Hybrid Circuits with Area Distributed Interface
Hybrid CMOS/Memristor Circuits

historic (first?) version:  
current version:

**WHAT:**  
• CMOS stack + simple nano add-on  
• nanowire crossbar + two-terminal devices (latching switches)

**WHY:**  
• CMOS functionality and infrastructure intact  
• inexpensive fabrication of reproducible nanodevices  
• advanced lithography with no need in layer alignment
Array Architecture
example of reading the memory cell state

access device, e.g. transistor
memory element, e.g. capacitor, variable capacitor, magnetic tunnel junction, floating gate transistor, etc.

data inputs/outputs

control inputs
common node (ground)
Array Architecture
alternative representation

2 N lines
N^2 devices
Crossbar Architecture

- Top (nano)wire level
- Bottom (nano)wire level
- Similar two-terminal devices at each crosspoint
Crossbar Architecture
reading/writing the memory cell state

USE MEMRISTOR IN DIGITAL REGIME

Read

Write

\[ V = V_r / 2 \]

\[ V = V_w / 2 \]
Access device functionality is integrated in cross-point device.
CMOL Circuits
area-distributed interface - vias

Strukov & Likharev, Nanotechnol. 16 137 (2005)
CMOL Circuits: CMOS circuitry

Strukov & Likharev, Nanotechnol. 16 137 (2005)
CMOL Circuits
accessing pair of vias independently

Strukov & Likharev, Nanotechnol. 16 137 (2005)
CMOL Circuits
full structure: CMOS + Xbar

Strukov & Likharev, Nanotechnol. 16 137 (2005)
CMOL Circuits: nanowire fabric

\[ \sin \alpha = \frac{F_{\text{nano}}}{\beta F_{\text{CMOS}}} \]
\[ \cos \alpha = r \frac{F_{\text{nano}}}{\beta F_{\text{CMOS}}} \]
where \( r \) is integer
CMOL Circuits
example of reading/writing a xpoint device

Unique access to any of xpoint devices

Strukov & Likharev, Nanotechnol. 16 137 (2005)
3D CMOL Circuits
virtual xbar array with devices from two layers

- $N^2/\beta^2$ maximum number of layers
- constant via density
- minimum number of masks (only one set for all layers)
- any flexibility in CMOS cell (but uniform pattern of vias)
- footprint: $4F^2_{\text{WIRE}}/M$

Strukov & Williams, PNAS (2010)
Manufacturable Layout
Digital Memories
Digital memories: Main Idea

* Memory element = crosspoint memristive device in the crossbar (no 1T)

* Digital regime for memristive device

* Sensing, driving, error correction in CMOS

* Area-distributed interface helps absorbing CMOS overhead

* Both lateral scaling (down to ~3 nm) and z-scaling
Memory Architecture

Top level architecture

Block architecture

Strukov & Likharev, Nanotechnol. 16 137 (2005)
CMOL Memory Simulation

Bottom line:
- density up to 1 Tb/cm² feasible
- speed, power OK
- defect tolerance acceptable (~10%)
- up to 1Pb/cm² for 3D CMOL

Xbar Memory Demo


Samsung (lithography/NiO) ~0.01 Tb/cm²

Caltech (imprint/mol.) ~0.1 Tb/cm²


Baek et al. IEDM (2005)

Julich Research Center (imprint/TiO₂) ~0.1 Tb/cm²

FPGAs Circuits
ASIC vs. FPGA vs. μP

- Manufacturing cost
- Performance
CMOL FPGA: Main Idea

**typical FPGA ...**

- metallization
- config. bits
- & logic

**... and with lifted config. bits**

- metallization & config. bits
- logic

~ 90% is interconnect (memory bits + three state buffers/pass gates)

Crosspoint memristive device somewhere in the layer above CMOS

Density, speed, power improvement → configurable ASIC
CMOL FPGA vs. FPNI: Logic Architecture

- All logic functions in CMOS, nano only for routing
- Less dense (factor 10) but better power consumption & speed and simpler nanodevices

Snider and Williams, Nanotechnology 18 035204 (2007)
Main Idea: CMOL FPGA Structure

- **Tile architecture**
  - Tile = 1 simple latch + 12 basic cells
  - Breaks in both nanowire layers

- **Basic cell**
  - CMOS select line
  - CMOS data line
  - Input nanowire
  - Output nanowire

- **Simple latch cell**
CMOL FPGA: Logic Architecture

Large fan-in gates are possible, e.g.,

Not shown nanodevices are in OFF state

Nanodevices do not change state!

Strukov & Likharev, Nanotechnology, 16 888 (2005)
Routing Architecture  local connections

- Input cell domain
- Output cell domain

- Shape and size of domain are roughly square and the same for all cells!
- # cells in the domain $\approx (\beta F_{\text{CMOS}} / F_{\text{nano}})^2 \sim 1600$ (for realistic parameters)
- Large fan-out ($\sim 50$) without delay degradation for minimum width inverter

Strukov & Likharev, Nanotechnology, 16 888 (2005)
Routing Architecture global connections

Gates located not within each other’s connectivity domain are connected with series of inverters

Strukov & Likharev, Nanotechnology, 16 888 (2005)
Design Automation: General Flow

Input circuit $\downarrow$ blif format

**SIS:** Technology (NOR gate and latch) mapping

- Defective cells
- Circuit processing
- Initial value of $K$

Heuristic placement

- Decrease $K$
- Increase $K$

Global router

- $K = 0$
- $\text{count}_{\text{max}} > T-K$
- $\text{count}_{\text{max}} < T-K - \Delta$
- otherwise

Detail router

- Defective nanodevices
- Exit without success
- Exit with success

$K$ – # cells allocated for placement per tile, e.g.,

- $T=12$
- $K$ (logic) = 5
- $T-K$ (routing) = 7

Flexible resource allocation!
Step 1: Technology Mapping
Step 2: Circuit Processing

- Remove all inverters, instead assign polarity property to each net
Step 3: Placement by Simulated Annealing

- **Tile domain**

- **Cost function (without polarity)**

  \[
  \text{cost} = \sum_{i=1}^{\text{lath}} \left[ \frac{2 \left( \max \left( |x_0 - x_i|, |y_0 - y_i| \right) - 1 \right)}{A - 1} \right]
  \]

- **Cost function example**

Case: \( F_{\text{CMOS}} = 45 \text{ nm}, F_{\text{nano}} = 4.5 \text{ nm}, \beta = 4 \)

\( A \approx \frac{F_{\text{CMOS}}}{F_{\text{nano}}} = 5 \)
Global Routing (Timing Driven)

- Objective is to connect global nets by using reserved (unused) basic cells, i.e. configuring them as inverters, in such way that post-placement delay is not increased.
- Ideally, the router should be able to
  - Find shortest path Steiner trees – NP hard!
  - Avoid congestions, i.e. requesting more than physically available routing inverters in a tile.
- Greedy algorithm with quasi-shortest path Steiner trees
  - Possible improvement: slack analysis.

![Diagram](image-url)
use tiebreak to provide for more even spread of routing inverter!
Example (dsip.blif): Initial Placement

- Global connections
- Local connections

Case: \( A = 10 \)
Example (dsip.blif): Final Placement

- Global connections
- Local connections

Case: \( A = 10 \)
Defective Tolerance: Faulty Cells

- **Faulty cells:**
  - defective interface pins
  - broken/shorted nanowires
  - defective CMOS cells
  - even “stuck-on-close” nanodefects

- **Example:** dsip.blif, $A = 10$

Trivial changes to placement and global routing steps!
Step 4: Global routing

- Objective is to connect global nets by using reserved (unused) basic cells, i.e. configuring them as inverters, in such way that post-placement delay is not increased.
- Ideally, the router should be able to:
  - Find shortest path Steiner trees – NP hard!
  - Avoid congestions, i.e. requesting more than physically available routing inverters in a tile.
- Greedy algorithm with quasi-shortest path Steiner trees.
Single Net Global Routing

not always optimal!
Example (dsip.blif): Global Routing

- Global connections
- Local connections

Case: \( A = 10 \)
## Main Results: Toronto Benchmark Set

<table>
<thead>
<tr>
<th>Circuit</th>
<th>CMOS FPGA $F_{\text{CMOS}} = 45\ \text{nm}$</th>
<th>CMOL FPGA $F_{\text{CMOS}} = 45\ \text{nm}, \ F_{\text{nano}} = 4.5\ \text{nm}, \ \text{max fanin} = 7$</th>
<th>Comparison</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Depth</td>
<td>LUTs</td>
<td>Linear size (tiles)</td>
</tr>
<tr>
<td>alu4</td>
<td>7</td>
<td>1274</td>
<td>19 \times 19</td>
</tr>
<tr>
<td>apex2</td>
<td>8</td>
<td>1602</td>
<td>21 \times 21</td>
</tr>
<tr>
<td>apex4</td>
<td>6</td>
<td>1147</td>
<td>34 \times 34</td>
</tr>
<tr>
<td>bigkey</td>
<td>3</td>
<td>1810</td>
<td>22 \times 22</td>
</tr>
<tr>
<td>clma</td>
<td>16</td>
<td>6779</td>
<td>42 \times 42</td>
</tr>
<tr>
<td>des</td>
<td>6</td>
<td>1263</td>
<td>19 \times 19</td>
</tr>
<tr>
<td>diffeq</td>
<td>14</td>
<td>987</td>
<td>16 \times 16</td>
</tr>
<tr>
<td>dsip</td>
<td>3</td>
<td>1362</td>
<td>19 \times 19</td>
</tr>
<tr>
<td>elliptic</td>
<td>18</td>
<td>2142</td>
<td>24 \times 24</td>
</tr>
<tr>
<td>ex1010</td>
<td>8</td>
<td>4050</td>
<td>33 \times 33</td>
</tr>
<tr>
<td>ex5p</td>
<td>7</td>
<td>950</td>
<td>16 \times 16</td>
</tr>
<tr>
<td>frisc</td>
<td>23</td>
<td>2320</td>
<td>25 \times 25</td>
</tr>
<tr>
<td>misex3</td>
<td>7</td>
<td>1178</td>
<td>18 \times 18</td>
</tr>
<tr>
<td>pdc</td>
<td>9</td>
<td>3901</td>
<td>32 \times 32</td>
</tr>
<tr>
<td>s298</td>
<td>15</td>
<td>1682</td>
<td>21 \times 21</td>
</tr>
<tr>
<td>s38417</td>
<td>11</td>
<td>4773</td>
<td>36 \times 36</td>
</tr>
<tr>
<td>s38584</td>
<td>9</td>
<td>4422</td>
<td>35 \times 35</td>
</tr>
<tr>
<td>seq</td>
<td>7</td>
<td>1427</td>
<td>20 \times 20</td>
</tr>
<tr>
<td>spla</td>
<td>8</td>
<td>3331</td>
<td>30 \times 30</td>
</tr>
<tr>
<td>tseng</td>
<td>13</td>
<td>781</td>
<td>14 \times 14</td>
</tr>
</tbody>
</table>
Toronto Benchmark: bigkey.blif

Initial placement  
Placement  
Global routing

Local connections  
Global connections

- Basic cell (logic)
- Basic Cell (routing inverter)
- Simple latch
- Primary input
- Primary output
Hybrid CMOS-Memristor FPGA: First Demo

Q. Xia et al. Nano Letters, 2009
CMOL DSP: convolution example

\[ T_{x,y} = \sum_{i=0}^{F-1} \sum_{j=0}^{F-1} S_{x+i,y+j} \varphi_{i,j} \]

\[ 0 \leq x, y \leq N - F - 1 \]

CMOL DSP:
~ 25 μs per frame

Aggressively scaled
CELL (45 nm): 3.5 ms

D. B. Strukov and K. K. Likharev, Tran. IEEE Nanotechnol. 7 151 (2007)