A Useful Computing Architecture Composed of 1 Replicated Structure

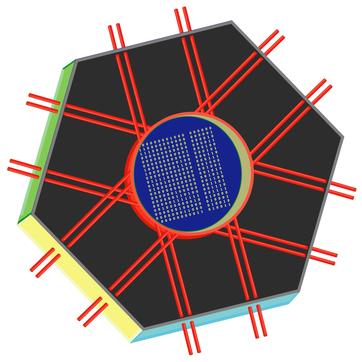
STRUCTURE

The Replicated Structure

Each cell is a junction point where a small amount of progress within a larger process is made. Cells compute and exchange information with neighboring cells.

The cell body is represented to the right by the hexagon. The red pipes are wires connecting the cell to its 6 neighbors, and the wires interact with each other in the center cylinder, where the interaction is precisely specified by a lookup table.

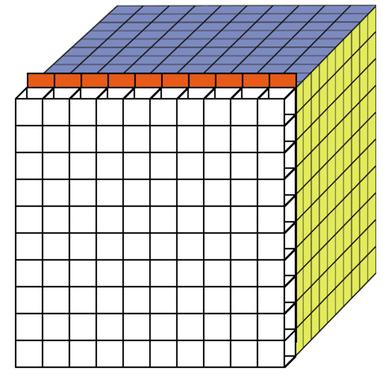
The lookup table contents can be changed by a neighboring cell. We take advantage of this distributed control to perform many system configuration operations in parallel. The final circuit can also be designed to be self-modifying and massively parallel.

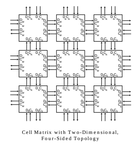


The Structure is Repeated to Form a Matrix

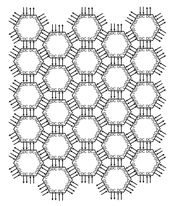
Cell and Matrix Structure is Flexible

A Matrix can occupy a space that is 2 or 3-dimensional





Cell Matrix with Two-Dimensional, Four-Sided Topology



Cell Matrix with Two-Dimensional, Six-Sided Topology

A Cell can be defined to have n Sides (3, 4, 6, 8, ...) and can be planar, curved, or 3-dimensional.

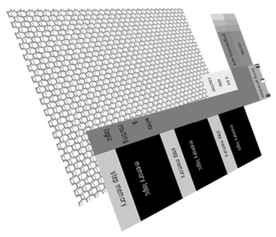
Matrix size is arbitrary: 4 cells to 4 trillion cells.

The topology of inter-cell connections can also be defined a number of ways.

The Matrix can then be Configured to Implement Any Digital Circuit

FUNCTION

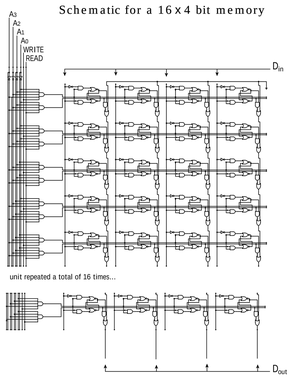
Standard Digital Circuits can be Layered on Top of a Cell Matrix



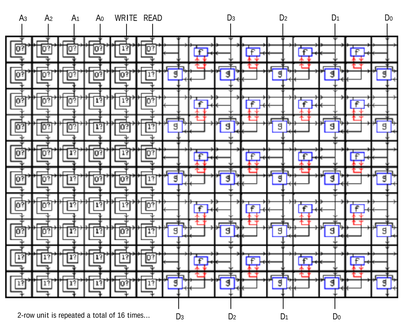
An Example:

Computer Memory is Implemented on a Cell Matrix via a Straightforward Translation of a Schematic Diagram.

Schematic for a 16x4 bit memory



16x4 bit memory implemented on a set of Cell Matrix cells



Parts

64	2 input ORs
32	5 input ANDs
192	2 input ANDs
128	2 input NORs
68	inverters
1280	wire segments

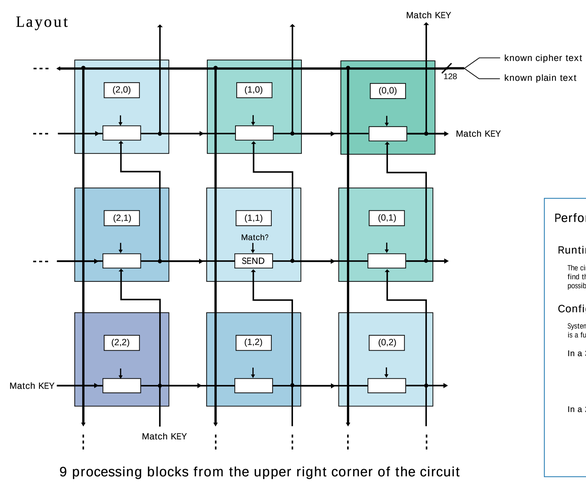
Parts

480	cell matrix cells

The Architecture Also Scales Up to Trillions of Cells

Example of a Huge, Fast Circuit

A 56-bit DES Encryption Cracker Utilizing A Spatially Distributed Parallel Algorithm



9 processing blocks from the upper right corner of the circuit

Size

56 bit keys = 2^{56} possible keys

Representing all keys at once takes about 10^{17} processors. In a 3D configuration, the dimensions are just under 10^{6} in each direction.

Each processor takes about 4 million cells, for a total of about 10^{23} cells.

Performance

Runtime = O(1) cycles

The circuit takes a constant amount of time to find the key, regardless of the number of possible keys.

Configuration Time

System Build/Setup time, or Configuration Time, is a function of the number of processors.

In a 3D matrix, system Configuration time is on order of

$\sqrt[3]{\text{number of processors}}$

In a 2D matrix, system Configuration time is on order of

$\sqrt{\text{number of processors}}$

We have a small, 4-bit version of this circuit that demonstrates this algorithm and circuit. Ask us about it.

Efficient Local, Parallel Control of Trillion-Cell Systems: How?

- Each cell is a controller. Thus, control structure scales with matrix size: the more cells a system contains, the more controllers it contains.
- Also, the relatively high likelihood of faults in a trillion-cell system is manageable because the architecture is inherently fault tolerant. It also provides efficient local detection and control of damaged hardware.
- In addition, setup of such a system does not require months or years, because the distributed, local control permits efficient, parallel system customization or configuration.

Greatly Expanded Computing Capabilities, Including:

Quickly Searching Large Spaces
Simulation of Phenomena
Fault Tolerant Systems
Image Processing
Data Processing
Large Matrix Operations
Wide Bit Arithmetic

Evolvable Hardware
Neural Network Implementation
Adaptive Hardware
Self-Optimizing Circuits and Systems
Self-Organizing Circuits and Systems
Spatially Distributed Algorithms
"1 Problem, 1 Machine" Customization