Asynchronous Circuit Design and Beyond

Chris J. Myers[1]

University of Utah, Salt Lake City, UT 84112, USA,
myers@ece.utah.edu,
WWW home page: http://www.async.ece.utah.edu/Myers

Abstract. In the 80s, there was a resurgence of interest in asynchronous circuit design. Technology challenges being faced by synchronous designers led researchers to reconsider the clock-less option. This work led to both new theoretical insights and computational design methods that were ultimately validated in a number of successful demonstration designs. After more than 30 years, there has been some industrial uptake of this research work, but the revolution that we envisioned has not occurred. Instead, the asynchronous mindset has had impact in a number of other domains from formal methods applied to mixed-signal and cyber-physical systems to design methods for synthetic biology. This paper will give a brief account of the asynchronous renaissance and its lasting impact.

Keywords: asynchronous circuit design, Petri nets, formal methods

1 A Brief History of Asynchronous Circuit Design

The universe doesn't allow perfection.
– Stephen Hawking, A Brief History of Time

Asynchronous circuit design has a long history. Many of the earliest mainframe computers including the ILLIAC and ILLIAC II designed at the University of Illinois and the Atlas and MU-5 designed at the University of Manchester in the 1950s and 1960s utilized asynchronous circuits. Asynchronous circuit design was chosen to improve reliability and provide easier maintenance [3]. Asynchronous circuit design though can be challenging due to the need to ensure that every transition on a signal wire is meaningful [14]. An asynchronous computation is coordinated using a *handshaking protocol* in which a circuit is *requested* to perform an operation, and it *acknowledges* completion of the operation. Even a single unwanted glitch on a request or acknowledge signal wire, known as a *logic hazard*, could lead to unintended behavior. A simple way to address this hazard problem is to utilize *synchronous circuit design*, which employs a periodic clock signal to filter out these glitches by allowing sampling of signals to occur only at prescribed times. For this reason, synchronous circuit design has become the dominant design methodology.

During the 1980s, however, asynchronous circuit design experienced a resurgence of interest. In particular, new effective design methodologies were developed to address the challenges to produce hazard-free asynchronous circuits.

There were two main camps: the language-based and the graph-based researchers. The language-based researchers included Alain Martin of Caltech [11] and Martin Rem of Eindhoven University [2]. They and their students developed design methods that started with behavioral descriptions of asynchronous designs written in a high-level language inspired by Hoare's *communication channels* [7], which are then compiled through a series of well defined transformations into a hazard-free asynchronous circuit implementation. The graph-based researchers included, among others, Tam-Anh Chu, a PhD student from the Massachusetts Institute of Technology [4], Teresa Meng, a PhD student from the University of California at Berkeley [13], and Alexander Yakovlev and Victor Varshavsky of the Leningrad Electrical Engineering Institute [19, 23]. These researchers and their colleagues developed design methods that started with graphical models, typically some variation on a *Petri net* [17], for their asynchronous designs. Using these models, all reachable states would be considered, sometimes implicitly, to produce asynchronous logic circuits free of hazards.

The late 80s and early 90s witnessed several exciting demonstrations of these new methodologies to produce asynchronous circuits with improved performance, power consumption, and robustness. The first fully asynchronous microprocessor was designed by Martin's group at Caltech in 1989 [12]. Utilizing techniques inspired by Ivan Sutherland's Turing Award paper on *micropipelines* [21], Steve Furber's group at the University of Manchester designed the first asynchronous microprocessor that was code-compatible with an existing synchronous processor [24]. At Phillips Research Laboratories, former students and colleagues of Martin Rem designed several low power circuits for commercial applications [1]. Finally, between 1995 and 1999, researchers working with Intel on the RAPPID project designed an asynchronous instruction-length decoder for the Pentium processor with three times better performance while consuming half the power [20].

These successes enabled several startup companies to explore asynchronous circuit design during the late 90s and early 2000s. For example, Alain Martin created Situs Logic, while his former students created Fulcrum Microsystems and Achronix Semiconductor. Other startup companies originated from Steve Furber's group including Cogency and Silistix. Finally, the Phillips group spun out a company called Handshake Solutions. These companies though faced the tremendous inertia in the semiconductor industry. The asynchronous solution had to provide not only substantial benefits, but it also must solve problems that could not be solved by synchronous methods. Therefore, the success of these startup companies has been quite limited to date.

2 The Asynchronous Mindset

> They are really smart, and we can teach them to do something real.
> – Manpreet Khaira, Top 10 Reasons to Hire an Asynchronous Designer

After the success of the RAPPID project, Intel hired numerous asynchronous researchers to join their Strategic CAD Labs that was led by Manpreet Khaira.

The quote above was from a panel discussion at the 1999 Symposium on Advanced Research in Asynchronous Circuits and Systems held in Barcelona Spain. This was the same conference in which the RAPPID results were first published and won the best paper award. Although perhaps a bit insulting to an asynchronous designer, I would argue that it need not be. The "asynchronous mindset" is indeed a very powerful tool that can be applied to a wide variety of applications. The world around us is inherently asynchronous, so those trained to reason in this way are perfectly equipped to be successful. This section briefly highlights three such research areas, ordered in increasing distance from electronic circuits, that we have applied, with some success, an "asynchronous mindset".

2.1 Formal Verification of Analog/Mixed-Signal Circuits

In order to design asynchronous circuits, one must reason about time as a continuous rather than a discrete variable. *Analog/mixed-signal* (AMS) circuits take this one step further and also require values to be represented as continuous variables. In this domain, we have extended a Petri net graph-based modeling formalism utilized for asynchronous design to represent these continuous voltages and currents [9]. This has enabled AMS circuits to be modeled and verified using techniques originally developed for the verification of asynchronous circuits [6]. Most recently, in collaboration with Alexander Yakovlev's group at Newcastle University, we have been closing the design loop by leveraging these formal techniques to optimize the design of asynchronous digital controllers for analog circuits [5].

2.2 Formal Verification of Cyber-Physical Systems

Cyber-physical systems (CPS) are any systems that tightly integrate computation, communication, and control with the physical world. These can include anything from nuclear power plants, to robotic surgeons, to autonomous vehicles. The physical world that these systems must interact with is inherently noisy, stochastic, continuous, and asynchronous. Since these systems are clearly safety critical, it is essential that they be formally verified to avoid catastrophic failures. Once again, a Petri net inspired modeling formalism, similar to the one used for asynchronous and AMS systems, can be efficiently employed to model and verify these systems [22]. We have also recently used a channel-level language inspired by the one that we used for asynchronous design [14] to model and formally verify a fault-tolerant asynchronous routing protocol for an automotive application [25].

2.3 Genetic Design Automation

Synthetic biology is an exciting area of research in which scientists are attempting to engineer new biological systems to solve a wide variety of environmental,

energy, and health problems. *Genetic design automation* (GDA) research is providing the design methods and tools to support this discipline [15, 16]. Since biological systems do not have a global clock, it only makes sense that these methods should be adopted from the asynchronous domain. To this end, we have abstracted biochemical models into asynchronous logical models, once again using an adapted Petri net formalism, enabling orders of magnitude more efficient analysis [8, 10]. We have also adapted asynchronous logic synthesis techniques to produce genetic circuit designs automatically [18]. Finally, we hope to some day potentially draw inspiration from the design in this very noisy environments to produce more robust asynchronous electronic circuits.

3 Discussion

While the asynchronous revolution that many of us hoped for when we began working in this area of research has not yet occurred, I believe we can take solace in the fact the "asynchronous mindset" is and will continue to have tremendous impact in a wide variety of fields. A continued emphasis in instilling this view of the world in our students make them better equipped to tackle the wide variety of problems that they will face in the ever changing asynchronous world.

References

1. Berkel, K.v., Burgess, R., Kessels, J., Peeters, A., Roncken, M., Schalij, F.: A fully-asynchronous low-power error corrector for the DCC player. IEEE Journal of Solid-State Circuits 29(12), 1429–1439 (Dec 1994)
2. Berkel, K.v., Rem, M.: VLSI programming of asynchronous circuits for low power. In: Birtwistle, G., Davis, A. (eds.) Asynchronous Digital Circuit Design. pp. 152–210. Workshops in Computing, Springer-Verlag, New York (1995)
3. Brearley, H.C.: ILLIAC II: A short description and annotated bibliography. IEEE Transactions on Computers 14(6), 399–403 (Jun 1965)
4. Chu, T.A.: Synthesis of Self-Timed VLSI Circuits from Graph-Theoretic Specifications. Ph.D. thesis, MIT Laboratory for Computer Science (Jun 1987)
5. Dubikhin, V., Myers, C.J., Yakovlev, A., Sokolov, D.: Design of mixed-signal systems with asynchronous control. IEEE Design & Test Magazine (2016)
6. Fisher, A., Batchu, S., Jones, K., Kulkarni, D., Little, S., Walter, D., Myers, C.: Lema: A tool for the formal verification of digitally-intensive analog/mixed-signal circuits. In: Circuits and Systems (MWSCAS), 2014 IEEE 57th International Midwest Symposium on. pp. 1017–1020 (Aug 2014)
7. Hoare, C.A.R.: Communicating Sequential Processes. Prentice-Hall, Englewood Cliffs, NJ (1985)
8. Kuwahara, H., Myers, C., Barker, N., Samoilov, M., Arkin, A.: Automated abstraction methodology for genetic regulatory networks. Trans. Comp. Syst. Biol. VI, 150–175 (2006)
9. Little, S., Walter, D., Myers, C., Thacker, R., Batchu, S., Yoneda, T.: Verification of analog/mixed-signal circuits using labeled hybrid petri nets. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 30(4), 617–630 (2011)

10. Madsen, C., Zhang, Z., Roehner, N., Winstead, C., Myers, C.: Stochastic model checking of genetic circuits. J. Emerg. Technol. Comput. Syst. 11(3), 23:1–23:21 (Dec 2014), http://doi.acm.org/10.1145/2644817
11. Martin, A.J.: Programming in VLSI: From communicating processes to delay-insensitive circuits. In: Hoare, C.A.R. (ed.) Developments in Concurrency and Communication. pp. 1–64. UT Year of Programming Series, Addison-Wesley, Reading, MA (1990)
12. Martin, A.J., Burns, S.M., Lee, T.K., Borkovic, D., Hazewindus, P.J.: The design of an asynchronous microprocessor. In: Seitz, C.L. (ed.) Advanced Research in VLSI. pp. 351–373. MIT Press, Cambridge, MA (1989)
13. Meng, T.H.Y., Brodersen, R.W., Messerschmitt, D.G.: Automatic synthesis of asynchronous circuits from high-level specifications. IEEE Transactions on Computer-Aided Design 8(11), 1185–1205 (Nov 1989)
14. Myers, C.: Asynchronous Circuit Design. John Wiley & Sons (2001)
15. Myers, C.J.: Engineering Genetic Circuits. Chapman and Hall/CRC (2009)
16. Myers, C.J.: Computational synthetic biology: Progress and the road ahead. Multi-Scale Computing Systems, IEEE Transactions on 1(1), 19–32 (2015)
17. Petri, C.A.: Communication with automata. Tech. Rep. RADC-TR-65-377, Vol. 1, Suppl. 1, Applied Data Research, Princeton, NJ (1966)
18. Roehner, N., Myers, C.J.: Directed acyclic graph-based technology mapping of genetic circuit models. ACS Synthetic Biology 3(8), 543–555 (2014), pMID: 24650240
19. Rosenblum, L.Y., Yakovlev, A.V.: Signal graphs: From self-timed to timed ones. In: Proc. of International Workshop on Timed Petri Nets. pp. 199–207. IEEE Computer Society Press, Los Alamitos, CA, Torino, Italy (Jul 1985)
20. Stevens, K., Rotem, S., Ginosar, R., Beerel, P., Myers, C., Yun, K., Kol, R., Dike, C., Roncken, M.: An asynchronous instruction length decoder. IEEE Journal of Solid-State Circuits 35(2), 217–228 (Feb 2001)
21. Sutherland, I.E.: Micropipelines. Communications of the ACM 32(6), 720–738 (Jun 1989)
22. Thacker, R., Jones, K., Myers, C., Zheng, H.: Automatic abstraction for verification of cyber-physical systems. In: Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems. pp. 12–21. ICCPS '10, ACM, New York, NY, USA (2010), http://doi.acm.org/10.1145/1795194.1795197
23. Varshavsky, V.I. (ed.): Self-Timed Control of Concurrent Processes: The Design of Aperiodic Logical Circuits in Computers and Discrete Systems. Kluwer Academic Publishers, Boston, Dordrecht, The Netherlands (1990)
24. Woods, J.V., Day, P., Furber, S.B., Garside, J.D., Paver, N.C., Temple, S.: AMULET1: An asynchronous ARM processor. IEEE Transactions on Computers 46(4), 385–398 (Apr 1997)
25. Zhang, Z., Serwe, W., Wu, J., Yoneda, T., Zheng, H., Myers, C.: An improved fault-tolerant routing algorithm for a network-on-chip derived with formal analysis. Science of Computer Programming (2016)