Logic Synthesis for Emerging Technologies

Giovanni De Micheli
EPFL
Lausanne, Switzerland
Giovanni.DeMicheli@EPFL.ch

***Abstract*—The unprecedented achievements of electronic systems are due to the combined progress of semiconductor technologies and design automation of integrated circuits. This work surveys new directions of logic synthesis in light of the expectation to exploit both established and emerging technologies in search of faster, smaller and more energy-savvy solutions.**

Keywords—IC, VLSI, EDA, logic synthesis, emerging technologies, nanowires, CNTs, superconducting electronics.

I. Introduction

Design and production of digital circuits have reached unprecedented levels of quality, being fueled by the needs of architectural support and acceleration of *artificial intelligence* and *machine learning* algorithms, the electrification and extended capabilities of vehicles and the ubiquitous presence of terminals for wireless communication, such as phones. These achievements have roots in two pillars: the advancement of nano-electronic technology and the improvement of performance and capabilities of design tools. Thus, the *design technology* - also called *electronic design automation* (EDA) - parallels the *fabrication technology* in the realization of advanced *integrated circuits* (ICs) today, and their synergic relation has been the enabler of the success of the microelectronic industry.

This survey summarizes advances in EDA and their direct applications to established and emerging technologies. It also points the interested reader to the referenced specialized literature.

II. Digital Design

The main workhorse for digital design is CMOS technology, encompassing planar, FinFET, nano-wires and nano-sheets devices [41]. As technology requires logic gates with small fanin, then design consists of formulating the logic function into *logic networks* [8,9] (called also *Boolean chains* [19]) and storage elements (register and memories of various types). In its simplest realization, a logic network can be seen as an interconnection of two-input logic operators or gates. The minimum number of such gates to realize a function is called the *complexity* of the network. Finding a minimum complexity network is a *computationally intractable* problem [19]. An even more relevant (and equally difficult) problem is minimizing the *depth* of a network, i.e., the maximum I/O critical path length. All models and problems described next can be cast as size or depth optimization problems.

Limiting the type of gates is a way of simplifying the problem in the search for effective heuristic solutions. As an example, an AIG or *and-inverter graph*, is a logic network consisting only of AND and INV (inverter) gates. The AIG model is universal, and its simplicity has enabled the design of effective heuristic design tools, such as ABC [10], the most successful open-source logic synthesis tool. Slightly more complex models involve the use of three-input gates as primitives. A thorough study of advantages and disadvantages of using 3-input gates was done for homogeneous networks, i.e., networks with the same gate type [23]. An interesting example among these networks are MIGs, or *majority-inverter graphs*. Amarù developed heuristic methods to optimize such networks that are very competitive with other optimization approaches, even when the final realization is remapped to other library gates [3]. Soeken et al. researched exact methods for MIG optimization [34]. They showed that such methods are practical only for limited-size functions. Nevertheless, they conceived an iterative optimization procedure where circuits are split into bounded-size blocks, such blocks are exactly optimized and re-merged with the others. Such an iteration leads to circuit realizations of competitive size as compared to others. Another model for three input gates leverages the MUX (multiplexer) gate. Under some restrictions on connectivity, multiplexer networks are abstracted well by *binary decision diagrams* (or BDDs) [11]. Whereas the computational model is extremely useful for logic synthesis and verification, direct mapping of such networks into a physical layout leads to “slow” realizations in ASICs, because chaining MUX gates generates connections in series of pass-transistor gates. Nevertheless, MUX networks have been used successfully in FPGA architectures.

Other interesting models include the use of two logic primitives, such as XOR and AND (XAG, *xor-and graph*) as well as XOR and MAJority (XMG, *xor-majority graph*). The first logic model has been object of interest for a long time, as it separates the linear from the non-linear component of a circuit. The *multiplicative complexity* of a network, defined as the number of AND gates [7], plays a role in the security of circuits and in the effectiveness of multi-party computation [42]. Thus, various procedures [37] have been devised to reduce it. Whereas the multiplicative complexity of a network can be directly measured, the multiplicative complexity of a function is the minimum that can be achieved while considering all its realizations. Its computation is an intractable problem, and hard to evaluate except for small (up to 6) inputs [12]. An interesting application of this model to quantum computing design is the following [25]. Given a reversible logic model of an oracle Boolean function, reducing the multiplicative complexity correlates to reducing the number of T-gates in a quantum circuit realization, where typically the T-gate is the most “expensive” primitive to realize.

It is interesting to remark that XOR gates cannot be used alone (even along with inverters) to realize logic functions, because they are linear and therefore they miss the nonlinear component of a digital circuit. As an example, they can be used to realize parity trees but not adders, whose multiplicative complexity is 1. Nevertheless, XOR gates pair well with other gates, such as ANDs (as described before) and majority. One interesting property is *self-duality* of logic gates and functions, i.e., the property that the function is equivalent to its complement with complemented inputs. Both majority gates and odd-input XOR gates are self-dual. The XMG representation is a very convenient data structure for logic optimization in two respects. First, it enables reconfigurations where XORs free-flow through majority gates (i.e., the shift of XOR gates from MUX inputs to outputs or vice versa) and second this model fits well mapping self-dual circuits (or parts thereof) to *controlled-polarity* (also called *ambipolar*) transistors [28].

Finally, it is important to remark that most libraries include gates of different types, and it is convenient to leverage all of them. Typical approaches to logic optimization consist first of restricting the logic network model to enable general transformations within *technology-independent* logic synthesis and then of mapping to library gates (weighted in terms of delay, area, load and fanout capability) in a second *technology-dependent* step. Such a distinction tends to disappear in novel approaches targeting new technologies with gates having specific characteristics. In a similar vein, technology advancement has led to libraries of a small number of cells, as compared to the past, that can be captured better by logic network models and graphs because of their higher uniformity [1,38].

III. Emerging technologies

In this work, we consider only a limited number of technologies, and we present those that benefit from new logic synthesis methods. At the time of this writing, the CMOS FinFET technology is an established technology, as well as CMOS *nanosheets* that evolved from silicon *nanowires* over the last decade [41]. Conversely, the realization of silicon nanowire/sheets (and also FinFETs) with controlled polarity devices is new and of particular interest for logic design. In these technologies [28], the transistors are formed by the metal/silicon Schottky junctions that can be polarized through a vertical external field provided by an additional *polarity gate*. As a result, carriers of one polarity only can transit in the channel, thus forming an N or a P transistor, controlled by a standard *control gate*. The symmetry and interchangeability of N and P transistors allow us to realize fully restoring XOR gates in a compact way [6]. Rai et al. [28] showed that this technology supports well the realization of self-dual functions, among others. Therefore, he developed specific synthesis tools that leverage XOR and majority gates. Similar considerations can be carried over to both *carbon nanotubes* (CNTs) [18] and 2D technologies, such as *tungsten di-selenide* (WSe_2) [29,30,31] and other devices based on *transition-metal di-chalcogenides*. Moreover, the search for data structures and algorithms for designing circuits with controlled-polarity transistors led to methods that perform well also on established CMOS technologies [3].

The quest for alternatives to silicon for high-performance application has led to the study of *superconducting circuits* that operate generally at 4K, where resistive effects disappear and thus interconnect does not penalize circuit performance and power. There are different families of superconducting circuits, that usually leverage the two states of *Josephson Junctions* (JJ) that can be either superconducting or resistive. Likharev and coworkers [22] proposed the *single-flux quantum* (SFQ) technology and its derivatives, that are based on transmitting information as pulses (that correspond to quanta of fluxes). Such a style enables fast JJ switching and thus low-power computation. The quest for very-low power computation is addressed by circuit families, such as *reciprocal quantum logic* (RQL) [17] and *adiabatic quantum flux parametron* (AQFP) [35], that combine adiabatic with superconducting operation. A few chips have been designed whose power consumption gets close to the known bound for computation [4,5]. The price for realizing circuits in superconducting technology is the increased complexity in design. First, all gates need to be clocked, thus yielding a circuit operating in *pipeline* mode, and then the pipeline needs to be *balanced* to insure correct operation. Next, fanout loads have to be connected by the use of signal *splitters*, that in turn may need to be clocked in some circuit families. Moreover, some of these circuit families, such as AQFP, have primitives whose native model is a majority function. As a result, logic synthesis tools based on MIGs and MIG optimizations are particularly effective [14].

In the case of *quantum cellular automata* (QCA) the logic can also be abstracted by MIGs, i.e., by majority and INV gates. In this case, the area cost of the inverter is much higher than the cost of the MAJ gate, thus requiring specific optimization techniques that minimize the inverter count and/or pushes them to the combinational circuit periphery so that they can be subsumed by registers [39].

Recent advances in semiconductor manufacturing have enabled the fabrication of memory elements during the *back end of line* (BEOL) steps. In essence, various types of memory devices can be realized in between metal layers, on top of the standard CMOS transistors. This approach falls into the large class of *in memory computing* paradigms. Some realizations exploit the use of *oxide RAM* (OxRAM) or *phase change* RAM (PCRAM) as a nonvolatile device. Such devices can be configured to act as storage devices or as a MAJ gates. Thus, combinational circuits can be mapped (in parts or as a whole) into an appropriate network of memory elements. This approach can be applied in various forms, ranging from providing additional gates on top of silicon to using an entire memory array to perform computation and storage, thus leveraging advances and compactness of new memory devices.

IV. Methods

Due to the inherent computational complexity and to the need of having effective tools for large-scale networks, we consider first heuristic approaches that improve them through a *sequence of steps*. It is possible to split heuristic logic synthesis and optimization methods into two classes. The former deals with rewriting a network using models based on an *algebra* and on a set of *transformation rules*. The first (historical) example is the use of *polynomial* *division* to factor, decompose, and extract common subexpressions [8,9]. A second example is given by algebraic rewrite rules as proposed for AIGs [10] and MIGs [2]. The latter class entails Boolean specifications including *don't care* conditions. Such conditions arise from the connection of the network under consideration to other logic blocks (*external don't cares*), as well as to the internal structure of the network (*internal don't cares*). In particular, *don't cares* are related to *controllability* and *observability* conditions at network inputs, outputs or internal nodes.

The first (historical) model represented circuits as networks of *sum of product* nodes, that can be optimized through two-level logic minimization (e.g., Espresso). A second (historical) model is the use of *permissible functions* [26] that express the latitude of modifying a part of the network. Later this process was codified as a sequence of changes in the network, each leading to a simpler one, with the constraint that the change in the network must not affect the network behavior at any output or equivalently that the change is contained in the *don't care* set. This check can be done using BDDs, very efficient for equivalence and containment checks [40]. In recent years, *satisfiability* (SAT) computational engines have been used also for this purpose.

Decomposition of logic functions into smaller components is a Boolean method that has been extensively studied in the literature. In particular, *disjoint-support decomposition* (DSD) has shown to be practical in many settings. Recently Chu [13] extended DSD with majority-based decomposition, and applied it to extracting XMGs from truth-table descriptions.

The most general step in Boolean optimization consists of replacing one (or more) nodes by a plurality of nodes. This task is called *substitution* or *re-substitution*. The search for the (partial) replacement nodes is not straightforward, and many strategies have been devised, including using simulation for determining patterns that can act as lampposts to guide substitution [20]. Substitution is still a topic of active research, and can be viewed as the most general Boolean method for optimization.

Exact logic minimization can be cast as solving a SAT problem given a set of constraints for each logic node (or library element) of the network and their interconnection [15, 16, 34]. Various encoding and methods have been proposed, and in general they search for a solution network with an increasing number of nodes realizing the function and satisfying critical-path delays. They typically work for small scale-networks (i.e., for functions with less than 6 inputs). While such an approach is not at all applicable to circuits of typical size in current ICs, they can be used to build optimum libraries of components of a network, that can then be connected using heuristics approaches.

The fanout problem is critical in established and emerging technologies. As devices scale down, their current source/sink capability is limited, and this leads to slow transitions in presence of large fanouts and corresponding large capacitive loads. An exact and a heuristic method for bounding the fanout in technology-independent synthesis, through buffer insertion and gate duplication, is presented in [24], applicable to both CMOS and superconducting technologies. In the latter case, fanout is handled by splitter cells, that can be modeled as buffers when restructuring a network with coarse-grain models of area and delay.

In general, a comprehensive solution is required to optimize superconducting circuits where gates are clocked and circuits operate in a deep pipeline fashion. There are two main cases of interest. In the first one, exemplified by SFQ circuits, buffers and splitters are not clocked. So, the path balancing problem can be handled independently from fanout considerations. Techniques based on *dynamic programming* [27] and *retiming* can be used to distribute optimally buffers and splitters in SFQ circuits. Conversely, for other families of circuits like AQFP, splitters are clocked and contribute to the pipeline delay. Thus, splitter insertion and path balancing are intertwined and should be addressed together. Exact and heuristic solutions have been proposed [21,36]. This problem can be generalized to that of encapsulating combinational logic among registers and I/O structures, and optimizing the latency, throughput and clocking parameters. Few techniques are used for sequential synthesis, mainly based on retiming and its extensions.

One interesting application of sequential logic synthesis is related to using logic in memory. The operation of some non-volatile memory cells (like OxRAMs) is sequential in nature. The state, i.e., the value stored at a given timepoint, is the majority function of the previous state, the value of a terminal and the complement of the value at the other terminal. Thus, a specific compiler can map a combinational logic function into a sequence of majority operations. The PLIM tool [32] does such a conversion, and enables a designer to realize combinational functions in memory.

In this brief review, I have considered the logic synthesis stage of circuit design. In general, the compilation of high-level models into logic circuits, also called high-level synthesis, is fairly independent of the fabrication technology of choice. Conversely, physical design models, algorithms and tools are very technology specific, and require frequent updates. Their availability is an asset or a constraint in the choice of an emerging technology.

V. Conclusions

This survey has shown both the evolution of logic synthesis and its relevance to design in both established and emerging technologies. Despite the availability of commercial and open-source academic tools [10,33], new models and methods for optimizing of digital circuits are still required, as current design tools provide us with solutions that can still be improved to empower us with the ultimate opportunity of leveraging any fabrication technology to achieve competitive products.

VI. References

[1] L. Amarù, P.-E. Gaillardon, S. Mitra and G. De Micheli. New Logic Synthesis As Nanotechnology Enabler, in Proceedings of the IEEE, vol. 103, num. 11, pp. 2168-2195, 2015.

[2] L. Amarù, P.-E. Gaillardon, A. Chattopadhyay and G. De Micheli. A Sound and Complete Axiomatization of Majority-n Logic, in IEEE Transactions on Computers, vol. 65, num. 9, p. 2889 - 2895, 2016

[3] L. Amarù, P.-E. Gaillardon and G. De Micheli. Majority-Inverter Graph: A New Paradigm for Logic Optimization, in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), vol. 35, num. 5, p. 806-819, 2016.

[4] C. Ayala, R. Saito, T. Tanaka, O. Chen, Takeuchi, et al. A semi-custom design methodology and environment for implementing superconductor adiabatic quantum-flux-parametron microprocessors, Superconductor Science and Technology 33, 5 (2020)

[5] C. L. Ayala, T. Tanaka, R. Saito, M. Nozoe, N. Takeuchi and N. Yoshikawa, MANA: A Monolithic Adiabatic iNtegration Architecture Microprocessor using 1.4zJ/op Superconductor Josephson Junction Devices, 2020 IEEE Symposium on VLSI Circuits, 2020, pp. 1-2

[6] S. Bobba and G. De Micheli. Layout Technique for Double-Gate Silicon Nanowire FETs With an Efficient Sea-of-Tiles Architecture, in IEEE Transactions on very Large Scale Integration (VLSI) Systems, vol. 23, num. 10, pp. 2103-2115, 2015.

[7] J. Boyar and R. Peralta, A new combinational logic minimization technique with applications to cryptology, Proc. Int. Symp. Experimental Algorithms, pp.178-189, 2010.

[8] R.Brayton, N.Brenner, G.De Micheli, C.McMulen and R. Otten, The Yorktown Silicon Compiler, Proc. ISCAS, Kyoto, 1985.

[9] R. K. Brayton, R. L. Rudell, A. L. Sangiovanni-Vincentelli, and A. R. Wang, MIS: A multiple-level logic optimization system, IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., vol. 6, no. 6, pp. 1062–1081, 1987.

[10] R. Brayton and A. Mischchennko, ABC:an academic industrial-strength verification tool, in Computer Aided Verification, 22nd International Conference, CAV 2010, 2010, pp. 24–40.

[11] R.Bryant, Graph-based Algorithms for Boolean function manipulation, IEEE Trans. Computers, vol. 35, no. 8, pp. 677–691, Aug. 1986

[12] [Ç.Çalık , M. Sönmez Turan, and R.Peralta, The multiplicative complexity of 6-variable Boolean functions, Cryptogr. Commun. **11**, 93–107 (2019)

[13] Z. Chu, M. Soeken, Y. Xia, L. Wang, G. De Micheli, Advanced Functional Decomposition Using Majority and Its Applications, Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol 39, Issue 8, 2020.

[14] G. De Micheli, The emerging majority: Technology and design for superconducting electronics, IEEE Design and Test, Vol. 38, No. 6, pp. 79-87, December 2021.

[15] N. Eén, Practical SAT—A tutorial on applied satisfiability solving, in Proc. FMCAD, 2007

[16] W. Haaswijk, M. Soeken, A. Mischenko, G. De Micheli, SAT-based Exact Synthesis: Encoding, Topology Families and Parallelism, IEEE Transactions on CAD, Vol.39, No.4, April 2020.

[17] Q. Herr, A. Herr, O. Oberg and A. Ioannidis, Ultra-Low Power Superconducting Logic, Journal of Applied Physics, 109, 2011.

[18] G. Hills, C. Lau, A. Wright,. et al. Modern microprocessor built from complementary carbon nanotube transistors. Nature **572,** 595–602

[19] D. E. Knuth,The Art of Computer Programming, Volume 4A, Addison-Wesley, 2011

[20] S.Y. Lee, A. Mishchenko, H. Riener and G. De Micheli, A Simulation-Guided Paradigm for Logic Synthesis and Verification, IEEE TCAD vol. 41, no. 8, pp. 2573-2586, Aug. 2022.

[21] S.-Y. Lee, H. Riener and G. De Micheli, Beyond Local Optimality of Buffer and Splitter Insertion for AQFP Circuit*s*, DAC 2022

[22] K. Likharev and V. Semenov, RSFQ Logic/Memory Family: A New Josephson-Junction Technology for Sub-terahertz Clock-Frequency Digital Circuits, IEEE Transactions on Applied Superconductivity, Vol. 1, No. 3, March 1991, pp. 3-28.

[23] D. S. Marakkalage, E. Testa, H. Riener, A. Mishchenko, M. Soeken and G. De Micheli, Three-Input Gates for Logic Synthesis, IEEE Transactions on CAD , Vol. 40, No. 10, October 2021, pp. 2184-2188.

[24] D. Marakkalage and G. De Micheli, Fanout-Bounded Logic Synthesis for Emerging Technologies - A Top-Down Approach, Proceedings of DATE, March 2023.

[25] G. Meuli, M. Soeken and G. De Micheli, Xor-And-Inverter Graphs For Quantum Compilation, Nature Partner Journal on Quantum Information, 8, 7, January 2022.

[26] S.Muroga, Y.Kambayashy, H.C.Lai, and J.N.Culliney, The transduction method – design of logic networks based on permissible functions, IEEE Trans. Comp., Vol 38, No. 10, pp. 1404–1424, Oct. 1989.

[27] G. Pasandi and M. Pedram, A dynamic programming-based path balancing technology mapping algorithm targeting area minimization, Int'l Conf. on Computer-Aided Design, 2019. [

[28] S. Rai, A. Tempia Calvino, H. Riener and G. De Micheli, Utilizing XMG-based Synthesis to Preserve Self-Duality for RFET-Based Circuits, IEEE TCAD, Vol. 452, No. 3, March 2023, pp. 914-927.

[29] G. Resta, A. Leondhart, Y. Balaji, S. De Gendt, P.-E. Gaillardon G. De Micheli.Devices and Circuits using Novel 2-Dimensional Materials: a Perspective for Future VLSI Systems, IEEE Transaction on Very Large Scale Integration Systems, vol. 27, num. 7, pp 1486-1503, July 2019.

[30] G.V. Resta, Y. Balaji, D. Lin, I.P. Radu, F. Catthoor, P.-E. Gaillardon, G. De Micheli, Doping-Free Complementary Logic Gates Enabled by Two-Dimensional Polarity-Controllable Transistors, ACS Nano vol. 12, num. 7, p. 7039-7047. 2018.

[31] G. V. Resta, S. Sutar, Y. Balaji, D. Lin and P. Raghavan et al. Polarity control in WSe_2 double-gate transistors, in Scientific Reports, vol. 6, num. 29448, 2016.

[32] M. Soeken, P.-E. Gaillardon, S. Shirinzadeh, R. Drechsler and G. De Micheli. A PLiM Computer for the Internet of Things, in Computer, vol. 50, num. 6, p. 35-40, 2017.

[33] M. Soeken, H. Riener, W. Haaswijk, E. Testa, B. Schmitt, G. Meuli, F. Mozafari and G. De Micheli, The EPFL Synthesis libraries, Arxiv: 1805.05121v2 (2019) and updates.

[34] M. Soeken, L. Amaru, P.-E. Gaillardon and G. De Micheli. Exact Synthesis of Majority-Inverter Graphs and Its Applications, in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 36, num. 11, p. 1842-1855, 2017.

[35] N. Takeuchi, Y. Yamanashi, and N. Yoshikawa, Adiabatic quantum-flux-parametron cell library adopting minimalist design, Journal of Applied Physics 117, 17, 2015.

[36] E. Testa, S-Y. Lee, H. Riener and G. De Micheli, Algebraic and Boolean Optimization Methods for AQFP Superconducting Circuits, ASPDAC, Tokyo, Japan, 2021.

[37] E. Testa, M. Soeken, L. Amaru, G. De Micheli, Reducing the Multiplicative Complexity in Logic Networks for Cryptography and Security Applications, DAC 2019, Las Vegas, USA, June 2019.

[38] E. Testa, M. Soeken, L.G. Amaru, G. De Micheli, Logic Synthesis for Established and Emerging Computing, Proceedings of the IEEE. Vol. 107, No. 1, pp 165-184. 2019.

[39] E. Testa et al., Inverter propagation and fanout constraints for beyond CMOS majority-based technologies, Proc VLSI, pp. 164-169, 2017.

[40] C. Yang and M. Ciesielski, BDS: A BDD-based logic optimization system, IEEE Trans on CAD, Vol. 21, no. 7, pp. 866–876, Jul. 2002.

[41] P. Ye, T. Ernst and M. V. Khare, The last silicon transistor: Nanosheet devices could be the final evolutionary step for Moore's Law, in IEEE Spectrum, vol. 56, no. 8, pp. 30-35, Aug. 2019

[42] M. Yu and G. De Micheli, Generating Lower-Cost Garbled Circuits: Logic Synthesis Can Help, HOST, Santa Clara, 2023.