Graphene-PLA (GPLA): A compact and ultra-low power logic array architecture

Graphene-PLA (GPLA): a Compact and Ultra-Low Power Logic Array Architecture

Valerio Tenace, Andrea Calimera, Enrico Macii, Massimo Poncino
Politecnico di Torino, 10129, Torino, Italy
andrea.calimera@polito.it

March 3, 2017

Abstract

The key characteristics of the next generation of ICs for wearable applications include high integration density, small area, low power consumption, high energy-efficiency, reliability and enhanced mechanical properties like stretchability and transparency. The proper mix of new materials and novel integration strategies is the enabling factor to achieve those design specifications.
Moving toward this goal, we introduce a graphene-based regular logic-array structure for energy efficient digital computing. It consists of graphene p-n junctions arranged into a regular mesh. The obtained structure resembles that of Programmable Logic Arrays (PLAs), hence the name Graphene-PLAs (GPLAs); the high expressive power of graphene p-n junctions and their resistive nature enables the implementation of ultra-low power adiabatic logic circuits.

1 Introduction

During the last decade, graphene, a two-dimensional allotrope of carbon, has emerged as one of the most appealing option for silicon replacement in the consumer electronics industry [1]. Mechanical strength and flexibility, combined with higher carrier mobility w.r.t. silicon, make graphene a perfect material for the implementation of wearable devices and sensing applications [2]. Nevertheless, pristine graphene is a zero band-gap material, i.e., valence and conductance bands touch each other near the Dirac points; this avoids graphene to efficiently implement the OFF-state.

Most of the worldwide research projects are pushing efforts to find practical, low-cost methods to open a band-gap in the material equal, or at least close, to that of silicon. Unfortunately, today's available solutions, e.g., patterning [3], chemical doping [4] and/or combination with other materials [5], increase the level of disorder of graphene with rather huge impact on the resulting electrical characteristics. Hence, the need of alternative techniques to adapt graphene for digital applications. Other solutions face the problem from a different perspective, that is, identify alternative techniques that better exploit the intrinsic properties of graphene rather than trying to modify them.

Electrostatic doping [6] falls in the latter category. External electrical fields applied through metal split gates allow a fine-tuning of the Fermi Energy level across the graphene sheet resulting into an equivalent p-type or n-type doping. Face to face regions with opposite doping profiles form an equivalent p-n junction [7], the key component behind any electronic circuit.

A p-n junction is a four-terminal device with two *control* pins, implementing the electrostatic doping, one *input* pin to which the input evaluation signal is fed, and one *output* pin, that eventually collects the input evaluation signal. When control signals have same polarity, the junction is in the ON-state, i.e., a low-

resistive state which allows the input evaluation signal to propagate through the junction and reach the output pin. With opposite polarities, the junction is in the OFF-state, i.e., high impedance state which avoids the input signal to propagate.

This behavior resembles that of a MOS transmission gate with an enhanced logic XNOR functionality; hence the name of *Pass-XNOR* gate. Such devices can be used as a new logic primitive to compactly implement logic circuits that are potentially faster and more power-efficient than silicon counterparts. Since graphene p-n junctions behave as voltage-controlled resistors, rather than ideal switches, when integrated in a CMOS-like strategy, they imply excessive static power consumption. This gave the basis for the introduction of new dynamic design styles that can fruitfully exploit the higher expressive power of graphene p-n junctions, e.g., the PXL style proposed in [8]. How to place and organize this new class of circuits has not been addressed yet.

Recent advances in wearable technologies, especially for medical applications [9, 10], have shown that circuit built upon a regular and programmable structure represent a more suited integration strategy. Our intuition is that the physical geometries and the layout topology of a graphene p-n junction well fit the regular organization of standard logic arrays and, in particular, *Programmable Logic Arrays (PLAs)* used for MOS technologies. We therefore introduce the *Graphene PLA* (GPLA), a PLA-like architecture which integrates graphene p-n junctions for combinational Boolean logic functions.

The main contributions of this paper can be summarized as follows: *(i)* we describe a design methodology aimed at the representation of generic Boolean functions by means of Pass-XNOR gates arranged in a GPLA structure, *(ii)* an algorithmic approach that allows to reduce area occupation (52% less devices) and improve performance/power figures (3.5X lower power-dealy product) over traditional silicon-based PLA structures.

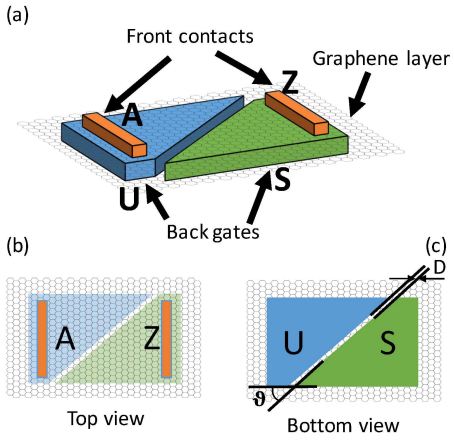


Figure 1: Graphene p-n junction, (a) 3-D view, (b) top view, and (c) bottom view.

2 Background

2.1 Graphene p-n junction

Figure 1-(a) shows an abstract 3-D view of a graphene p-n junction. It consists of a graphene sheet on top of which two metal-to-graphene contacts, A and Z, serve as signal input and output respectively, and a thick layer of oxide that isolates the two back-gates, S and U, from the graphene itself.
Exploiting the principle of the electrostatic doping, voltage potentials on terminals S and U work as a control knob to tune the Fermi Energy (E_F) of the overlapping graphene regions [11]; a negative voltage shifts down E_F in the valence band leading to p-type doping, whereas a positive voltage shifts E_F up in the conductance band leading to n-type doping. When symmetric voltages are concurrently applied on S and U terminals, i.e., $V(S) = +V$ and $V(U) = -V$, the device implements the p-n junction. As described in [11], under such configuration, carriers transmission from the p-region towards the n-region is subject to the transmission probability $T(\theta)$:

$$T(\theta) = \cos^2(\theta) \mathrm{e}^{-\pi k D \sin^2(\theta)} \quad (1)$$

where θ is the angle between the electron's wave vector $\vec{k}$ and the normal of the junction (45 ° as imposed by the triangular shape of the back-gates), and D is the width of the metal gap between the back-gates,

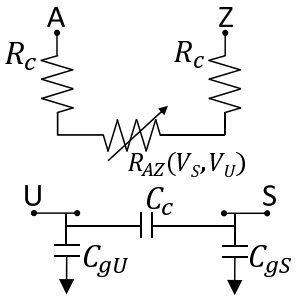


Figure 2: Electrical model of a graphene p-n junction.

assumed to be $18nm$, as described in Figure 1-c. It is worth noticing that $T(\theta) = 1$, i.e., 100% of carrier transmitted, when voltages applied at the back-gates are concordant (n-n or p-p doping configurations). Concerning physical dimensions, the equivalent area occupation of a graphene p-n junction is $0.191\mu m^2$ [7].

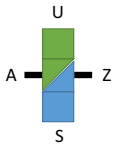
2.2 p-n junction electrical model

Figure 2 shows the electrical model of the p-n junction. There are two R_C resistors connected to pins A and Z representing the parasitic resistance of metal-to-graphene contacts. The resistor R_{AZ} models the resistive path across the graphene layer between the input A and the output Z; its analytical expression is given as $R_{AZ} = \frac{R_0}{N_{ch}T(\theta)}$, where $R_0 = \frac{h}{4q^2}$ is the quantum resistance per propagation mode, N_{ch} is the number of excited propagation modes, and $T(\theta)$ is the transmission probability described by (1). As reported in [7], values of R_{AZ} ranges from $R_{ON} = 300\Omega$, (under n-n or p-p configurations), to $R_{OFF} = 10^7\Omega$ (p-n or n-p configurations). The model also integrates the coupling capacitance C_C between the two metal split gates, and two lumped capacitances connected to the back-gates S and U, namely, C_{gS} and C_{gU} respectively, which consist of the series of the oxide capacitance and the quantum capacitance of the graphene sheet[1].

[1]For a more detailed discussion on the p-n junction and its electrical model, interested readers can refer to [7].

2.3 Pass-XNOR Gate

A Pass-XNOR gate (PXG), depicted in the leftmost picture of Figure 3, simply consists of a p-n junction where the back-gates U and S are fed with digital control signals, while the front contacts A and Z work as source and drain of a stimulus ramp pulse (the *evaluation signal*) used to evaluate the logic function. As shown in the table of Figure 3, when U and S have same logic value, the equivalent in-to-out front resistance R_{AZ} is set to R_{ON} allowing the input pulse on A to pass through the junction reaching Z; this represents the '1'-logic at the output. Opposite logic values, on the contrary, set the in-to-out front resistance R_{AZ} to R_{OFF} ($\gg R_{ON}$), forcing Z in a high-impedance state; this represents the '0'-logic at the output.



A	U	S	R_{AZ}	Z
Pulse	0	0	R_{ON}	A
V_{dd}	0	1	R_{OFF}	Hi-Z
	1	0	R_{OFF}	Hi-Z
0 V	1	1	R_{ON}	A

Figure 3: Logic symbol and functional behavior of the Pass-XNOR logic gate.

3 Graphene logic arrays

3.1 Logic computation through PXGs

Pass-XNOR gates show a higher expressive power if compared to CMOS counterparts, i.e., they require a smaller number of devices to implement XNOR/XOR-dominated logic functions [8]. Unfortunately, the XNOR operator ($\neg\oplus$) *per sé* is not functional complete, therefore other logic connectives are needed. It is possible to pattern multiple PXGs in series in order to obtain a product-of-XNOR, i.e., an AND operation ($\wedge$) between XNOR terms. In other words, since information is propagated by means of the input evaluation signal, the latter is allowed to propagate throughout all the PXG gates *iff* all PXG devices are in the ON-state. On the other hand, multiple PXGs in parallel implement a sum-of-XNOR, i.e., an OR operation ($\vee$) between XNOR terms. This

means that the XNOR is the primitive logic function, whereas AND and OR connectives are implemented by means of different circuit topologies, series and parallel connections, respectively. Such a simple, yet effective, circuit organization allows to fruitfully exploit the expressive power of PXGs to implement any complex Boolean function.
As an example, let us assume a Boolean function defined by:

$$f = ((x_1 \neg\oplus x_2) \wedge x_3) \vee ((x_3 \neg\oplus x_4) \wedge (\neg x_5)) \quad (2)$$

Figure 4 depicts the equivalent PXG-based implementation. Each product term is defined over a single physical row of PXG devices (*AND-connections*); the overall function f is achieved by parallel connections of those product terms (*OR-connections*). The regularity of the structure is guaranteed by a particular property of PXG devices, that is, the possibility to easily implement *identity* and *complement* ($\neg$) functions by means of proper control-signal patterns. For our example, variable x_3 is paired with 1-logic to implement the corresponding identity operation, whereas x_5 is paired to a 0-logic signal as to obtain its complement $\neg x_5$.

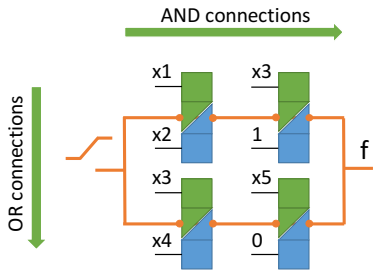


Figure 4: Graphene-based regular strcuture

As one can intuitively understand, graphene PXG gates can be efficiently packed as to form regular, two-dimensional meshes very similar to MOS Programmable-Logic-Arrays.
Logic computing through the proposed Graphene-PLA (GPLA) consists of two distinct phases: the *configuration phase* and the *evaluation phase*. In the configuration phase the primary logic inputs, i.e., the literals composing the logic function, are fed to the back-gates of the p-n junctions. At the end of this phase the doping profile of each and every PXG device is fixed and the resistive paths of the network set up. In the evaluation phase, the input ramp pulse is injected in the network through the front input, i.e., the root of the network, and eventually propagated to the front output. A pulse detected at the front output evaluates the implemented function as TRUE.

3.2 From PLAs to GPLAs

The Programmable Logic Array (PLA) is a well-known implementation style for Boolean logic functions described in the form of Sum-of-Products (SOP). A PLA is made up of two main components: the *AND plane*, where primary inputs are properly connected in order to generate the product terms of the function, and an *OR plane* that collects the product terms for their sum.
Let us consider the single output Boolean function $f(x_1, x_2, x_3)$ described by the equation (3):

$$\begin{aligned} f(x_1, x_2, x_3) = {} & \neg x_1 \neg x_2 \neg x_3 \\ & \vee \neg x_1 x_2 x_3 \\ & \vee x_1 \neg x_2 x_3 \\ & \vee x_1 x_2 \neg x_3 \\ & = \pi_1 \vee \pi_2 \vee \pi_3 \vee \pi_4 \end{aligned} \quad (3)$$

The resulting PLA is depicted in Figure 5. As can be seen, the AND plane (left side) implements all the four product terms π_k by means of physical connections between primary inputs. The latter can be obtained by connecting a MOSFET transistor, typically an n-type MOSFET, through a fuse/anti-fuse mechanism as to create/drop connections. The same happens in the OR plane (right side), where selected product terms are fused/anti-fused to the primary output they belong to. Neglecting the pre-charge transistors, the total number of devices employed for AND/OR connections counts of 16 transistors (black dots in Figure 5).
The PLA structure is well suited for the binding of graphene PXG-based circuits. It is straightforward to map AND/OR connections among PXGs using a

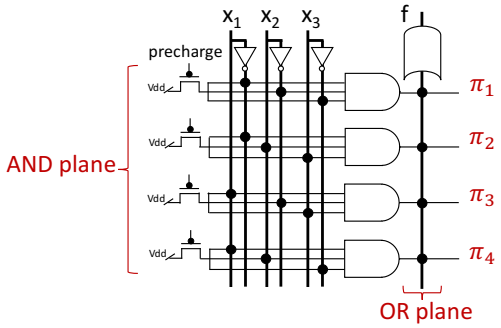


Figure 5: PLA structure of function (3)

PLA-like style. Moreover, while for standard silicon PLAs the physical primitives are simple switch transistors, with GPLAs those primitives work as XNOR operators thereby enabling more compact circuit implementations.

It is worth noticing that, while for PLAs standard SOP-based synthesis methodologies do apply, GPLAs require a different decomposition method for representing Boolean logic functions in terms of Sum-of-XNOR-Products. To this extent, we resort to the PXL-expansion paradigm described in [12].

Let us consider the same example described in (3); the function f can be XNOR-decomposed as described in (4), where each π_k^{gpla} represent the k-th product term:

$$\begin{aligned} f(x_1,x_2,x_3) = &(x_1 \neg\oplus x_2) \wedge (x_2 \neg\oplus x_3) \wedge (\neg x_3 \neg\oplus 1) \\ &\vee (\neg x_1 \neg\oplus x_2) \wedge (x_2 \neg\oplus x_3) \wedge (x_3 \neg\oplus 1) \\ &\vee (\neg x_1 \neg\oplus x_2) \wedge (\neg x_2 \neg\oplus x_3) \wedge (x_3 \neg\oplus 1) \\ &\vee (x_1 \neg\oplus x_2) \wedge (\neg x_2 \neg\oplus x_3) \wedge (\neg x_3 \neg\oplus 1) \\ &= \pi_1^{gpla} \vee \pi_2^{gpla} \vee \pi_3^{gpla} \vee \pi_4^{gpla} \end{aligned} \tag{4}$$

The equivalent GPLA structure is depicted in Figure 6-(a). The AND plane is composed of a chain of graphene PXG devices with control signals connected to the primary inputs; the OR plane is obtained by connecting the product terms to the output rail. Notice that logic signals `1-logic` and `0-logic` are needed for the implementation of identity and complement operators. The resulting GPLA counts of 12 PXG devices, achieving a 25% device reduction w.r.t. the classical PLA structure. It is also worth noticing that PXGs are smaller than MOS devices.

Moreover, due to the resistive nature of PXGs, the *adiabatic charging principle* [8] holds for GPLAs, that is, given a slow-rising ramp as the input evaluation signal, the network switches adiabatically, i.e., at zero power consumption (at least ideally).

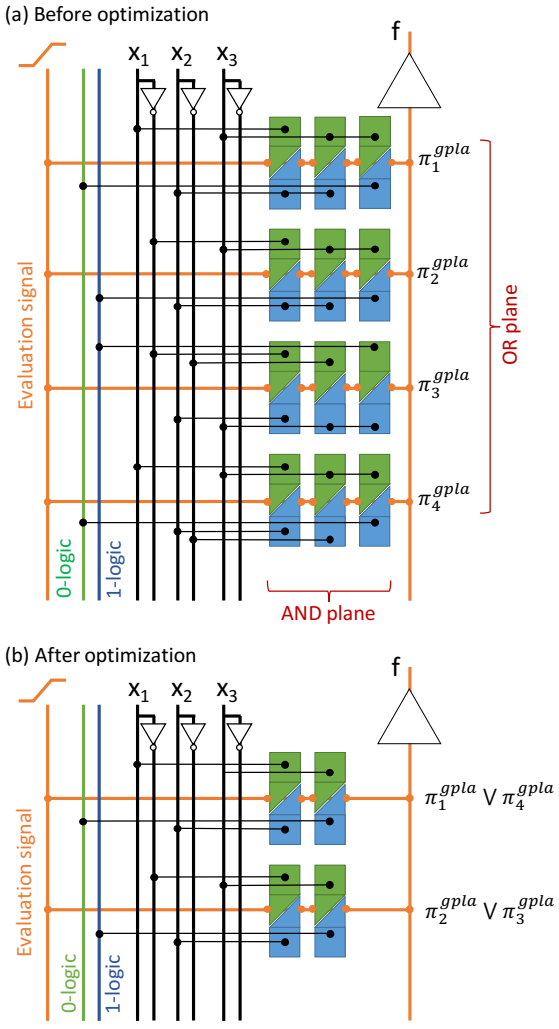


Figure 6: GPLA structure of function (4) w/o optimization (a) and w/ optimization (b)

3.3 GPLA optimization

Though GPLAs, by themselves, are more compact that standard SOP-based MOS implementations, it is still possible to apply optimization rules that sensible reduce their complexity.

Let us first explain this by the same example given in (4). The terms π_1^{gpla} and π_4^{gpla} can be grouped together, with an equivalent expression described as:

$$((x_1 \neg\oplus x_2) \wedge (\neg x_3 \neg\oplus 1)) \wedge ((x_2 \neg\oplus x_3) \vee (\neg x_2 \neg\oplus x_3)) \tag{5}$$

Since the term $(x_2 \neg\oplus x_3) \vee (\neg x_2 \neg\oplus x_3)$ results in a tautology, we can say that both π_1^{gpla} and π_4^{gpla} can be represented with a single product term:

$$\pi_1^{gpla} = \pi_4^{gpla} = (x_1 \neg\oplus x_2) \wedge (\neg x_3 \neg\oplus 1) \tag{6}$$

The same rule holds between π_2^{gpla} and π_3^{gpla}.

The original Boolean function (4) can be therefore represented as:

$$\begin{aligned} f(x_1, x_2, x_3) = &(x_1 \neg\oplus x_2) \wedge (\neg x_3 \neg\oplus 1) \\ &\vee (\neg x_1 \neg\oplus x_2) \wedge (x_3 \neg\oplus 1) \end{aligned} \tag{7}$$

Equation (7) demonstrates that, with few Boolean optimizations, the total number of product terms can be reduced by 50%. The optimized GPLA structure is depicted in Figure 6-(b).

As a rule of thumb, we can state that this *Tautology-based elimination* is feasible only between product terms with overlapping support set. Using a more formal description: *given a Boolean function F, defined over a support set S, there exist at least two generic product terms $\pi_p^{gpla}(S_s)$ and $\pi_q^{gpla}(S_s)$, where $\pi_p^{gpla}, \pi_q^{gpla} \in F$ and $S_s \subset S$, such that grouped factors $\phi_p \in \pi_p^{gpla}(S_s)$ and $\phi_q \in \pi_q^{gpla}(S_s)$ are subject to the $\phi_p \vee \phi_q = 1$ equality.*

Algorithm 1 reports the pseudo-code of the *Tautology-based elimination* routine we implemented in C language. As main input parameter, it takes an XNOR-decomposed function obtained through the synthesis procedure described in [12].

The algorithm compares each product term $\pi_j \in F$ with any other product term $\pi_h \in F$ that belongs to the same output cone. As soon as two product terms can be grouped, the algorithm checks whether they result in a tautology. If so, the reduction rule applies; both product terms are removed from the original function F and the reduced term is appended to the optimized function OF. Otherwise, product term π_j cannot be optimized, and hence it is included in the final solution as it is.

Algorithm 1: Tautology-based elimination routine

```
   Input: PXL-expanded function F
   Output: Optimized function OF
1  OF = ∅
2  foreach π_j ∈ F do
3  |  foreach π_h ∈ F, where π_j ≠ π_h do
4  |  |  if Group(π_j, π_h) returns a tautology
   |  |   then
5  |  |  |  π_t ← new Π(π_j, π_h)
6  |  |  |  OF.append(π_t)
7  |  |  |  F.remove(π_j) F.remove(π_h)
8  |  |  else
9  |  |  |  OF.append(π_j)
10 |  |  end
11 |  end
12 end
```

4 Experimental results

4.1 GPLA simulation models

4.1.1 Delay Modeling

The total delay D_p of a GPLA logic circuit is estimated as the sum of delays due to the configuration phase D_{conf} and the evaluation phase D_{eval}, namely, $D_p = D_{conf} + D_{eval}$; D_{conf} is the time primary logic inputs take to charge the parasitic capacitances at the back-gates of the graphene PXG device, whereas D_{eval} is the propagation delay of the input pulse through the front resistive paths of the network.

In this work, we make use of SPICE simulations for accurate delay estimation, but more compact analytical models based on an Elmore delay approximation are currently under development.

4.1.2 Dynamic Power Modeling

Also the total dynamic power consumption is estimated as the sum of the two contributions during the configuration and the evaluation phase, i.e., $P_{dynamic} = P_{conf} + P_{eval}$, where P_{conf} is due to charging/discharging the input gate capacitance at the back-gates (similar to the input power consumed by CMOS gates) and P_{eval} is the power consumed when charging/discharging the capacitive load at the front output.

A more accurate analysis, however, reveals that a GPLA network simply reduces to an equivalent resistor R_{eq} (calculated as series/parallel connections of R_{ON} and R_{OFF} depending on the back-gates configurations) in series with the load capacitance C_l. Hence the power consumed across the resistor mesh can be calculated as $P_{eval}(t) = R_{eq} i^2_{C_l}(t)$, with i_{C_l} the current finally injected into C_l. The average value can be therefore obtained as:

$$P_{eval} = \frac{1}{T_{rf}} \int_0^{T_{rf}} R_{eq} i^2_{C_l}(t) dt = \frac{R_{eq}}{T^2_{rf}} C_l^2 V_{dd}^2 \qquad (8)$$

where T_{rf} is the rise/fall output transition time, and $i_{C_l}(t)$ is the current charging C_l. As one can observe, the slower the ramp signal, i.e., larger T_{rf}, the smaller the amount of power consumed across the resistors. When T_{rf} is large enough, the entire charging phase completes at zero-power, namely, adiabatically [13]. As for delay estimation, in this work we use accurate SPICE simulations to estimate R_{eq} and $i_{C_l}(t)$.

4.1.3 Static Power Modeling

Concerning the static power consumption, it is worth emphasizing that during the configuration phase and the idle periods, the front input ramp signal is quiescent, i.e., frozen at 0V. This implies a zero potential difference between front input and front output, and thus, *zero static power consumption*, which is the key strength of the proposed logic style. The only contribution to static power is given by the tunneling current at the back-gates, similar to the gate current of MOSFETs. An intuitive model is given as follows:

$$P_{static} = \sum_{i}^{2N} I_g \qquad (9)$$

where N is the number of PXG devices, each of them with two back-gates, and I_g is the tunneling current through a single back-gate.

4.2 Results

This section has a twofold objective: *(i)* demonstrate that the proposed GPLA architecture compactly implements a wide variety of Boolean functions; and *(ii)* prove that GPLAs can reach ultra-low power regimes thanks to the adiabatic charging principle. In order to give a comparison between GPLAs and traditional silicon PLA architectures we implemented two different design flows, described as follows.

GPLA architecture: the subject of this work. Each benchmark is first processed with the ABC synthesis tool [14] in order to guarantee the optimum SOP representation for each Boolean function. The resulting implicant table is firstly used to obtain the equivalent XNOR-decomposed function [8]. The optimized GPLA description is obtained by applying the *Tautology-based elimination* algorithm. The resulting circuit is then mapped using PXGs devices, as described in Section 3.

PLA architecture: each benchmark is processed with the ABC synthesis tool and mapped into a standard silicon PLA structure, i.e., AND/OR connection done through n-MOS transistors.

The selected benchmarks represent a set of heterogeneous (i.e., not just XOR/XNOR-rich) open-source logic circuits. Simulation results are obtained through accurate SPICE simulations. For graphene PXGs we used the Verilog-A model discussed in Section 2.2.

Table 1 summarizes the collected results. It reports the total number of product terms describing the logic circuits (columns **PT**), as well as the total number of devices deployed, n-type MOSFET transistors for PLA (column **T**) and PXGs (graphene p-n junctions) for the GPLA (column **P-N**). For GPLA the table also shows the total area savings w.r.t. PLA (column **Savings**), and the execution time (column **Time**) of the synthesis procedure.

As first comment, one can note that the proposed GPLA architecture has, on the average, 52% less devices. Best results are obtained for those cir-

	PLA		GPLA			
	PT	T	PT	P-N	Savings (%)	Time (s)
xnor3	4	12	2	4	66.67	4.6e-5
4gt4	11	44	8	29	34.09	6e-5
alu	16	80	12	56	30.00	7e-5
xor5	16	80	8	32	60.00	6.2e-5
sym6	50	300	29	153	49.00	2.97e-3
max64	46	395	42	355	10.13	1.85e-4
9sym	87	522	81	480	8.05	3.31e-4
apex4_z03	72	568	69	541	4.75	2.78e-4
apex4_z01	81	592	78	566	4.39	2.88e-4
life	84	672	63	483	28.13	6.67e-4
table5_z01	74	823	72	805	2.19	3.64e-4
alu4_z07	290	2934	276	2766	5.73	2.83e-3
sym10	837	8370	456	4179	50.07	6.83e-2
parity	32768	524288	16384	245760	53.13	52.35
Total	-	**539680**	-	**256209**	**52.53**	-
Average	**2459.71**	-	**1255.71**	-	-	**3.74**

Table 1: PLA vs. GPLA synthesis results

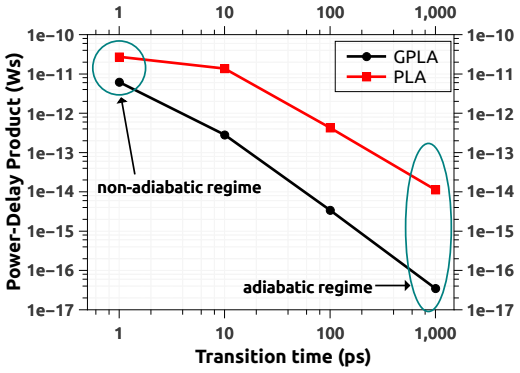


Figure 7: Power-Delay Product (averaged over benchmarks)

cuits where *Tautology-based elimination* does apply, namely, circuits whose internal Boolean functions contain a large number of product terms with overlapping support set, e.g., `sym10` and `parity`. This characteristic is more frequent for well-specified circuits (implicant tables with fewer don't cares). By contrast, benchmarks `table5_z01` and `apex4_z01`, the circuits with smallest savings, just have a handful of product terms with overlapping support set.

Similar conclusions can be obtained by looking at the total number of product terms (columns **PT**). On the average, with the proposed GPLA flow, PT is reduced by a roughly 49% w.r.t. the original minimum-sized SOP representation (from 2460 to 1256). This is due to the higher expressive power of the XNOR-operator, but, above all, thanks to the efficiency of the proposed elimination rule.

Also notice that the GPLA synthesis and optimization flow is quite efficient in terms of CPU usage (below 1 second for most of the benchmarks, just 52.35 seconds for the largest one with more than 32k implicant rows)[2].

Last, but not for importance, Figure 7 depicts the power-delay product (PDP), averaged over all the benchmarks, as function of the transition time T_r of the input evaluation signal.

The internal structure of regular arrays combined with their dynamic nature (pre-charge of the evaluation signal) enable the adiabatic charging principle. PDP reduces by more than 5 orders of magnitude (for GPLA) and 3 orders of magnitude (for PLA) when moving from fast T_r (non-adiabatic regime) to slower T_r (adiabatic regime). This reflects into larger energy-per-operation efficiency, particularly suited for wearable applications.

Moreover, and this is the most important aspect, GPLAs (circle mark) show better PDPs than PLAs (square mark) along the entire range, from a minimum of 0.5X less PDP at $T_r = 1ps$, namely when circuits are outside the adiabatic region, up to a maximum of 3.5X when $T_r = 1ns$, i.e., in the deep-adiabatic region. This result is motivated by the fact that: *(i)* MOSFET-based PLAs show a higher power consumption due to the large amount of simultaneous active/switching devices; *(ii)* MOSFET transistors are active devices, namely, they behave as resistors only in the triode, while p-n junction are pure resistors, more suited for adiabatic operation.

5 Conclusions

In this work we introduced a graphene-based regular array, the GPLA, that efficiently implements a wide class of Boolean logic functions. We demonstrated that, through XNOR-decomposition and Tautology-based elimination, the proposed architecture outperforms classical PLA structure by means of area occupation (52% improvement). Finally, exploiting

[2]For larger benchmarks we collected CPU times in the order of 10 seconds.

the adiabatic charging principle, we also proved that GPLAs have remarkable ultra-low power features.

References

[1] A. K. Geim and K. S. Novoselov, "The rise of graphene," *Nature materials*, vol. 6, no. 3, pp. 183–191, 2007.

[2] J. A. Rogers, T. Someya, and Y. Huang, "Materials and mechanics for stretchable electronics," *Science*, vol. 327, no. 5973, pp. 1603–1607, 2010.

[3] P. Sessi, J. R. Guest, M. Bode, and N. P. Guisinger, "Patterning graphene at the nanometer scale via hydrogen desorption," *Nano letters*, vol. 9, no. 12, pp. 4343–4347, 2009.

[4] H. Liu, Y. Liu, and D. Zhu, "Chemical doping of graphene," *Journal of materials chemistry*, vol. 21, no. 10, pp. 3335–3345, 2011.

[5] H. Wang, L.-F. Cui, Y. Yang, H. Sanchez Casalongue, J. T. Robinson, Y. Liang, Y. Cui, and H. Dai, "Mn3o4- graphene hybrid as a high-capacity anode material for lithium ion batteries," *Journal of the American Chemical Society*, vol. 132, no. 40, pp. 13 978–13 980, 2010.

[6] C. Ahn, A. Bhattacharya, M. Di Ventra, J. Eckstein, C. D. Frisbie, M. Gershenson, A. Goldman, I. Inoue, J. Mannhart, A. J. Millis *et al.*, "Electrostatic modification of novel materials," *Reviews of Modern Physics*, vol. 78, no. 4, p. 1185, 2006.

[7] S. Tanachutiwat, J. U. Lee, W. Wang, and C. Y. Sung, "Reconfigurable multi-function logic based on graphene pn junctions," in *DAC'10: Design Automation Conference*. ACM/IEEE, 2010, pp. 883–888.

[8] V. Tenace, A. Calimera, E. Macii, and M. Poncino, "Pass-xnor logic: A new logic style for pn junction based graphene circuits," in *DATE'14: Design, Automation and Test in Europe Conference and Exhibition*. IEEE, 2014, pp. 1–4.

[9] T. Ahola, P. Korpinen, J. Rakkola, T. Ramo, J. Salminen, and J. Savolainen, "Wearable fpga based wireless sensor platform," in *EMBS'07: International Conference of Engineering in Medicine and Biology Society*. IEEE, 2007, pp. 2288–2291.

[10] A. Armato, E. Nardini, A. Lanata, G. Valenza, C. Mancuso, E. P. Scilingo, and D. De Rossi, "An fpga based arrhythmia recognition system for wearable applications," in *ISDA'09: International Conference on Intelligent Systems Design and Applications*. IEEE, 2009, pp. 660–664.

[11] V. V. Cheianov and V. I. Falko, "Selective transmission of dirac electrons and ballistic magnetoresistance of n- p junctions in graphene," *Physical Review B*, vol. 74, no. 4, p. 041403, 2006.

[12] V. Tenace, A. Calimera, E. Macii, and M. Poncino, "One-pass logic synthesis for graphene-based pass-xnor logic circuits," in *DAC'15: Design Automation Conference*. ACM/IEEE, 2015, pp. 1–6.

[13] S. Miryala, A. Calimera, E. Macii, and M. Poncino, "Ultra low-power computation via graphene-based adiabatic logic gates," in *DSD'14: Euromicro Conference on Digital System Design*. IEEE, 2014, pp. 365–371.

[14] B. L. Synthesis and V. Group, "ABC: A System for Sequential Synthesis and Verification," http://www.eecs.berkeley.edu/~alanmi/abc/, 2014.