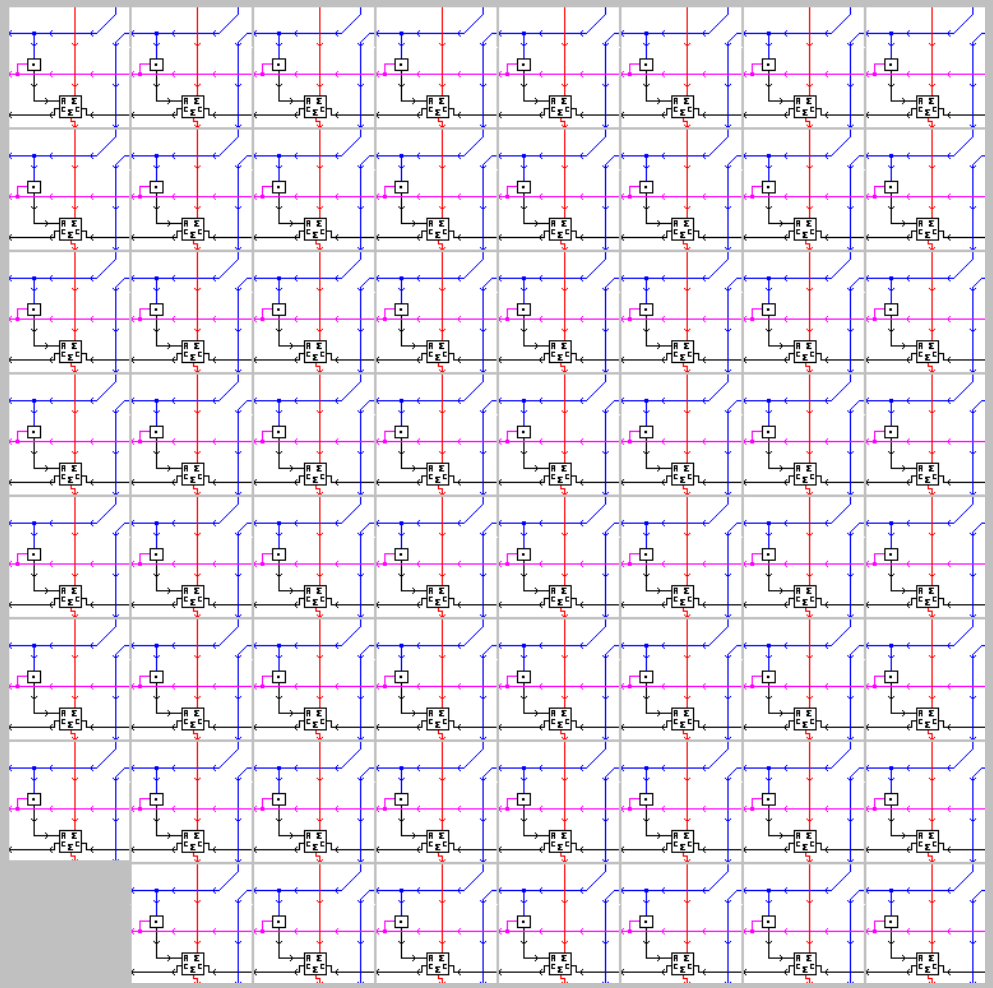
Combinatorial Multiplier



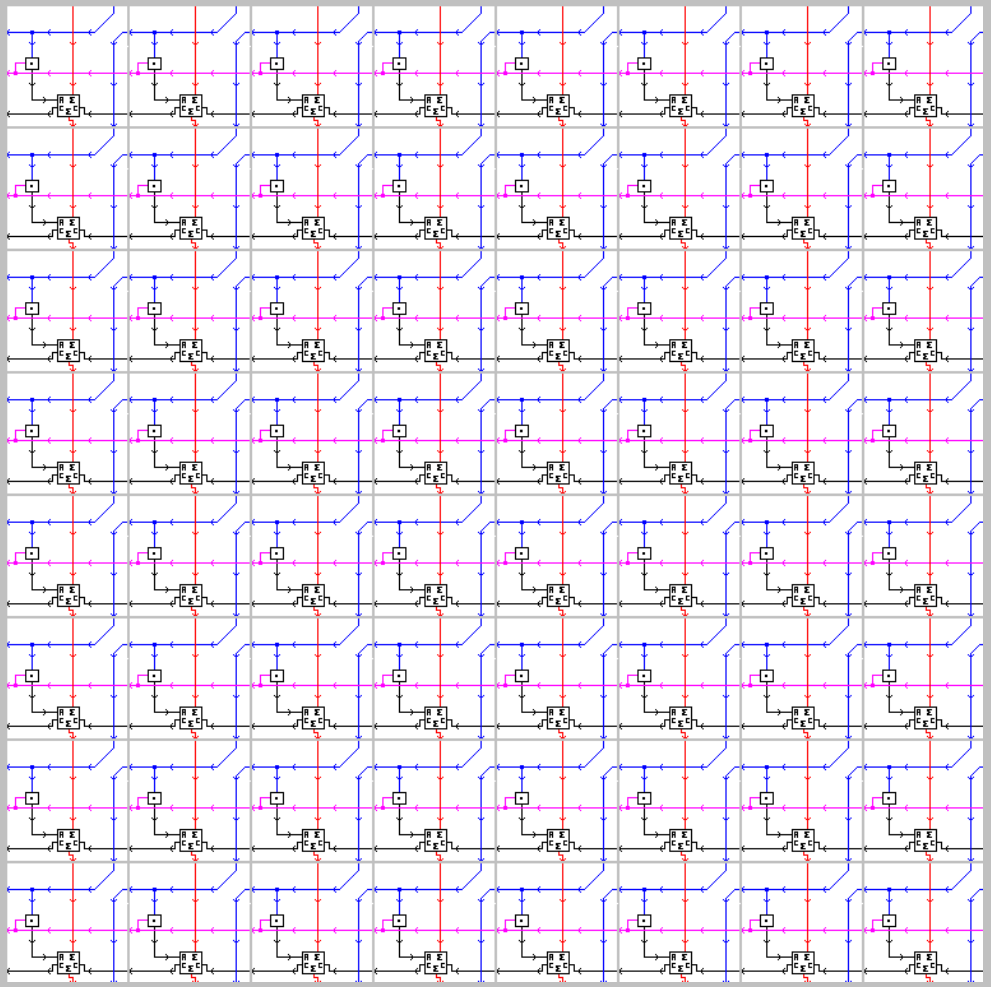
Combinatorial Multiplier



Combinatorial Multiplier



Combinatorial Multiplier



Configuration Speed

- Other devices: configure n blocks in O(n) time
- 2-D Cell Matrix:configure n blocks in $O(n^{1/2})$ time
- 3-D Cell Matrix:configure n blocks in $O(n^{1/3})$ time

Configuration Speed

- Configuring 10^{19} blocks
- Assume 1 uSec per block

Configuration Speed

- Configuring 10^{19} blocks
- Assume 1 uSec per block
- FPGA: 10^{19} uSec

Configuration Speed

- Configuring 10^{19} blocks
- Assume 1 uSec per block
- FPGA: 10^{19} uSec

  (317,097 years, 11 months, 1 day)

Configuration Speed

- Configuring 10^{19} blocks
- Assume 1 uSec per block
- FPGA: 10^{19} uSec

  (317,097 years, 11 months, 1 day)
- 3-D Cell Matrix: 2,154,435 Steps

Configuration Speed

- Configuring 10^{19} blocks
- Assume 1 uSec per block
- FPGA: 10^{19} uSec
  (317,097 years, 11 months, 1 day)
- 3-D Cell Matrix: 2,154,435 Steps
- Each step takes 1 uSEC

Configuration Speed

- Configuring 10^{19} blocks
- Assume 1 uSec per block
- FPGA: 10^{19} uSec
  (317,097 years, 11 months, 1 day)
- 3-D Cell Matrix: 2,154,435 uSec
  (2.15 Sec)

Massively-Parallel Problem Solving

- Once matrix is configured, will execute very quickly
- Search
- Simulation

Other Applications

- Electronic Embryology
- Finite Elements
- DES
- Fault Tolerant Circuits
- Matrix Arithmetic

Summary

- Massively parallel reconfigurable system
- New type of reconfiguration strategy (self-configurability)
- Rapid configuration
- Unique features, unique manufacturing potential
- Numerous applications, old and new

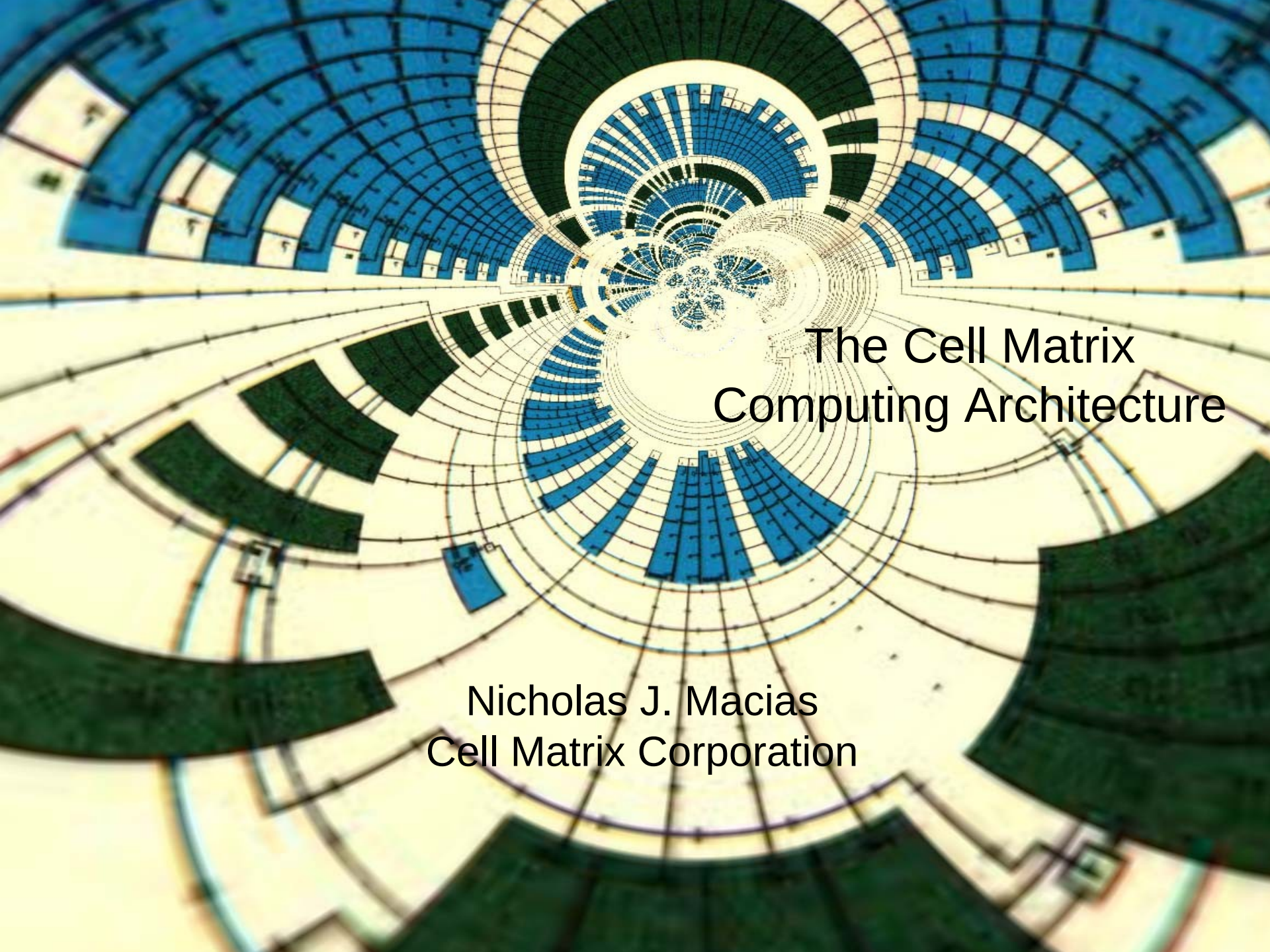
References/Contact

- http://cellmatrix.com
  - Publications, articles, simulators
- US Patents #5,886,531 and #6,222,381-Others Pending
- nmacias@cellmatrix.com
- (801) 414-8900

  (877) 473-0882

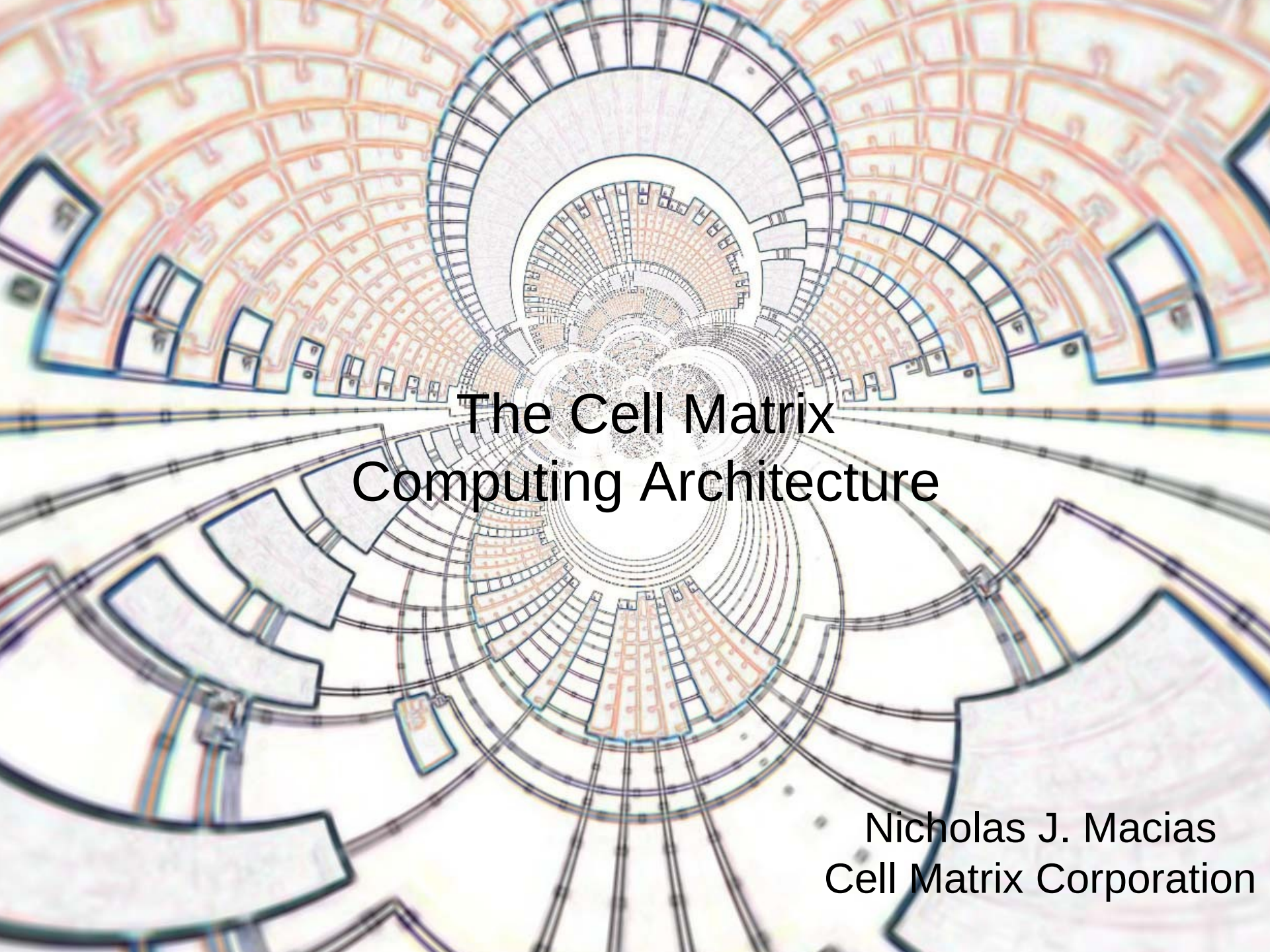
The Cell Matrix Computing Architecture

Nick Macias
Cell Matrix Corporation



The Cell Matrix Computing Architecture

Nicholas J. Macias
Cell Matrix Corporation



The Cell Matrix Computing Architecture

Nicholas J. Macias
Cell Matrix Corporation