fiction: Electronic Design Automation for Field-coupled Nanocomputing Circuits

Marcel Walter[1] Robert Wille[2,3] Frank Sill Torres[1,3] Daniel Große[1,3] Rolf Drechsler[1,3]

[1]Group of Computer Architecture, University of Bremen, Germany
[2]Johannes Kepler University Linz, Austria
[3]Cyber Physical Systems, DFKI GmbH, Bremen, Germany
{m_walter, frasillt, grosse, drechsler}@uni-bremen.de, robert.wille@jku.at

https://github.com/marcelwa/fiction (v0.2.1)

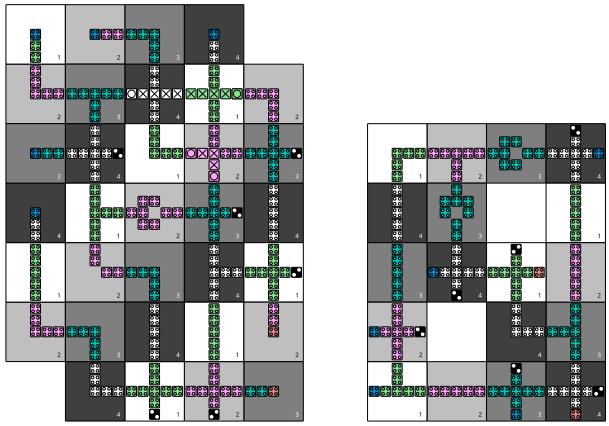
***Abstract*—As a class of emerging post-CMOS technologies, *Field-coupled Nanocomputing* (FCN) promises computation with tremendously low energy dissipation. Even though ground breaking advances in several physical implementations like *Quantum-dot Cellular Automata* (QCA) or *Nanomagnet Logic* (NML) have been made in the last couple of years, design automation for FCN is still in its infancy and often still relies on manual labor. In this research demo proposal, we present an open source framework called *fiction* for physical design and technology mapping of FCN circuits. We furthermore evaluate the effectiveness of implemented algorithms in two case studies. The capabilities of the proposed framework to explore different design criteria, scripting and logging functionalities, and extensibility provide a basis for future research in this domain.**

I. Introduction and Background

Field-coupled Nanocomputing (FCN) [1] is a class of emerging technologies which conduct computations fundamentally different from conventional systems relying e. g. on CMOS. Here, information is represented in terms of the polarity or magnetization of nanoscale cells and can be propagated to adjacent ones using repelling forces of local fields [2], [3]. This results in devices that allow to represent and process binary information without electrical current flow. Consequently, numerous contributions on their physical realization have been made in the past and several of them in the last three to four years, e. g. *molecular Quantum-dot Cellular Automata* (mQCA) [4], *atomic Quantum-dot Cellular Automata* (aQCA) [5], [6], or *Nanomagnet Logic* (NML) [7].

Moreover, this way of representing and processing information is doable with highest processing performance and remarkably low energy dissipation – as confirmed by several theoretical and experimental studies (see e. g. [8], [9], [10]). This makes FCN a promising alternative to conventional integrated circuit technologies. However, no exhaustive automatic design flow is available for FCN technologies so far. Also, due to significant differences to the design rules of CMOS technologies, existing classical methods are not applicable to the FCN domain.

In this research demo proposal, we present *fiction* [11], a C++ framework for the design of FCN circuits which is based on the *EPFL Logic Synthesis Libraries* [12]. With this tool, we especially tackle the physical design steps of FCN like placement, routing, timing, and technology mapping under the domain specific constraints.



(a) `exact -ixbs 2ddwave4` (b) `exact -ps use`

Fig. 1: Two differently layouted variants of c17.v

II. Proposed Design and Experimental Evaluation

The aim of *fiction* is to explore the physical design of FCN technologies and, in doing so, providing a platform for future research in this domain. Therefore, *fiction* comes with out-of-the-box support for different layout paradigms, namely tile-based and cell-based layouts [13], [14], an easily extensible set of clocking schemes, e. g. *2DDWave* [15], *USE* [16], *RES* [17] and *BANCS* [18], as well as irregular clockings, and various cost metrics to evaluate layout quality. Furthermore, even exotic ideas like clock-driven synchronization elements [19] can be exploited. Starting from structured Verilog netlist files, created layouts can be written as physical simulation files for the *QCADesigner* [20] and as SVG graphics. Furthermore, built-in scripting and logging functionalities provide an environment for experimental evaluations.

On top of that, two state-of-the-art physical design algorithms presented in earlier papers have already been implemented in *fiction* to demonstrate its capabilities: a *Satisfiability Modulo Theories* (SMT)-based approach called `exact` [21] and an *Orthogonal Graph Drawing* (OGD)-based technique called `ortho` [22].

TABLE I: Evaluating `exact`

Benchmark			Previous s-o-t-a [25]			Proposed `exact` [21]		
Name	**#Gates**	**I / O**	**Dimension**	**CP**	**t in s**	**Dimension**	**CP**	**t in s**
2:1 MUX	5	3 / 1	4 × 5	5	9	3 × 3	5	< 1
XOR	6	2 / 1	4 × 7	7	11	3 × 3	5	< 1
XNOR	8	2 / 1	6 × 6	8	13	3 × 5	9	< 1
Half adder	10	2 / 2	7 × 6	8	55	5 × 5	13	10
c17	11	5 / 2	10 × 6	13	15	3 × 5	9	< 1
ParGen	14	3 / 1	9 × 10	14	27	3 × 8	16	6
4:1 MUX	16	6 / 1	11 × 8	19	9612	5 × 7	15	55
ParCheck	21	4 / 1	10 × 14	14	3014	6 × 7	15	224

#Gates Gate count plus fan-outs **CP** Critical path
Dimension Occupied area in FCN tiles **t in s** Time in seconds

TABLE II: Evaluating `ortho`

Benchmark			Proposed `ortho` [22]		
Name	**#Gates**	**I / O**	**Dimension**	**CP**	**t in s**
c432	551	36 / 7	426 × 161	584	< 1
c499	963	41 / 32	690 × 306	995	< 1
c1355	1515	41 / 32	1243 × 369	1611	< 1
c1908a	2043	33 / 25	1540 × 536	2077	< 1
c2670a	2455	155 / 64	1756 × 760	2511	< 1
c3540a	3588	50 / 22	2523 × 1111	3639	1
c5315a	5478	177 / 123	3857 × 1751	5577	2
c6288	6928	32 / 32	5714 × 1246	6957	2
ctrl	498	7 / 25	356 × 149	495	< 1
router	658	60 / 3	488 × 231	717	< 1
int2float	699	11 / 7	514 × 196	708	< 1
i2c	3508	133 / 127	2515 × 1123	3632	1
bar	8592	135 / 128	6547 × 2180	8724	6
sin	14314	24 / 25	10549 × 3828	14374	14
voter	39476	1001 / 1	30542 × 9935	40476	10

#Gates Gate count plus fan-outs **CP** Critical path
Dimension Occupied area in FCN tiles **t in s** Time in seconds

The `exact` approach is highly parameterizable and allows for exploration of e. g. different clocking schemes, wire crossings and length restrictions, designated I/O ports, (un-)balanced paths, etc. and guarantees minimality of the resulting layouts in terms of area. By this, the approach solves an $\mathcal{NP}$-complete problem [23] in an exact fashion and is therefore only applicable to rather small circuits. Resulting graphics of two different layouts for the same circuit are exemplarily shown in Figure 1.

The `ortho` technique on the other hand is based on a linear-time algorithm. It restricts the search space by assuming a fixed clocking scheme. While this leads to circuits which are not guaranteed to be minimal anymore, it allows for an efficient design even for circuits with high numbers of gates.

Eventually, both algorithms generate a gate-level abstraction of an FCN circuit grid, which can be compiled down to cell level in a technology mapping step. A pre-defined gate library used for physical simulations and visualization is QCA-ONE [24]. Experimental results comparing `exact` against the former state-of-the-art algorithm are reported in Table I; such demonstrating `ortho`'s scalability are printed in Table II respectively.

Having in mind the brevity of this demo proposal, we refer the reader to the original papers and the *fiction* documentation for further information and a user guide.

III. Conclusion

In this research demo proposal, we introduced *fiction*, an extensible open source framework written in C++ for physical design task, i. e. placement, routing, timing, and technology mapping, of Field-coupled Nanocomputing Circuits. The framework allows to explore various design criteria, comes with state-of-the-art algorithms, as well as scripting and logging functionalities. The efficiency of the implemented algorithms was indicated by case studies. We thereby provide a foundation for future research in the community.

Acknowledgment

We would like to thank Gregor Kuhn and Mario Kneidinger for code contributions and the authors of the *EPFL Logic Synthesis Libraries* for their inspiring work.

References

[1] N. G. Anderson and S. Bhanja, *Field-coupled Nanocomputing: Paradigms, Progress, and Perspectives*, 1st ed. New York: Springer, 2014.
[2] C. S. Lent and P. D. Tougaw, "A Device Architecture for Computing with Quantum Dots," *Proceedings of the IEEE*, vol. 85, no. 4, pp. 541–557, 1997.
[3] D. Giri, G. Causapruno, and F. Riente, "Parallel and Serial Computation in Nanomagnet Logic: An Overview," *TVLSI*, pp. 1–11, 2018.
[4] C. S. Lent *et al.*, "Molecular Cellular Networks: A non von Neumann architecture for molecular electronics," in *ICRC*, 2016, pp. 1–7.
[5] S. Bohloul, Q. Shi, R. A. Wolkow, and H. Guo, "Quantum Transport in Gated Dangling-Bond Atomic Wires," *Nano Letters*, vol. 17, no. 1, pp. 322–327, 2017.
[6] T. R. Huff, H. Labidi *et al.*, "Atomic White-Out: Enabling Atomic Circuitry through Mechanically Induced Bonding of Single Hydrogen Atoms to a Silicon Surface," *ACS Nano*, vol. 11, no. 9, pp. 8636–8642, 2017.
[7] X. K. Hu *et al.*, "Edge-Mode Resonance-Assisted Switching of Nanomagnet Logic Elements," *IEEE Trans. Magn.*, vol. 51, no. 11, pp. 1–4, 2015.
[8] J. Timler and C. S. Lent, "Power Gain and Dissipation in Quantum-dot Cellular Automata," *J. Appl. Phys.*, vol. 91, no. 2, pp. 823–831, 2002.
[9] J. Pitters, L. Livadaru, M. B. Haider, and R. A. Wolkow, "Tunnel Coupled Dangling Bond Structures on Hydrogen Terminated Silicon Surfaces," *JCP*, vol. 134, no. 6, 2011.
[10] F. S. Torres, R. Wille, P. Niemann, and R. Drechsler, "An Energy-aware Model for the Logic Synthesis of Quantum-Dot Cellular Automata," *TCAD*, vol. PP, no. 99, 2018.
[11] M. Walter, "fiction: An Open Source Framework for the Design of Field-coupled Nanocomputing," https://github.com/marcelwa/fiction, 2019.
[12] M. Soeken, H. Riener, W. Haaswijk, and G. De Micheli, "The EPFL Logic Synthesis Libraries," 2018, arXiv:1805.05121.
[13] E. Blair and C. Lent, "Clock Topologies for Molecular Quantum-Dot Cellular Automata," *Journal of Low Power Electronics and Applications*, vol. 8, no. 3, p. 31, 2018.
[14] R. Wang *et al.*, "Effect of a Clock System on Bis-Ferrocene Molecular QCA," *IEEE-NANO*, vol. 15, no. 4, pp. 574–582, 2016.
[15] V. Vankamamidi, M. Ottavi, and F. Lombardi, "Clocking and Cell Placement for QCA," in *IEEE-NANO*, vol. 1, 2006, pp. 343–346.
[16] C. A. T. Campos, A. L. P. Marciano, O. P. V. Neto, and F. S. Torres, "USE: A Universal, Scalable, and Efficient Clocking Scheme for QCA," *TCAD*, vol. 35, no. 3, pp. 513–517, 2016.
[17] M. Goswami *et al.*, "An efficient clocking scheme for quantum-dot cellular automata," *Electron. Lett.*, pp. 1–14, 2019.
[18] R. E. Formigoni *et al.*, "BANCS: Bidirectional Alternating Nanomagnetic Clocking Scheme," in *SBCCI*, 2018, pp. 1–6.
[19] F. S. Torres, M. Walter, R. Wille, D. Große, and R. Drechsler, "Synchronization of Clocked Field-Coupled Circuits," in *IEEE-NANO*, 2018, pp. 1–5.
[20] K. Walus, T. J. Dysart, G. A. Jullien, and R. A. Budiman, "QCADesigner: A Rapid Design and Simulation Tool for Quantum-dot Cellular Automata," *TNANO*, vol. 3, no. 1, pp. 26–31, 2004.
[21] M. Walter, R. Wille, D. Große, F. S. Torres, and R. Drechsler, "An Exact Method for Design Exploration of Quantum-dot Cellular Automata," in *DATE*, 2018, pp. 503–508.
[22] M. Walter, R. Wille, F. S. Torres, D. Große, and R. Drechsler, "Scalable Design for Field-coupled Nanocomputing Circuits," in *ASP-DAC*, 2019, pp. 197–202.
[23] M. Walter, R. Wille, D. Große, F. S. Torres, and R. Drechsler, "Placement & Routing for Tile-based Field-coupled Nanocomputing Circuits is NP-complete," in *JETC*, 2019.
[24] D. A. Reis *et al.*, "A Methodology for Standard Cell Design for QCA," in *ISCAS*, 2016, pp. 2114–2117.
[25] A. Trindade, R. S. Ferreira, J. A. M. Nacif, D. Sales, and O. P. V. Neto, "A placement and routing algorithm for Quantum-dot Cellular Automata," in *SBCCI*, 2016.