Monolithically Stackable Hybrid FPGA

Dmitri Strukov
ECE Department, UC Santa Barbara
Santa Barbara, CA, USA
strukov@ece.ucsb.edu

Alan Mishchenko
EECS Department, UC Berkeley
Berkeley, CA, USA
alanmi@eecs.berkeley.edu

Abstract — The paper introduces novel field programmable gate array (FPGA) circuits based on hybrid CMOS/resistive switching device (memristor) technology and explores several logic architectures. The novel FPGA structure is based on the combination of CMOL (Cmos + MOlecular scale devices) FPGA circuits and recent improvements and generalization of the CMOL concept to allow multilayer crossbar integration, compatibility with state-of-the-art foundries, and a wide range of available memristive crosspoint devices. Preliminary results indicate that with no optimization and only conventional CMOS technology, the proposed circuits can be at least ten times denser (and potentially faster) than CMOS FPGAs with the same design rules and similar power density. The second part of this paper shows that this performance can be further improved using optimal MUX-based logic architecture.

Keywords: FPGA, hybrid circuits, logic architecture, resistance switching, memristors, three dimensional circuits

I. INTRODUCTION

Field programmable gate arrays (FPGAs) become increasingly attractive alternatives to both conventional microprocessors and application specific integrated circuits (ASICs) for a variety of applications [1]. This is primarily due to the steady increase in their functionality making them practical for new applications. Indeed, with miniaturization of complementary metal oxide semiconductor (CMOS) technology during past decades, the capacity of FPGAs is now close to a million programmable logic blocks opening new opportunities, in particular in signal, image and network processing, cryptography, scientific and high performance computing. Further improvements of FPGA circuits, however, cannot rely on just lateral device shrinking since CMOS technology is getting close to its fundamental scaling limits [2], which motivates ongoing research.

One promising research direction relies on the fact that in contemporary FPGAs large portion of the area of the configurable fabric (as high 50% percent [3]) is taken by configuration bits typically implemented as static random access memory (SRAM). Further, the majority of the remaining resources is devoted to configurable routing so that the “useful” area is even less, from 5% to 15% of the total chip area depending on the architecture [3, 4] (and even less for multi-context FPGAs [5]). With the remaining scaling of CMOS technology, say, if scaling continues from today’s 45-nm to the future 18-nm node, the problem should only get worse since the number of routing resources is determined by Rent’s rule [6, 7]. Adding more metal layers would not help FPGAs (as opposed, e.g., to ASICs) since all active routing circuitry (configuration memory, buffers, multiplexers) is typically located on the silicon substrate.

We propose novel three-dimensional hybrid FPGA circuits (Fig. 1), which are based on CMOS technology and monolithically stackable resistance switching devices (memristors). Logic density can be improved significantly by lifting all configuration bits (including those used for logic functions) and large portion of the routing circuitry from the CMOS plane to multiple planes of metallization and memristive layers. The additional layers enable a very rich and area-efficient interconnect fabric, in which connections between overlapping wires in adjacent metallization layers can be set directly (programmed) by changing the resistance of the corresponding crosspoint memristor. This can be implemented with a variety of CMOS compatible material systems.

More broadly, the proposed FPGA architecture shares some features with CMOL (Cmos + MOlecular scale device hybrids) FPGA circuits [8, 9] and recent extensions of the CMOL concept to allow for multilayer crossbar integration, compatibility with state-of-the-art integrated foundries, and a wide range of memristive crosspoint devices [10].

In the following sections, we first briefly describe CMOL FPGA circuits and follow with the discussion of novel FPGA structure. In the second part of the paper, we present preliminary results of logic architecture optimization. We conclude the paper with a discussion of future work as well as advantages of the proposed concept over previously reported research and commercial FPGAs.

II. CMOL FPGA CIRCUITS

CMOL FPGAs are hybrid CMOS/nanodevice/nanowire programmable circuits [8, 11]. Such circuits were conceived to take advantage of emerging technologies such as nanoimprint technology [12], which allows patterning of nanoscale sub-CMOS features with F_{ss} half pitch, and self-assembled nanoscale molecular latching switches, and combine it with the flexibility and versatility of CMOS technology. Figures 2a-c show the generic structure of CMOL circuits with the following key features: (i) an “area interface” between the CMOS and nanosubsystem; (ii) the crossbar array which is rotated by an angle \( \alpha \) with respect to the mesh of CMOS-controlled vias; and (iii) a double decoding scheme that provides unique access to each crosspoint device.

(a) typical FPGA …  (b) … and the one with lifted config. bits

Figure 1. (a) Conventional FPGAs and (b) proposed circuits
More specifically, as Figures 2a-c show, two types of vias, one connecting to the lower (shown with blue dots) and the other to the upper (red dots) wire level in the crossbar, are arranged into a square array with sides $2\beta F_{\text{CMOS}}$, which is also equal to the side length of the “cells” grouping two vias of each kind. Here $\beta > 1$ is a dimensionless number that depends on the cell size (i.e., complexity) in the CMOS subsystem while $F_{\text{CMOS}}$ is the CMOS half pitch. The crossbar is rotated by an angle $\alpha = \arcsin(1/\beta)$ relative to the via array such that vias naturally subdivide the wires into fragments of length $2\beta F_{\text{CMOS}}/F_{\text{nano}}$. The factor $\beta$ is chosen from the spectrum of possible values $\beta = (r^2 + 1)^{1/2} \times F_{\text{nano}}/F_{\text{CMOS}}$, where $r$ is an integer so that the precise number of devices on the wire fragment is $r^2 - 1 \approx \beta^2 (F_{\text{CMOS}}/F_{\text{nano}})^2$.

The decoding scheme in CMOL is based on two separate address arrays (one for each level of wire in the crossbar so that there is a total of $4N$ edge channels to provide access to two different via controllers (one ‘blue’ and one ‘red’) in each of $N^2$ addressing cells in the CMOS plane. In contrast to standard memory arrays, in CMOL each control and data line pair electrically connects the peripheral input/outputs to a via, instead of a single memory element. In turn, each via is connected to a wire fragment in the crossbar.

The two perpendicular sets of wire fragments provide unique access to any crosspoint device even for large values of $\beta$. For example, selecting pins $\delta v$ and $b4$ (which are highlighted with blue and red circles, respectively) gives access to the leftmost of the two shown devices on Fig. 2b, while pins $\delta v$ and $e4$ for the rightmost device. The total number of crosspoint devices that can be accessed by the $N\times N$ array of CMOS addressing cells is $-N^2 \beta^2 (F_{\text{CMOS}}/F_{\text{nano}})^2$, which can provide a significant multiplicative factor when comparing CMOS to crossbar implementation, especially if the lithographic feature size of the crossbar is smaller than that of the CMOS. An alternate way of viewing this is that one can use complex CMOS circuitry built with a significantly larger feature size to address regular crossbars built at a much finer lithographic scale.

To implement logic operations, each cell (or supercell) hosts some CMOS logic gate, besides the (pass transistor) configuration circuitry.

In the original CMOL FPGA, the cell includes CMOS inverter, so that any multi-input NOR can be implemented with one inverter and molecular devices with diode-like functionality [8]. The later extensions of CMOL [9] suggested using more complex gates, such as NAND/AND (Fig. 2d). In the reconfiguration stage, all logic gates are disabled, e.g. by activating the LE signal, and any crosspoint device can be configured to the high or low resistive state, similar to that described above. Once configured, crosspoint devices do not change state and those set to the low resistive state represent electrical connection between logic gates.

For example, Fig. 2e shows that a particular input of the NAND/AND gate can be connected to any of the outputs of other NAND/AND gates that are within the “connectivity domain” (shown as straight connections on Fig. 2d and as a highlighted gray cells on Fig. 2b) by programming the corresponding molecular devices.

The cells which are not within the connectivity domain of each other can be connected by dedicating some CMOS cell for routing purposes, e.g., to use two invertors in sequence as a buffer, or a single NAND/AND gate as a repeater. This is why for better area efficiency the connectivity domains should be as large as possible. Note that, for convenience the connectivity domain in Fig. 2e is shown unrealistically small, while the typical connectivity domain sizes are of the order of

---

Figure 2. Original CMOL FPGA circuits [8]. (a) Cut-away illustration showing two types of vias connecting the CMOS control circuitry to the lower (blue) and upper (red) wire levels of the crossbar. (b) Top view of the crossbar structure showing $\alpha$ for $r = 3$. (c) Corresponding equivalent circuit diagram of the configuration logic in CMOS layer for the $N = 5$ primitive cell array. (d) Types of logic functionality of primitive cells considered in Refs. [8, 9]. (e) Example of the connectivity (schematically) of a particular input of a logic cell (shown with blue circle on panels b and c. For clarity, panel b shows only two crosspoint devices (green dots) which are used for the explanation in text.
The challenge is to keep the size of connectivity domain large, scale features but rather rely on lithographically defined vias. This process does not require sharply pointed nanoscale pins to contact sub-CMOS-using the standard CMOS process. This process does not and overlay alignment than those demanded by CMOS technology standards.

For example, the best commercially available tool, Imprio300 [12], has much worse defect density and overlay alignment than those demanded by CMOS technology standards.

1000 - 2000 CMOS cells, e.g., achieved with parameters $\beta = 4$, $F_{\text{CMOS}} = 45$ nm, and $F_{\text{nano}} = 4.5$ nm.

III. 3D HYBRID FPGAS

The implementation of original CMOL FPGA circuits would mainly rely on the progress of emerging technologies, such as nanoimprint lithography. While one can expect advances in nanoimprint technology in the future, currently it is still relatively immature to be used in high performance electronic circuits. For example, the best commercially available tool, Imprio300 [12], has much worse defect density and overlay alignment than those demanded by CMOS technology standards.

This is why, in contrast to the original design of CMOL circuits, we propose a more practical means of building them, using the standard CMOS process. This process does not require sharply pointed nanoscale pins to contact sub-CMOS-scale features but rather rely on lithographically defined vias. The challenge is to keep the size of connectivity domain large, which in this case equal to $\beta^2$ since $F_{\text{CMOS}} = F_{\text{nano}}$. This can be achieved by stacking multiple crossbar layers [10] using specific connection patterns to ensure the unique addresses for every crosspoint device and discussed next. The second, rather minor change to the original CMOL concept adopted for the novel circuits presented in this paper, is the use of much broader class of resistive switching crosspoint devices.

The CMOS area interface of the CMOL architecture actually can address a much larger number of crosspoint devices than are present in the single crossbar. To see the total size of the addressing space, the decoding is shown in Fig. 3c, which was constructed by flattening physical decoder layout shown in Fig. 2c into the corresponding graph representation to create a virtual two-dimensional crossbar. From Fig. 3c, the first level of decoding selects from $2N^2$ vias with $4N$ edge channels using a four-label address. The second level of decoding can, in principle, enable selection of $N^2$ crosspoint devices using the $2N^2$ internal lines (vias) of the area interface, but there are only $N^2\beta^2$ of them in a single array to be selected.

Thus, most of the addressing space available in CMOL is not used, and the virtual array is populated very sparsely. The solution is to stack multiple crossbar arrays on top of each other using just one set of vias to connect all of the arrays to the cells, as shown in Fig. 3a. The problem of stacking multiple layers becomes one of geometry to ensure that only one crosspoint device in all of the arrays can be addressed by any allowed set of four address labels (or pair of vias).

For example, one way to place the next crossbar in a sequence, is to translate it with respect to the fixed locations of one kind of via (e.g., blue vias in Fig. 3d) by a distance such that the contacted wire fragments in the new layer are connected to a new connectivity domain that is different from any preceding layer [10]. Other approaches that do not use extra metallization layers can be developed but with some sacrifice of the number of addressable crosspoint memristors.

The significant feature of such 3D architecture is that theoretical maximum number of crossbar layers, each defined with the same pitch, which one can stack and still uniquely access all of the crosspoint devices could be very large, of the order of $M = N^2/\beta^2$, thus allowing very large connectivity domains beyond those given by the scaling limits of lateral feature sizes.

IV. LOGIC ARCHITECTURE EXPLORATION

This section focuses on the basic fine-grained FPGAs while other promising techniques (such as increasing heterogeneity and coarsening by adding hardwired blocks and using bit-sliced routing [1, 7]) are orthogonal and can be considered complimentary to the proposed novel structure.
Traditionally, fine-grained FPGAs can be crudely divided into three categories, i.e., those based on sea-of-gates, look-up tables (LUTs), and multiplexers (MUXes). So far only the first variety is studied in detail for CMOL circuits [9, 11]. For example, Figs. 4a-c show an example of half-adder mapped onto the sea-of-gate fabric similar to the one used in [9]. Novel implementation of the LUT-based and pure-MUX-based fabrics, both composed of MUX cells, are shown in Figs. 4d-h. (Note that the conclusion in Ref. [8] that LUT-based design are not efficient are based on the assumption that each LUT appears as a separate crossbar structure with its own CMOS decoding, while here such overhead is effectively hidden within distributed crossbar style of CMOL.)

The implementation based on pure MUXes seems to be the most area efficient in this example. However it does not take into account the efficiency of logic synthesis tools.

In the following experiment, we compared several logic fabric styles by mapping combinational logic of 10 industrial designs.

The experiment was conducted on a Lenovo S10 workstation with Intel 2.66GHz Q9450 CPU and 8Gb RAM. No more than 300Mb was used by any of the designs.

First, the logic was transformed into an And-Inverter Graph (AIG) and optimized using combinational synthesis in ABC (command dc2) [13]. The resulting AIG statistics (the number of two-input AND gates and the number of gate levels) can be found in the second and third columns of Table 1 and 2. The last two lines in the tables show the arithmetic and geometric averages of values in the corresponding columns.

Three different evaluations were performed: (1) Single-cell standard-cell library mapping (Table 1). The optimized AIG was mapped into a given standard-cell library using cut-based area-oriented technology mapper map—as [15]. The cost is the number of gates (unit-area model). (2) k-LUT mapping (Table 2, left). The optimized AIG was mapped into a library composed of k-LUTs using priority-cut-based area-oriented mapper if—a [16]. The cost of a k-LUT was assumed to be 1 (unit-area model). (3) Hybrid 2:1 MUX and k-LUT mapping (Table 2, right). The optimized AIG was mapped into a library of 2:1 MUXes and k-LUTs using priority-cut-based area-oriented mapper if—a [16]. The cost of a 2:1 MUX was assumed to be 1, while the cost of a k-LUT was $2^k - 1$.

In Table 2, several standard-cell libraries were used. Each of them contained cells of one type, without inverters at the inputs/outputs of each cell. Note that the number of nodes in the optimized AIG is smaller, in terms of the number of nodes, compared to the result of mapping for NAND2, because the AIG is a network of AND2s with complemented inputs. To implement the complements, additional NAND2 gates are needed. Note also that the AIG is smaller than AND2/OR2 library because inverters are free in the AIG, while they required additional gates in the library.

Mapping all 10 designs into one library took about one minute on a single core. The results were verified by a SAT-based combinational equivalence checker &ece [14].

V. DISCUSSION AND SUMMARY

Results in Tables 1 and 2 indicate that expressive power of MUX cells is the best, if used a single-function library, without complemented inputs. Increasing the LUT size (k) helps reduce the total number of k-LUTs. On the other hand, increasing the LUT size (k) does not help reduce the cost of k-LUT+MUX mapping. When k-LUT+MUX are used, the cost is less than MUX only, by a small margin.

More generally, these results indicate that significant improvement over previously studied logic architectures is possible and should be explored while using a larger set of sequential benchmarks and performing placement and routing.

There are at least two different approaches to implement sequential logic. The first approach is to introduce into logic fabric dedicated flip-flops cells and the clock network [9, 11]. The problem with this approach is that these resources will be fixed. As a result, when circuits are purely combinational and do not require pipelining these resources will be wasted.

Another approach is to allocate and configure flip-flops on demand, e.g., by connecting NAND/AND cells in the cross-coupled fashion. The only fixed resource in this case would be special CMOS clock cells similar to special control cells introduced in Ref. [17]. These can be very sparse so that the hardwired penalty is minimal. Using such special cells the clock signal would be selectively routed by configuring corresponding memristive devices attached to the crossbar wires of the clock cells, so that a flip-flop can be configured in any location of the programmable fabric.

Performance of novel FPGAs can be crudely estimated by leveraging previous work on similar circuits. For example,
architecture might have several distinct advantages: First, it is
commercial devices. However, the proposed FPGA
circuits can match connectivity of circuits in Ref. [9], and
therefore their density, with only 10 crossbar layers. This is
just slightly larger than the number of metal layers in today’s
microprocessors while optimal routing and logic architectures
should make this number smaller. Indeed, in the original
CMOL FPGA routing architecture is based on direct links for
local connections, i.e. for the gates within connectivity
domains of each other, and (multiple) hops through the CMOS
subsystem to implement global connections.

Routing efficiency could be significantly improved by
designing interconnect hierarchy similar to the segmented
architecture of the traditional FPGAs. For example, typical
wire-length in integrated circuits follows Poisson distribution
as shown on Fig. 3 and thus

For example, the circuitry in the first several crossbar layers
should make this number smaller. Indeed, in the original
CMOL FPGA routing architecture is based on direct links for
local connections, i.e. for the gates within connectivity
domains of each other, and (multiple) hops through the CMOS
subsystem to implement global connections.

The idea of lifting configuration bits from CMOS plane
can be traced to several earlier FPGA architectures and even
commercial devices. However, the proposed FPGA
architecture might have several distinct advantages: First, it is

Table 1: Evaluation of single-cell libraries (area: Gate = 1)

<table>
<thead>
<tr>
<th>Design</th>
<th>AIG nodes</th>
<th>AIG NAND2</th>
<th>NOR2</th>
<th>NAND3</th>
<th>NOR3</th>
<th>NAND4</th>
<th>NOR4</th>
<th>(N)AND2</th>
<th>MUX</th>
<th>AOI</th>
<th>Davio</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(ab)</td>
<td>(a+b)</td>
<td>(a+b+c)</td>
<td>(a+b+c+d)</td>
<td>ab</td>
<td>(a+b+c+d)</td>
<td>ab</td>
<td>(a+b+c+d)</td>
<td>ac+lab</td>
<td>l(ab+cd)</td>
<td>(ab)c</td>
</tr>
<tr>
<td>01</td>
<td>123247</td>
<td>113</td>
<td>137887</td>
<td>179922</td>
<td>131753</td>
<td>137296</td>
<td>131753</td>
<td>124495</td>
<td>130835</td>
<td>107522</td>
<td>106366</td>
</tr>
<tr>
<td>02</td>
<td>68216</td>
<td>18</td>
<td>103399</td>
<td>106745</td>
<td>89636</td>
<td>92698</td>
<td>89636</td>
<td>88026</td>
<td>72641</td>
<td>53983</td>
<td>66404</td>
</tr>
<tr>
<td>03</td>
<td>127551</td>
<td>72</td>
<td>178785</td>
<td>184430</td>
<td>139151</td>
<td>144616</td>
<td>139151</td>
<td>135748</td>
<td>135729</td>
<td>89951</td>
<td>134770</td>
</tr>
<tr>
<td>04</td>
<td>179318</td>
<td>75</td>
<td>281495</td>
<td>261164</td>
<td>215146</td>
<td>215146</td>
<td>162644</td>
<td>188858</td>
<td>140396</td>
<td>106921</td>
<td>171578</td>
</tr>
<tr>
<td>05</td>
<td>180861</td>
<td>75</td>
<td>263406</td>
<td>267143</td>
<td>210429</td>
<td>208813</td>
<td>210429</td>
<td>193573</td>
<td>190188</td>
<td>138306</td>
<td>165530</td>
</tr>
<tr>
<td>06</td>
<td>361308</td>
<td>73</td>
<td>526536</td>
<td>523229</td>
<td>420397</td>
<td>420397</td>
<td>420397</td>
<td>386572</td>
<td>379914</td>
<td>276327</td>
<td>330872</td>
</tr>
<tr>
<td>07</td>
<td>113817</td>
<td>40</td>
<td>176017</td>
<td>179922</td>
<td>131753</td>
<td>137296</td>
<td>131753</td>
<td>124495</td>
<td>130835</td>
<td>107522</td>
<td>106366</td>
</tr>
<tr>
<td>08</td>
<td>179521</td>
<td>44</td>
<td>281495</td>
<td>261164</td>
<td>215146</td>
<td>215146</td>
<td>162644</td>
<td>188858</td>
<td>140396</td>
<td>106921</td>
<td>171578</td>
</tr>
<tr>
<td>09</td>
<td>180861</td>
<td>75</td>
<td>263406</td>
<td>267143</td>
<td>210429</td>
<td>208813</td>
<td>210429</td>
<td>193573</td>
<td>190188</td>
<td>138306</td>
<td>165530</td>
</tr>
<tr>
<td>10</td>
<td>361308</td>
<td>73</td>
<td>526536</td>
<td>523229</td>
<td>420397</td>
<td>420397</td>
<td>420397</td>
<td>386572</td>
<td>379914</td>
<td>276327</td>
<td>330872</td>
</tr>
<tr>
<td>Ave</td>
<td>1.000</td>
<td>0.985</td>
<td>0.798</td>
<td>0.781</td>
<td>0.798</td>
<td>0.721</td>
<td>0.707</td>
<td>0.507</td>
<td>0.545</td>
<td>0.507</td>
<td>0.660</td>
</tr>
<tr>
<td>Geo</td>
<td>1.000</td>
<td>0.983</td>
<td>0.797</td>
<td>0.780</td>
<td>0.797</td>
<td>0.718</td>
<td>0.705</td>
<td>0.543</td>
<td>0.543</td>
<td>0.659</td>
<td>0.699</td>
</tr>
</tbody>
</table>

Table 2: Evaluation of k-LUT libraries (area: LUT = 1) and 2:1MUX / k-LUT libraries (area: MUX = 1, LUT = 2k – 1)

<table>
<thead>
<tr>
<th>Design</th>
<th>LUT3 only</th>
<th>LUT4 only</th>
<th>LUT5 only</th>
<th>LUT6 only</th>
<th>LUT3+Max</th>
<th>LUT4+Max</th>
<th>LUT5+Max</th>
<th>LUT6+Max</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>66503</td>
<td>48039</td>
<td>37209</td>
<td>32554</td>
<td>102791</td>
<td>102785</td>
<td>102784</td>
<td>102782</td>
</tr>
<tr>
<td>02</td>
<td>34485</td>
<td>26310</td>
<td>23439</td>
<td>19656</td>
<td>51759</td>
<td>51749</td>
<td>51749</td>
<td>51749</td>
</tr>
<tr>
<td>03</td>
<td>53463</td>
<td>38913</td>
<td>32379</td>
<td>27731</td>
<td>82950</td>
<td>82904</td>
<td>82877</td>
<td>82877</td>
</tr>
<tr>
<td>04</td>
<td>79859</td>
<td>48077</td>
<td>44096</td>
<td>39442</td>
<td>137224</td>
<td>137224</td>
<td>137224</td>
<td>137224</td>
</tr>
<tr>
<td>05</td>
<td>87677</td>
<td>65957</td>
<td>53173</td>
<td>46993</td>
<td>131969</td>
<td>131934</td>
<td>131930</td>
<td>131926</td>
</tr>
<tr>
<td>06</td>
<td>17512</td>
<td>131740</td>
<td>106221</td>
<td>93837</td>
<td>263716</td>
<td>263679</td>
<td>263679</td>
<td>263679</td>
</tr>
<tr>
<td>07</td>
<td>63883</td>
<td>44799</td>
<td>37050</td>
<td>30763</td>
<td>106174</td>
<td>106174</td>
<td>106174</td>
<td>106168</td>
</tr>
<tr>
<td>08</td>
<td>62424</td>
<td>49270</td>
<td>40174</td>
<td>35303</td>
<td>119348</td>
<td>119174</td>
<td>118790</td>
<td>118790</td>
</tr>
<tr>
<td>09</td>
<td>52406</td>
<td>42152</td>
<td>37108</td>
<td>33768</td>
<td>75941</td>
<td>75941</td>
<td>75941</td>
<td>75941</td>
</tr>
<tr>
<td>10</td>
<td>20885</td>
<td>16304</td>
<td>14484</td>
<td>13261</td>
<td>32583</td>
<td>32577</td>
<td>32577</td>
<td>32570</td>
</tr>
<tr>
<td>Ave</td>
<td>1.000</td>
<td>0.739</td>
<td>0.792</td>
<td>0.547</td>
<td>1.000</td>
<td>1.000</td>
<td>0.999</td>
<td>0.999</td>
</tr>
<tr>
<td>Geo</td>
<td>1.000</td>
<td>0.737</td>
<td>0.791</td>
<td>0.544</td>
<td>1.000</td>
<td>1.000</td>
<td>0.999</td>
<td>0.999</td>
</tr>
</tbody>
</table>
necessarily the best device for our purposes, are already in the development stage [31] while other RRAM technologies are just slightly lagging behind in terms of maturity [32, 33].

Finally, the scheme of constructing 3D circuits is compatible with state-of-the-art integrated circuit foundries and potentially cost-efficient. This is because the features in all crossbar layers are similar and can be patterned using the same set of masks, while adding new layer does not require changes in the CMOS substrate [10].

Note that the specific representation of a CMOL in Figs. 2-4 is chosen for convenience while more manufacturable layouts are available. The one shown in Fig. 5 eliminates the need for dense vias and is based on only Manhattan features while still incorporating all the key properties of CMOL.

Secondly, it is the architectural aspects of the proposed FPGA circuits. (1) All configuration bits, including those used for logic functions, and a large portion of the routing circuitry (e.g., all multiplexers) are lifted from the CMOS plane to multiple planes of metallization and programmable connections, resulting in a rich and area-efficient interconnect fabric. It is more area efficient than just lifting memory bits which control CMOS pass transistors for routing [23, 25, 27]. Since connections are essentially electrical shorts, there is no non-Ohmic drop of voltage on them (e.g., as opposed to that of a pass transistor), so that the need for buffering in routing is relaxed. (2) The double decoding scheme of CMOL allows for a relatively small programmability over-head. Only one other type of FPGAs based on CMOL was proposed before [28] but it cannot be generalized to monolithic stacking because of wafer bonding. (3) The proposed FPGA circuit architecture allows for higher logic utilization due to flexible logic/routing/memory resource allocation [11].

The concept introduced in this paper shows high promise. To substantiate performance advantages, and justify various ideas presented in the paper, more rigorous interdisciplinary research is required. Most importantly, this includes development of design automation tools and modeling of the most promising memristive devices. The latter is especially important since memristors cannot combine all the desired properties because of the intrinsic tradeoffs imposed by device physics and circuit level operation.

REFERENCES