

A Reconfigurable Architecture for Hybrid CMOS/Nanodevice Circuits *

Dmitri B. Strukov and Konstantin K. Likharev

Stony Brook University
Stony Brook, NY 11794-3800, U.S.A

dstrukov@ic.sunysb.edu, klicharev@cc.notes.sunysb.edu

ABSTRACT

This report describes a preliminary evaluation of possible performance of an FPGA-like architecture for future hybrid “CMOL” circuits which combine a semiconductor-transistor (CMOS) stack and two-level nanowire crossbar with molecular-scale two-terminal nanodevices (programmable diodes) formed at each crosspoint. This cell-based architecture is based on a uniform CMOL fabric, with four-transistor CMOS cells, which may be reconfigured around defective nanodevices to provide high defect tolerance. Using semi-custom design automation tools we have evaluated CMOL performance for the Toronto 20 benchmark set, so far without optimization of several parameters including the power supply voltage and nanowire pitch. The results show that even without such optimization, CMOL FPGA circuits may provide an advantage of more than two orders of magnitude in the area over the traditional CMOS FPGA with the same CMOS design rules, at acceptable power consumption and potentially high defect tolerance.

Categories and Subject Descriptors

B.6.1 [Logic Design]: Design styles—*logic arrays*; B.7.1 [Integrated Circuits]: Types and Design Styles—*advanced technologies*

General Terms

Design

Keywords

Programmable logic, integrated hybrid circuits, nanoelectronics, programmable interconnect

1. INTRODUCTION

It is now believed that the growing problems with scaling of CMOS technology [9, 15] may be only circumvented by a

*(Produces the permission block, copyright information and page numbering).

radical paradigm shift from lithographic based fabrication to the so-called “bottom-up” approach [11, 19, 25]. In this approach, the smallest active devices of integrated circuits are not defined lithographically but assembled from parts with fundamentally defined size and structure, e.g., few-nm-scale molecules. Since such devices have limited functionality, most efforts in the development of nanoelectronic architectures focus on hybrid CMOS / nanodevice circuits - see, e.g., Refs. 6, 10, 13, 14, 21, 23, and also recent reviews 7, 16, 22. In most of the proposed hybrid circuits, relatively large silicon MOSFETs are used for signal restoration, long-range communications, I/O, testing / bootstrapping, and some other critical functions, while a set of dense nanodevices provides most of information storage and signal processing.

We believe that the most promising species of CMOS / nanodevice hybrids are “CMOL” circuits [15, 16]. As in several earlier hybrid proposals, nanodevices in CMOL circuits are formed at each crosspoint of a “crossbar” array, consisting of two levels of nanowires (Fig. 1). However, in order to overcome the CMOS / nanodevice interface problems pertinent to earlier concepts, in CMOL circuits the interface is provided by pins that are distributed all over the circuit area, on the top of the CMOS stack. By activating two pairs of perpendicular CMOS lines, two pins (and two nanowires they contact) may be connected to CMOS data lines (Fig. 1b). As Fig. 1c shows, pins of each type (reaching to the lower and upper nanowire level) are arranged into a square array with side $2\beta F_{\text{CMOS}}$, where F_{CMOS} is the half-pitch of the CMOS subsystem, and β is a dimensionless factor larger than 1 that depends on the CMOS cell complexity. Relative to the CMOS pin array, the nanowire crossbar is turned by angle $\alpha = \arcsin(F_{\text{nano}}/\beta F_{\text{CMOS}})$, where F_{nano} is the nanowiring half-pitch. Figure 1c illustrates the fact that this approach allows a unique access to any nanodevice, even if $F_{\text{nano}} \ll F_{\text{CMOS}}$ - see Ref. 16 for a detailed discussion of this point¹. If the nanodevices have a sharp current threshold, like the usual diodes, such access allows to test each of them. Moreover, if such a diode may be switched be-

¹In the initial version of CMOL circuits [16], interface pins leading to the upper nanowire level would pass between nanowires of the lower level. This had restricted the maximum CMOL circuit yield (without nanoscale alignment) to 50%. This work is based on an improved version of CMOL, with oxide-covered pins intentionally interrupting the lower nanowires - see Fig. 1. While keeping our prior results on CMOL FPGA [23] valid, this modification improves the circuit yield substantially, raising its theoretical upper bound to 100%.

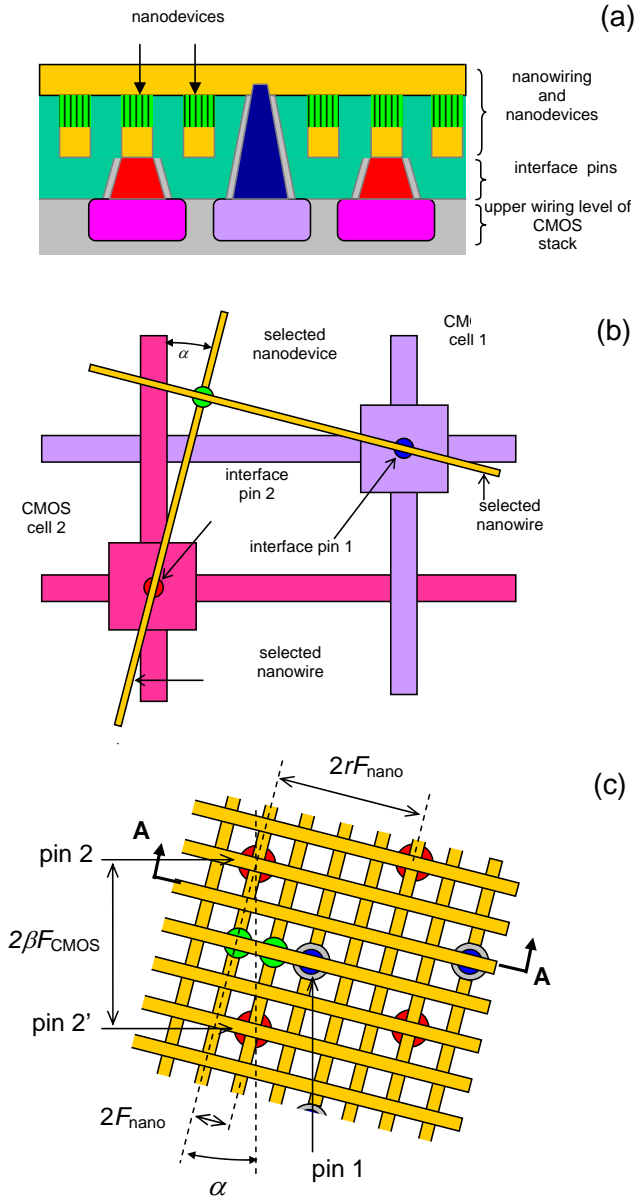


Figure 1: Low-level structure of the generic CMOL circuit: (a) schematic side view; (b) the concept of addressing a particular nanodevice, and (c) zoom-in on several adjacent pins. The last panel shows that any nanodevice may be addressed via the appropriate pin pair (e.g., pins 1 and 2 for the left of the two shown devices, and pins 1 and 2' for the right device). On panel (b), only the activated CMOS lines and nanowires are shown, while panel (c) shows only two devices. (In reality, similar nanodevices are formed at all nanowire crosspoints.) Also disguised on panel (c) are CMOS cells and wiring.

tween two internal states, e.g., as the single-electron latching switch [8, 15], each device may be turned on or off by applying voltages $\pm V_W$ to the selected nanowires, so that voltage $V = \pm 2V_W$ applied to the selected nanodevice exceeds the corresponding switching threshold, while half-selected devices (with $V = \pm V_W$) are not disturbed [16].

Earlier we have shown that CMOL circuits may be used to build several types of reconfigurable, highly defect-tolerant circuits including terabit-scale memories [16, 24] and mixed-signal neuromorphic circuits capable of advanced information processing [26]. However, the most important application of CMOL technology may be in reconfigurable Boolean logic circuits [23] whose structure reminds the so-called cell-based FPGAs [18]. In these “CMOL FPGA” circuits (Fig. 2 a,b) an elementary CMOS cell includes two pass transistors and one inverter, and is connected to the nanowire/ molecular subsystem via two pins. During the configuration process the inverters are turned off, and the pass transistors may be used for setting the binary state of each molecular device, just as in CMOL memories [16, 23]. Each pin of a CMOS cell can be connected through a nanowire-nanodevice-nanowire link to each of $M \equiv 2r^2 - 2r - 2$ other cells within a square-shaped “connectivity domain” around the pin (painted light-gray in Fig. 2a). Figure 3 shows how such fabric may be configured for the implementation of a NOR gate. This is already sufficient to implement any logic function, though other gates (e. g., NAND) are clearly possible.

In our previous work [23] we have analyzed defect tolerance and performance of only two combinational (latch-free) logic circuits which might be manually mapped on the CMOL FPGA fabric: a 32-bit Kogge Stone adder and a 64-bit full crossbar. The results have implied that CMOL FPGA may provide an advantage beyond two orders of magnitude over purely CMOS FPGA circuits, at manageable power consumption, simultaneously with high defect tolerance (above 20% of bad nanodevices). However, these results were insufficient to evaluate the benefits of the CMOL FPGA concept for real-life computational tasks. The goal of this report is to describe our next step in this direction: an analysis of CMOL FPGA performance for all circuits of the Toronto 20 benchmark set [1]. For this purpose, we have developed semi-custom software for an automated CMOL FPGA design flow.

2. HARDWARE ARCHITECTURE

A genuine optimization of CMOL FPGA circuit architectures would require a completely new set of CAD tools, whose development is a daunting task. At this preliminary stage, our choice was instead to get as much leverage as possible from the existing software used for mapping and architecture exploration for semiconductor logic, in particular, from the design automation tools for island-type CMOS FPGAs [5].

In order to use these tools, we have focused on a specific simple type of space-periodic CMOL circuits. (Though these circuits are a generalization of the flat CMOL FPGA fabric considered in Ref. 23, it is still a subset of all possible CMOL circuits, so that the results described below may be certainly improved in future.) The circuits are formed from a periodic mesh of square-shaped “tiles” (Fig. 4a). Each

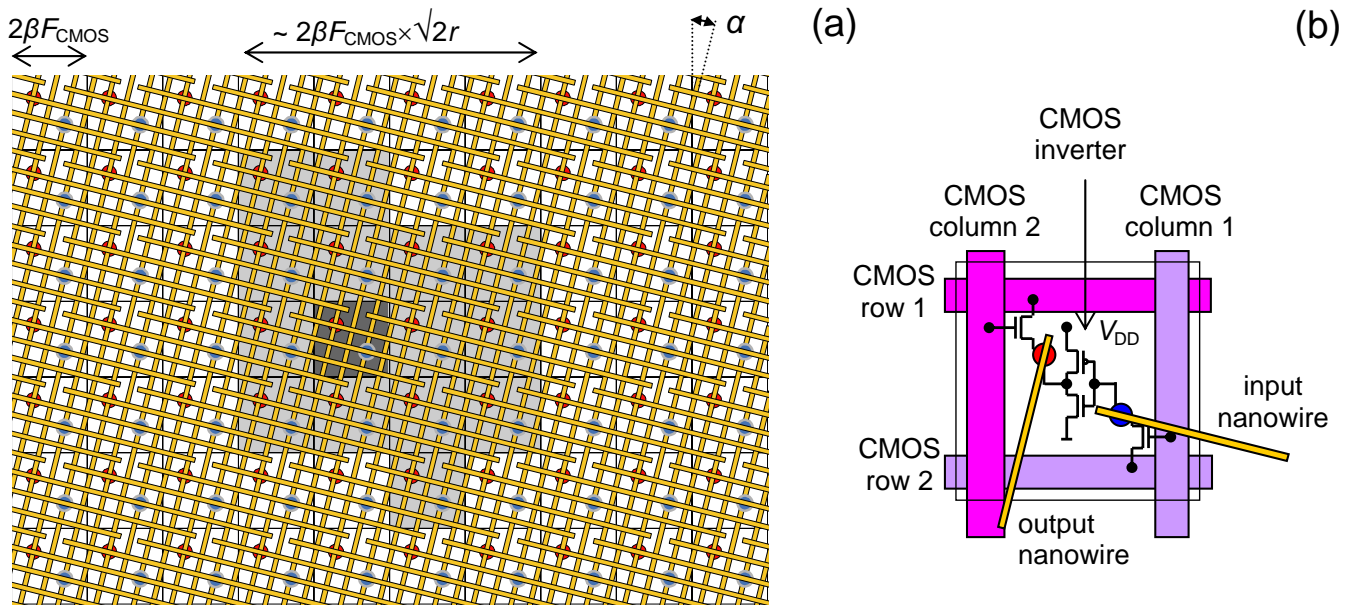


Figure 2: CMOL FPGA: (a) the general structure of the circuit and (b) a single CMOS cell. In panel (a), the cells painted light-gray may be connected to the input pin of a specific cell (shown dark-gray) via a single nanowire-nanodevice-nanowire link. For the sake of clarity, panel (b) shows only two nanowires (that contact the given cell).

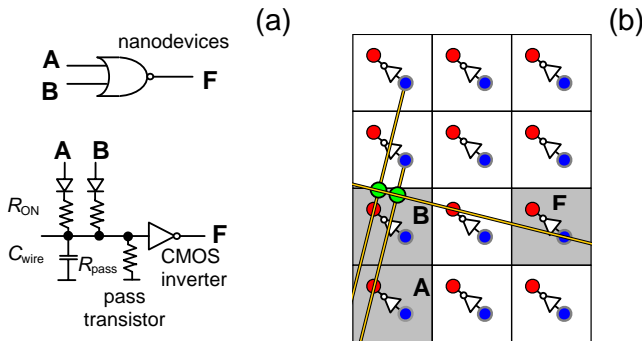


Figure 3: (a) NOR gate: (a) equivalent circuit and (b) physical implementation in CMOL. For the sake of clarity, only two inputs are shown. Panel (b) shows only the nanowires used inside the gate.

tile consists of a shell of T regular CMOS cells (Fig. 2b) surrounding a single “CMOS latch” cell. The latch is just a D flip-flop implemented in the CMOS subsystem, with a connection to 8 interface pins - see Fig. 4b. Note that all four pins of each (either input or output) group are always connected, so that the nanowires they contact always carry the same signal. This means that at the configuration and reconfiguration stages [16, 23], groups of four nanodevices sitting on these wires (rather than individual devices) may be turned on or off only together. A simple analysis shows that this does not impose any restrictions on the CMOL FPGA fabric functionality.

CMOS layout estimates have shown that the latch cell requires an area approximately four times larger than that of

a regular cell. As a result, for this analysis we have accepted $T = 12$, so that the total tile area is $T + 4 = 16$ CMOS cells. This provides a latch / logic resource ratio comparable to those of conventional CMOS FPGAs. In fact, the 4-input parity function (the worst-case Boolean function of 4 inputs) can be implemented with 14 4-input NOR gates, while an average 4-input Boolean function requires much less (6 to 8) such gates. Hence each CMOL tile is crudely similar to the basic logic element with a 4-input LUT and one latch [5].

3. DESIGN AUTOMATION

The convenience of the proposed CMOL hardware structure is that, from the software point of view, a CMOL tile can be treated in the same way as that of the island-type CMOS FPGA. Indeed, consider design flow shown in Fig. 5. Using the SIS package [20], we first map the original pre-optimized logic circuit onto a network of NOR gates (with a certain maximum fan-in) and latches (if any), to produce a netlist in blif format. Next, we fix a certain number (N) of CMOS cells inside each tile to perform logic operations, while the rest $T - N$ CMOS cells are committed to routing. The netlist of NOR gates is then partitioned into logic clusters of N gates with the help of the T-VPack program [5]. This code, while originally written to pack LUTs, can work as well for NOR gates, since the representations in the blif format for both gate libraries are the same and any CMOL NOR gate occupies exactly one CMOS cell regardless of its fan-in [16, 23]. The only modification which has been made to T-VPack is the addition of a latch counter to prevent packing of more than one flip-flop into one cluster.

The logic clusters of CMOS cells are then mapped on the CMOL tile fabric (one cluster on a tile) using the VPR [4,5] tool with a modified cost function. More specifically, the

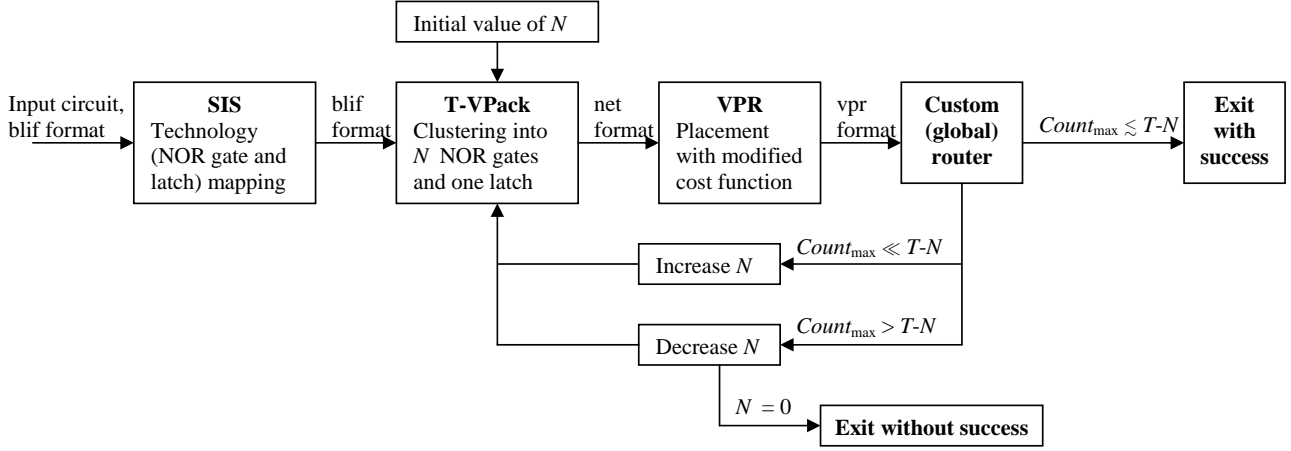


Figure 5: CMOL FPGA design flow used in this work.

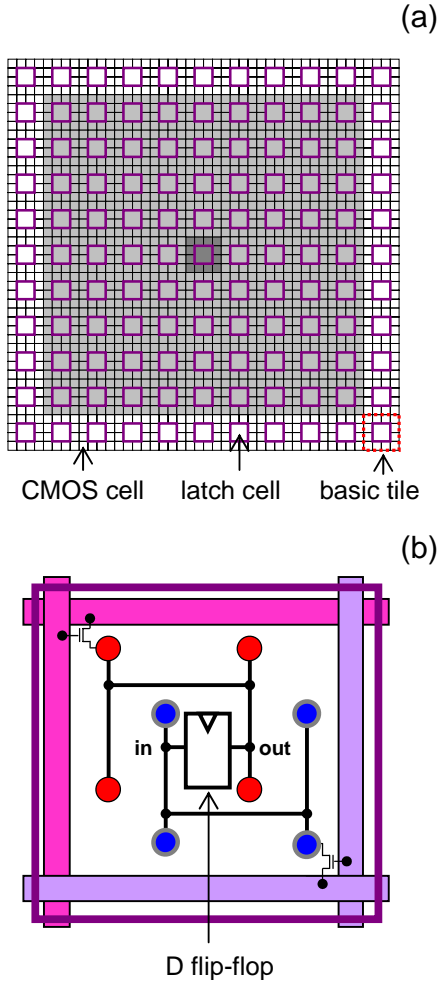


Figure 4: (a) CMOL FPGA tile fabric with latches and (b) CMOS latch cell structure. In panel (a), any cell in tiles painted light gray can be connected to any cell of the tile shown dark-gray via a single nanowire-nanodevice-nanowire link.

cost function for some cluster (tile) located at the position x_0, y_0 and connected to N_{clusters} other clusters is calculated as

$$Cost = \sum_{i=1}^{N_{\text{clusters}}} \lceil \frac{\max(|x_0 - x_i|, |y_0 - y_i|) - 1}{\mathbb{R}} \rceil, \quad (1)$$

where the x, y is a position of the corresponding tile inside the array (defined exactly as in Ref. 5), while $\mathbb{R} = \lfloor (\sqrt{2}r/\sqrt{T} + 4 - 1)/2 \rfloor$ is the “tile connectivity radius”. This radius characterizes the linear size of a “connectivity domain” of tiles such that any cell from the tile of that domain can be connected to any cell of the initial tile directly, i.e. via one nanowire-nanodevice-nanowire link. This domain is a square with a side of $2\mathbb{R} + 1$ tiles. For example, Fig. 4a shows a cluster connectivity domain for a realistic case $\sqrt{2}r = \beta F_{\text{CMOS}}/F_{\text{nano}} = 40$, i.e. $\mathbb{R} = 4$. Using the new cost function, VPR tries to place connected logic clusters close to each other, ideally within tile connectivity radius.

The next routing step, which is required to connect clusters located within the distance larger than tile connectivity radius, is performed using a custom tool. The algorithm starts with routing nets one at a time by assigning routing cells (inverters) to tiles. For any “source” cluster to “sink” cluster connection, the algorithm uses the minimum number of hops physically possible for a given placement. If the net has more than one “sink” cluster, the program tries to minimize the total number of routing cells by sharing them among different connections. In the meantime, if a given net can be routed with the same minimal number of cells using different paths, the preference is given to those paths which utilize the tiles with the least utilized routing cells. To accomplish this task, we keep an occupation counter of a total number of routing cells ($Count$) requested by algorithm in each tile. At the start of the routing procedure, the counter value is set to either zero (if the cluster, assigned to the given tile, is fully packed by T-VPack) or to the corresponding negative number (in the opposite case, i.e. if the cluster has less than N logic cells).

After processing all the nets, the algorithm makes the second step: it identifies the nets which were routed using tiles with the maximum number of requested routing cells

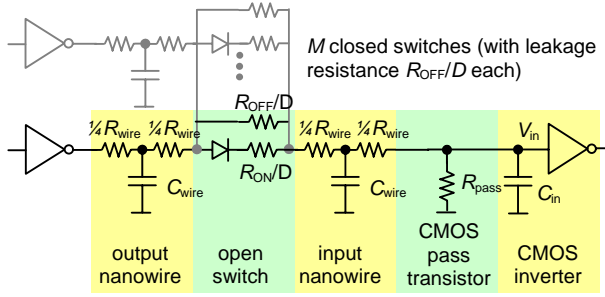


Figure 6: The equivalent circuit of a CMOL logic stage.

($Count_{\max}$) and tries to reroute those nets by using the same algorithm as in the first step, starting with the longest nets (in a hope that those nets have more paths to chose from and hence will be rerouted around the congestions). This step yields a more even spread of routing resources and thus improves $Count_{\max}$.

Finally, when $Count_{\max}$ can be no longer improved it is compared with $T - N$, and if it is larger, the whole design flow (starting from the T-VPack stage) is repeated with a reduced value of N (Fig. 5).² Alternatively, if $Count_{\max}$ is much smaller than $T - N$, the cycle is repeated with an increased value of N . In case when $Count_{\max} \leq N$ (in our particular case we required that the difference is less than 2), the algorithm stops.

4. PERFORMANCE CALCULATION

Generally, our performance analysis follows that of described in Ref. 23 with some simplifications. In particular, at this stage we have not yet optimized the nanowire pitch, but rather have simply calculated the performance for a case $F_{\text{CMOS}} = 45 \text{ nm}$, $F_{\text{nano}} = 4.5 \text{ nm}$ which seems technologically plausible at the initial stage of CMOL technology development [16]. Neither was the power supply voltage optimized; we have just accepted the value $V_{\text{DD}} = 0.3 \text{ V}$, which is in the ballpark of the results of V_{DD} optimization for the two circuits analyzed in Ref. 23 for these values of F_{CMOS} and F_{nano} .

4.1 Area

We will show below that the typical current through molecular devices in the ON state is of the order of $1 \mu\text{A}$. With a saturation current density of $1000 \mu\text{A} \mu\text{m}^{-1}$, typical for the long term CMOS projections [2], such current may be provided with a MOSFET channel as narrow as 1 nm . Hence, we can assume that all four transistors of the CMOS cell are of the minimum width.³ Using SCMOS design rules [17], the CMOS cell area is about $A_{\text{cell}} = 64(F_{\text{CMOS}})^2$, i.e., $\beta \approx 4$. We assume that the latch can be fitted into four CMOS cells

²In some cases, e.g., for very large circuits with rich connectivity, $Count_{\max}$ may be larger than $T - N$ even for $N = 1$, though this have never happened for any of the circuits from the considered benchmark set. Such situation would require a change in hardware parameters - say, a reduction of F_{nano} to increase r and hence the tile connectivity domain.

³The minimum-width CMOS inverter in the cell can provide a very large (> 20) fan-out without any latency degradation.

using the most compact circuit style from Ref. 12. Additional area overhead associated with auxiliary circuitry such as clock buffers, peripheral logic for reconfiguration, etc. has not been taken into account at this stage, but is probably negligible.

4.2 Nanowires and Nanodevices

Even taking into account the additional surface scattering in nanowires [23], estimated resistance between the center and the end of a nanowire fragment, of the length $(\beta F_{\text{CMOS}})^2 / F_{\text{nano}} = 7.2 \mu\text{m}$, is about $20 \text{ K}\Omega$. This resistance is negligible, because it is connected in series with the parallel resistance of D nanodevices in a single nanowire crosspoint (Fig. 6). Even in the ON state, the latter resistance is an order of magnitude larger - see Subsection 4.4 below. With the wire capacitance per unit length, which was calculated earlier [23], to be close to $0.2 \text{ fF}/\mu\text{m}$, the capacitance of the full nanowire fragment is about $C_{\text{wire}} = 2.9 \text{ fF}$.

Based on the experimental data for self-assembled monolayers the footprint of a single molecule may be estimated as 0.25 nm^2 ; so for the number of molecules per crosspoint we have used the value $D = F_{\text{nano}}^2 / 0.25 = 81$.

As in our previous calculations, the smallest acceptable resistance R_{ON} of a single molecular device in the ON state is defined by the maximum manageable power density $p_{\text{max}} = 200 \text{ W}/\text{cm}^2$ [2]. For our estimates we have taken into account only the static power dissipated in nanodevices turned ON. (See Fig. 15a of Ref. 23 for justification of this point.) Hence, the resistance of the nanodevice R_{ON} can be found from

$$R_{\text{ON}} = \frac{DN_{\text{cell}}(V_{\text{DD}})^2}{2A_{\text{cell}}p_{\text{max}}}, \quad (2)$$

where N_{cell} is the average number of crosspoints turned ON per one CMOS cell.

Though we have not optimized V_{DD} , we have still checked that the ratio of resistances in the OFF and ON states provided by the second-order quantum effect of elastic cotunneling ($R_{\text{OFF}}/R_{\text{ON}} = R_{\text{ON}}/R_{\text{Q}}$) is less than the maximum value of this ratio, which is determined by classical thermal activation ($R_{\text{OFF}}/R_{\text{ON}} \leq \cosh^2(V_{\text{DD}}/2k_{\text{B}}T)$). Here $R_{\text{Q}} \equiv \hbar/e^2 \approx 4.1 \text{ k}\Omega$ is the quantum unit of resistance. Hence our estimates based on the former formula are correct - for more details, see Ref. 23.

4.3 Delay

In order to decrease the NOR gate latency τ_0 we minimize V_{in} by choosing an appropriate resistance of the pass transistor $R_{\text{pass}} \simeq V_{\text{in}}/V_{\text{DD}} \times R_{\text{ON}}/D$ (Fig. 6). More specifically, we use for V_{in} a condition similar to that used in Ref. 23, i.e. require a bit error rate of one gate less than 10^{-28} . (This corresponds to the mean time to failure of the whole VLSI circuit of about 10 000 hours [2].) However, in the condition resulting from this requirement,

$$V_{\text{in}} \geq \Delta V_{\text{T}} = 23\sqrt{k_{\text{B}}T_{\text{eff}}/C_{\text{wire}}}, \quad (3)$$

we take T_{ef} to be the effective temperature $T_{\text{ef}} = (eV_{\text{DD}}/2k_{\text{B}} \times \coth(eV_{\text{DD}}/2k_{\text{B}}T))$ determined by shot noise in nanoscale devices (diodes) rather than the physical temperature.⁴ For our parameters, $T_{\text{ef}} \approx 2,000$ K.

The digital noise resulting from coupling of the input of CMOS cell to output of others CMOS cells through nanodevices turned off may be neglected since it is much less than the thermal and shot noise ΔV_{T} - for details, see Eq. 8 of Ref. 23.

Finally, at our set of parameters we could use the following simplified formula for gate delay:

$$\tau_0 \approx \ln(2I) \times (C_{\text{wire}}R_{\text{ON}}/D) \times (V_{\text{in}}/V_{\text{DD}}), \quad (4)$$

where I is the circuit fan-in [23].

4.4 Results

The largest value of the average nanodevice utilization factors among all circuits from Toronto 20 benchmark set has turned out to be about 1.5 nanodevices per cell. Plugging N_{cell} into Eq. (2), we find that $R_{\text{ON}} = 21$ M Ω and the ON resistance of a single crosspoint is $R_{\text{ON}}/D = 260$ K Ω . These values justify the simplifications described in the previous subsections.

Since most of the circuits benefited from mapping on large fan-in NOR gates we have chosen the maximum value allowed by T-VPack, $I = 7$. According to Eq. (4) the delay of a 7-input NOR gate turns out to be about 0.8 ns. The full delay of the considered circuits is calculated assuming the absolutely worst case scenario when all the gates on the critical path, which is calculated after mapping and routing step, are maximum fan-in gates. This assumption evidently gives the unrealistically large delays, so that the speed performance results presented below may be only considered as a conservative lower bound. (Our plans are to carry out more realistic speed calculations in near future.)

Table 1 summarizes the performance results for the Toronto 20 benchmark circuit set. Note that in contrast with some earlier work, the results for different circuits were obtained for the CMOL FPGA fabric with exactly the same operating conditions and physical structure, thus enabling a fair comparison with CMOS FPGA.

5. DISCUSSION

In order to compare our results with purely CMOS FPGA, the same benchmark circuits were synthesized into cluster-based island-type logic block architecture [5]. This was done with original T-VPack and VPR tools using the architectural parameters resulting in optimal area-delay product, i.e. cluster size of 4, 4-input LUTs [3] and VPR default architecture file (4x4lut-sanitized.arch). More specifically, we first found the worst case segment width required to route every circuit successfully, which has turned out to be 70 for pdc.blif circuit. Then, using an architecture with such segment width we have mapped and routed all circuits, and then extracted their delay and area (for the optimistic case

⁴The latter assumption made in Ref. 23 has led to some underestimate of noise effects and hence of the CMOL FPGA circuit delay.

of buffer sharing). Assuming the $1/s$ scaling for the delay and assuming the area of the minimum-width transistor to be $25(F_{\text{CMOS}})^2$, we have got the results shown in the left part of Table 1.

Clearly, CMOL FPGA circuits are much smaller in area than the purely CMOS FPGA circuits with the same CMOS design rules. The area of benchmark circuits for CMOL FPGA also favorably compares with that implemented using the nanoPLA concept [6], taking into account the fact that the latter results had been calculated assuming a smaller nanowire half-pitch $F_{\text{nano}} = 2.5$ nm.⁵

It is safe to suggest that the improvement in area would be even larger if one used CMOS FPGA for much larger circuits, because the area of CMOS FPGA is always determined by the worst case routing requirement. On the other hand, a distinctive feature of CMOL FPGA suggested in this work is that the same resources, CMOS cells, are used to perform both logic and interconnect functions. Using the proposed CAD flow, the resources can be allocated flexibly according to the specific logic to routing ratio of the circuit. For example, in order to synthesize the relatively large pdc.blif circuit, only about 15% of the cells have been allocated for logic operation, while this number is about 60% for the smaller dsip.blif (Table 1).

Concerning speed performance, in contrast with our initial results obtained for simple circuits manually mapped on CMOL FPGA fabric [23], the delays calculated in this work for all benchmark circuits are larger than those of their CMOS FPGA counterparts. This is partly due to large connectivity domains (resulting in large capacitance of nanowire fragments) needed for successful routing by our relatively inefficient automated mapping tools. Moreover, as has been noted above, an accurate extraction of the critical path delay may reduce the actual delay value very significantly. For example, the delay of the inverter (i.e. 1-input NOR) is 3.8 times smaller than that of a 7-input NOR gate.

Another evident resource for result improvement is the optimization of F_{nano} and V_{DD} . Next, optimization of the maximum fan-in for each circuit may also give substantial improvements. For example, the area of the s298.blif circuit would be twice smaller, and its pre-mapped depth by 30% lower, if the maximum fan-in was 16 (rather than 7). Finally, our routing algorithm may be evidently improved further.

While we have not performed a defect tolerance analysis in this work, the fact that the NOR gate locations inside the tiles are not fixed gives the gate placement freedom similar to that used in our first work to ensure high defect tolerance. In this analogy, the linear size of the tile has to be compared with the difference $(r - r')$ [23]. In Ref. 23 we have shown that the difference $r - r' = 2$ allows to reach defect tolerance above 20% (for simple defects similar to “stuck-on-open” faults). Since in our current architecture the linear size of the tile is larger (4), we may expect the defect tolerance of these circuits to be even higher. A verification of this expectation is one of our next goals.

⁵Also, the nanoPLA results might be different if all circuits were mapped on a hardware fabric of fixed geometry, as we have done for CMOL architecture.

Circuit	CMOS FPGA ($F_{\text{CMOS}} = 45 \text{ nm}$)					CMOL FPGA ($F_{\text{CMOS}} = 45 \text{ nm}$, $F_{\text{nano}} = 4.5 \text{ nm}$, max fan-in = 7)							Comparison	
	Depth	LUTs	Array size (clusters)	Area (μm^2)	Delay (ns)	Depth	CMOS cells	Array size (clusters)	N	Nano-devices	Area (μm^2)	Delay (ns)	$A_{\text{CMOS}}/A_{\text{CMOL}}$	$A_{\text{nanoPLA}}/A_{\text{CMOL}}$
alu4	7	1274	19x19	137700	5.1	23	1854	22x22	5	9788	1004	18.4	137	0.28
apex2	8	1602	21x21	166050	6.0	26	1928	21x21	6	11365	914	20.7	182	3.09
apex4	6	1147	34x34	414619	5.5	19	1176	18x18	6	7781	672	15.2	617	0.58
bigkey	3	1810	22x22	193388	3.1	20	2065	20x20	6	10207	829	16.0	233	1.82
clma	16	6779	42x42	623194	13.1	75	7585	67x67	2	48746	9308	59.9	67	1.74
des	6	1263	19x19	148331	4.2	28	2321	23x23	6	12610	1097	22.3	135	3.21
diffeq	14	987	16x16	100238	6.0	73	2004	24x24	6	10799	1194	58.3	84	2.27
dsip	3	1362	19x19	148331	3.2	26	1615	20x20	7	9905	829	20.7	179	1.63
elliptic	18	2142	24x24	213638	8.6	81	4799	47x47	4	25415	4581	64.6	47	1.63
ex1010	8	4050	33x33	391331	9.0	43	2986	41x41	3	28746	3486	34.3	112	0.28
ex5p	7	950	16x16	100238	5.1	27	902	20x20	4	6875	829	21.5	121	0.19
frisc	23	2320	25x25	230850	11.3	114	4715	45x45	4	25869	4199	91.0	55	2.64
misex3	7	1178	18x18	124538	5.3	24	1397	22x22	4	9211	1004	19.2	124	0.56
pdic	9	3901	32x32	369056	9.6	54	4752	49x49	2	14841	4979	43.1	74	0.15
s298	15	1682	21x21	166050	10.7	45	1030	20x20	4	10161	829	35.9	200	1.33
s38417	11	4773	36x36	462713	7.3	52	8289	67x67	3	53156	9308	41.5	50	1.24
s38584	9	4422	35x35	438413	4.8	64	6502	69x69	3	50275	9872	51.1	44	-
seq	7	1427	20x20	151369	5.4	23	1832	25x25	4	11027	1296	18.4	117	1.15
spla	8	3331	30x30	326025	7.3	40	4240	38x38	3	24808	2994	31.9	109	0.12
tseng	13	781	14x14	78469	6.3	75	1866	24x24	6	4918	1194	59.9	66	2.48

Figure 7: Performance results for Toronto 20 Benchmark Set

To summarize, we believe that even the preliminary results presented in this report show a possible dramatic impact of the FPGA circuit transfer from CMOS to CMOL technology.

6. ACKNOWLEDGMENTS

The authors would like to thank Alan Mishchenko and Deming Chen for providing a pre-optimized benchmark set, as well as Jacob Barhen, Andre DeHon, Dan Hammerstrom, Ramesh Karri, Phil Kuekes, Alex Orailoglu, Greg Snider, Mircea Stan, and R. Stan Williams for valuable discussions. This work was supported in part by AFOSR and ARDA.

7. REFERENCES

- [1] *FPGA place-and-route challenge*. available online at <http://www.eecg.toronto.edu/vaughn/challenge/challenge.html>
- [2] *International Technology Roadmap for Semiconductors. 2003 Edition, 2004 Update*. available online at <http://public.itrs.net/>.
- [3] E. Ahmed and J. Rose. The effect of lut and cluster size on deep-submicron fpga performance and density. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 12(3):288, 2004.
- [4] V. Betz and J. Rose. Vpr: A new packing, placement and routing tool for fpga research. In *FPL '97: Proceedings of the 7th International Workshop on Field-Programmable Logic and Applications*, pages 213–222, London, UK, 1997. Springer-Verlag.
- [5] V. Betz, J. Rose, and A. Marquardt. *Architecture and CAD for deep-submicron FPGAs*. Kluwer international series in engineering and computer science ; SECS 497. Kluwer Academic, Boston ; London, 1999.
- [6] A. DeHon. Design of programmable interconnect for sublithographic programmable logic arrays. In *International Symposium on Field-Programmable Gate Arrays*, pages 127–137, Monterey, CA, 2005. Association for Computing Machinery.
- [7] A. DeHon and K. Likharev. Hybrid cmos/nanoelectronics digital circuits: Devices, architectures, and design automation. *accepted to IEEE Transactions on Computer Aided Design*, 2005.
- [8] S. Fölling, Ö. Türel, and K. K. Likharev. Single-electron latching switches as nanoscale synapses. In *International Joint Conference on Neural Networks*, pages 216–221, Mount Royal, NY, 2001. Int. Neural Network Soc.
- [9] D. J. Frank, R. H. Dennard, E. Nowak, P. M. Solomon, Y. Taur, and H. S. P. Wong. Device scaling limits of Si MOSFETs and their application dependencies. *Proceedings of the IEEE*, 89(3):259–288, 2001.

- [10] S. C. Goldstein and M. Budiu. NanoFabrics: Spatial computing using molecular electronics. In *Proceedings of the 28th International Symposium on Computer Architecture 2001*, 2001.
- [11] J. R. Heath and M. A. Ratner. Molecular electronics. *Phys. Today*, 56(5):43, 2003.
- [12] S. Heo, R. Krashinsky, and K. Asanović. Activity-sensitive flip-flop and latch selection for reduced energy. In *19th ARVLSI*, Salt Lake City, UT, March 2001.
- [13] P. J. Kuekes, D. R. Stewart, and R. S. Williams. The crossbar latch: Logic value storage, restoration, and inversion in crossbar circuits. *J. Appl. Phys.*, 97(3):034301, 2005.
- [14] K. Likharev, A. Mayr, I. Muckra, and O. Türel. Crossnets - high-performance neuromorphic architectures for cmol circuits. *Ann. NY Acad. Sci.*, 1006:146, 2003.
- [15] K. K. Likharev. Electronics below 10 nm. In *Nano and Giga Challenges in Microelectronics*, pages 27–68. Elsevier, Amsterdam, 2003.
- [16] K. K. Likharev and D. B. Strukov. CMOL: Devices, circuits, and architectures. In G. Cuniberti, G. Fagas, and K. Richter, editors, *Introducing Molecular Electronics*. Springer, Berlin, 2005. to be published as Chapter 16.
- [17] C. Mead and L. Conway. *Introduction to VLSI systems*. Addison-Wesley 1980, Reading, Mass. ; London, 1980.
- [18] J. M. Rabaey, A. P. Chandrakasan, and B. Nikolic. *Digital integrated circuits : A design perspective*. Pearson Education, Upper Saddle River, NJ, 2nd edition, 2003.
- [19] J. R. Reimers, C. A. Picconatto, J. C. Ellenbogen, and R. Shanshidar, editors. *Molecular Electronics III*, volume 1006 of *Annals of the New York Academy of Sciences*. 2003.
- [20] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. Sangiovanni-Vincentelli. SIS: A system for sequential circuit synthesis. Technical report, 1992.
- [21] G. Snider, P. Kuekes, and R. S. Williams. Cmos-like logic in defective, nanoscale crossbars. *Nanotechnology*, 15(8):881, 2004.
- [22] M. R. Stan, P. D. Franzon, S. C. Goldstein, J. C. Lach, and M. M. Ziegler. Molecular electronics: From devices and interconnect to circuits and architecture. *Proceedings of the IEEE*, 91(11):1940–1957, 2003.
- [23] D. B. Strukov and K. K. Likharev. CMOL FPGA: A cell-based, reconfigurable architecture for hybrid digital circuits using two-terminal nanodevices. *Nanotechnology*, 16:888–900, 2005.
- [24] D. B. Strukov and K. K. Likharev. Prospects for terabit-scale nanoelectronic memories. *Nanotechnology*, 16:137, 2005.
- [25] J. Tour. *Molecular Electronics*. World Scientific, Singapore, 2003.
- [26] O. Türel, J. H. Lee, X. L. Ma, and K. K. Likharev. Neuromorphic architectures for nanoelectronic circuits. *Int. J. Circ. Theory App.*, 32(5):277, 2004.