ELectronics EXpress

IEICE Electronics Express, Vol.16, No.11, 1–5

LETTER

Design of low-power low-area asynchronous iterative multiplier

Heng You[1,2], **Yong Hei**[1], **Jia Yuan**[1], **Weidi Tang**[3], **Xu Bai**[1,2], **and Shushan Qiao**[1,2a)]

Abstract In this paper, a 16 times 16 low-power low-area asynchronous iterative multiplier is proposed. The multiplier diminishes 2 bits at a time with an iterative structure, to filter out the useless switching activities, we employ a finishing detector to dynamically detect the end of the computation and stop iteration ahead of schedule. Additionally, with the employment of finishing detectors, the proposed multiplier could provide a much faster average speed than synchronous approach. Post-layout simulation results show that the asynchronous multiplier offers up to 74% power reduction compared with the synchronous design. Simultaneously, the proposed design also exhibits a prominent area reduction compared with other non-iterative multiplier benefited from the iterative architecture.

Keywords: low-power, low-area, iterative multiplier, asynchronous circuits, useless switching reduction

Classification: Integrated circuits

1. Introduction

With the development of the artificial intelligence, more and more computations are needed to train the model. As one of the fundamental operations, multiplication should be fulfilled with as less overhead as possible. The major target of multiplier is undoubtedly lowering down the power and area overhead without sacrificing the processing performance.

Various strategies have been devoted to decreasing the power of different type of multipliers [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. Some of them are briefly introduced below. In [1], three methods have been adopted to decrease the power consumption such as signal flow optimization, left-to-right leapfrog structure and upper/lower split structure. In [4], a technology called Spurious Power Suppression Technique (SPST) has been applied for low power purpose. In [6], the multiplier uses a detecting unit to detect the dynamic range of the inputs, and adopts three separate Wallace trees for the 4 bits, 8 bits, and 16 bits multiplications. This lowers down the power consumption with conspicuous area overhead. The design in [8] develops a dynamic-range detector to dynamically detect the effective dynamic ranges of two input operands. The detection result is used to pick the operand with smaller dynamic range for Booth encoding and deactivate the redundant switching activities in ineffective ranges using the way similar with [6]. The design in [10] realizes multiplications with a network of shifts, adders, and subtracters where the multiplier coefficients are constant to reduce power consumption, lowering down the flexibility of the multiplier. Tree-based or array-based multipliers usually come with a significant area overhead while shift-and-add multipliers need less area, consuming more energy. Shift-and-add multipliers have been used in many other applications for their simplicity and relatively small area requirement [11].

All the strategies above are synchronous methods, but asynchronous design is usually known as a powerful low power strategy because asynchronous computational blocks can be designed to consume energy only when and where needed [12]. Asynchronous strategies have been applied to so many low power designs [13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27]. Several classic asynchronous low-power designs are briefly introduced in the following text. In [14], a low power asynchronous AES core is presented and could cipher 128-bit data/key in 300 ns and consumes 5.47 mW. TrueNorth, a well-known neurosynaptic chip, is the largest chip developed at IBM Inc. with 5.4 billion transistors [18]. The event-driven asynchronous operation realizes 65 mW ultra-low power consumption for real-time operation over the entire chip. [21] presents an asynchronous FFT design for low-power M2M communication. For a 10 MHz input data rate, the FFT design consumes only 5.9 nJ of energy at 1 V voltage supply in a 65 nm process. In [25], an asynchronous neural signal processor is presented which demonstrates robust subthreshold operation down to 0.25 V, while consuming only 460 nW. As for asynchronous multiplier, [26] minimizes power consumption of a multiplier by three methods such as asynchronous control, radix-2 algorithm, and split registers. The design in [27] is scalable to arbitrary operand lengths while maintaining a constant cycle time per Booth iteration.

The results in [26] shows the effectiveness of asynchronous control in low power design. However, the early termination scheme in [26] just divides the cycles into two kinds, the shift registers still need to shift during early termination cycles, wasting too much energy and computation time. In this work, we propose an iterative multiplier architecture which combines asynchronous control and dynamic-range detecting methods. Once the multiplier finishes effective computations, the controller no longer needs to generate any request signals. With flexible control strategy of asynchronous designs and pre-decision of dynamic-range detector, our proposed architecture can not only lower down the power consumption but also provide a much faster average speed. Benefited from the iterative architecture, the multiplier is also an area efficient design.

---

[1] Institute of Microelectronics of Chinese Academy of Sciences, Chaoyang, Beijing 100029, China
[2] University of Chinese Academy of Sciences, Shijingshan, Beijing 100049, China
[3] University of Science and Technology of China, Shushan, Anhui 230027, China
a) qiaoshushan@ime.ac.cn

DOI: 10.1587/elex.16.20190212
Received March 29, 2019
Accepted May 8, 2019
Publicized May 23, 2019
Copyedited June 10, 2019

2. Architecture of the proposed multiplier

The proposed low-power asynchronous iterative multiplier consists of a group of input latches, a dynamic range detector, an iterative calculation unit composed of a 16 times 2 multiplier and an 18 bits full adder, a 32 bits shifter and a group of output latches as shown in Fig. 1.

Since synchronous shift-and-add based multiplier fulfills a multiplication with fixed number of cycles, the switching activities become futile once the rest of M1 digits are all zeroes, wasting too much energy. In order to overcome the shortcomings of the synchronous approach, we fabricate a dynamic range detector comprised of a shifter and NOR gates. The shifter shifts 2 bits per clk1 cycle, results in the digits that should be sent to iteration unit locating in lower 2 bits, the rest of the digits in upper 14 bits. While the upper 14 bits are all zeroes, namely, the output of the NOR gate turns into one, it means the effective computation will be finish after the next clk1 cycle, so a stop signal is generated to ask the PG (pulse generate) unit to stop generating req/clk1 signals after the next clk1 rising edge. The clk2 signal is controlled by req signal, the last req signal will generate the last clk2 signal after accomplishing the last iteration, simultaneously, telling the output latch controller the end of the computation. For different iterations, different number of shifting bits should be applied to get the correct result.

The multiplier operates as follows: The input data comes with the request signal Rin asserting to one, then the dynamic range detector begin to detect whether the iterative operations need to be stopped ahead of schedule. The iterative calculation unit calculates 2 bits per cycle until the out of the dynamic range detector turns to one and the PG does not generate clock pulse. Finally, the result of the iterative unit is shifted through a 32 bits shifter to get the correct result, and the result is pulled out with the request signal Rout asserting to one.

The detailed architecture of the control units such as four-phase latch controller, PG (pulse generate) and PB (pulse bucket) will be discussed in **Section 2.1** and **2.2**.

2.1 Four-phase latch controller

As is well known that latches occupy half the area and capacitance of edge-triggered registers, hence replacing registers with latches could achieve smaller area and power consumption. In synchronous design, employing latches would make it hard to analysis timing constrains of the whole circuit, but with the more flexible control strategy, it's convenient to do such replacement in asynchronous approach. In our design, we replace the input and output registers with latches, concomitantly, a four-phase single rail latch controller has been developed as shown in Fig. 2 which is based on Liu's controller [26].

The latch controller is designed as fully-decoupled to attain higher speed. In the meantime, for the sake of power reduction of our design, the controller is designed as normally opaque so that glitches will be prevented from propagating into the multiplier. The corresponding timing diagram is shown in Fig. 2(b). In consideration of the timing assumption of bundled-data, proper delay lines should be inserted between the latch controller and adjacent stage. Since the rising edge of request signal indicates the arriving of stable data, the Rout signal of input latch controller and the Rin signal of output latch controller should be delayed to satisfy the setup time of the registers and latches. The falling edge is just used to reset the multiplier and shorter delay would attain higher speed, so asymmetric delay-line is applied to both DL1 and DL3 in Fig. 1, using a cascade of asymmetric rising and falling delay inverters shown in Fig. 3(a) and (b).

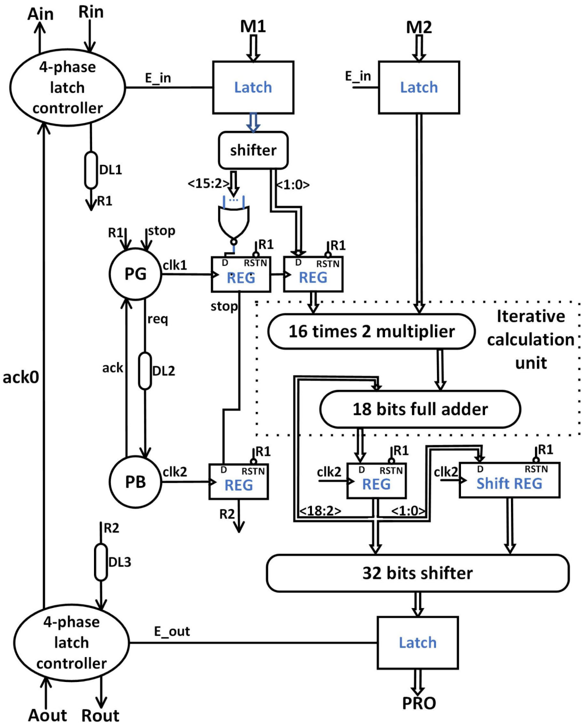
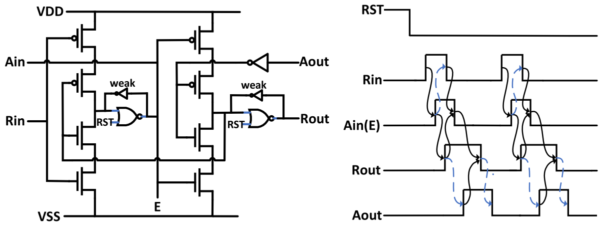
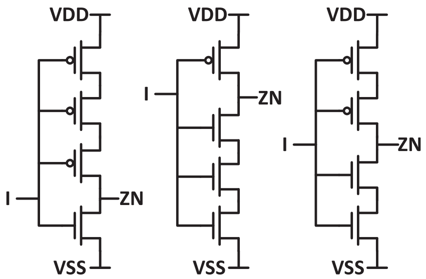


Fig. 1. Top-level architecture of the proposed multiplier.



(a) (b)

Fig. 2. (a) Four-phase latch controller. (b) Corresponding timing diagram.



(a) (b) (c)

Fig. 3. (a) Asymmetric rising inverter. (b) Asymmetric falling inverter. (c) Symmetric stack inverter.

2.2 Asynchronous iteration controller

A two-stage pipeline is applied in body of the multiplier, one for dynamic range detection while the other for computation as shown in Fig. 1. Since the computation adopts an iterative architecture, employing latches as storage units would entangle the iteration controllers, simultaneously, slowing down the multiplier. Hence, even though latches have half the area and capacitance, we employ edge-triggered registers as the storage units in order to remedy the shortcomings. With the employment of registers, true-four-phase handshaking protocol [28] has been applied to decrease the area and power consumption of the delay-line. The architecture of PG (pulse generate) and PB (pulse bucket) is shown in Fig. 4(a) and Fig. 4(b), while Fig. 4(c) illustrates the corresponding timing diagram.

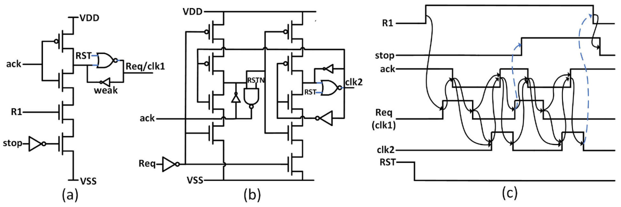


Fig. 4. (a) Schematic of PG. (b) Schematic of PB. (c) Corresponding timing diagram.

Fig. 4(c) displays a complete operation where a multiplication needs to be iterated twice. The solid line in Fig. 4(c) means the transition could occur with the listed signals while the dash line means the transition need other conditions. For example, the rising edge of stop needs the upper 14 bits of M1 become all zeroes last clk1 cycle. It is obvious that the latency between clk1 and clk2 includes both phases of the delay line, making it possible to use symmetric delay line. In particular, the symmetric delay line could be half the size of asymmetric delay line, reducing energy and area overhead of delay line. Stack architecture shown in Fig. 3(c) is applied in DL2 to further reduce the overhead of delay line.

2.3 Delay line design

As the bundled data scheme is employed in the proposed multiplier, a timing constraint must be enforced: the delay of the request signal must always be longer than the worst-case data transmission. To support this constraint, a matched delay must be inserted. Thus, the design of the delay elements is also critical to support the correct calculation of the proposed multiplier.

The body of the iterative unit is synthesized using Design Compiler (DC) and place-and-routed with IC Complier (ICC) so that we could get the delay information of the data path from the timing reports. And then the delay of the delay elements is adjusted manually to satisfy the timing assumption of the bundled-data implementation. Meanwhile, appropriate slack is inserted to guarantee the correct operation of the multiplier.

3. Results and discussion

The asynchronous iterative multiplier is implemented using SMIC 55 nm CMOS technology. The asynchronous handshaking controllers are implemented using full-custom design strategy while the body of the iterative unit synthesized using Design Compiler (DC) and place-and-routed with IC Complier (ICC). The two parts are merged in Cadence Virtuoso Layout to get the intact layout of our asynchronous iterative multiplier. For comparison, two synchronous iterative multiplier are implemented with a similar architecture of proposed asynchronous multiplier. The only difference between the two synchronous implementations is that one adopts clock-gating strategy while the other not.

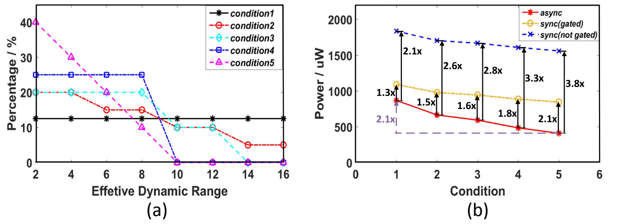


Fig. 5. (a) Dynamic range of 5 benchmarks. (b) Power dissipation of three multipliers (sync-synchronous; async-asynchronous).

To determine the effectiveness of the power reduction for different conditions of the proposed asynchronous architecture, 5 benchmarks with different effective dynamic ranges are applied to the three multipliers where effective dynamic range of an input vector represents the first bit that is not zero. Each benchmark is displayed in Fig. 5(a), together with 1000 pairs of numbers as the input vectors. Conditions 1 to 5 in Fig. 5(a) represent the distributions of the input vectors of 5 benchmarks. The effective dynamic range tapers off from condition 1 to condition 5 so that we could analysis the power consumption for different kind of input vectors.

Synopsys Hsim is employed to measure the average power of the three multipliers. Fig. 5(b) demonstrates the post-layout simulation results of the power dissipation of the three multipliers for different conditions. The power simulation is performed at a speed of 110M multiplications per second with a 1.2 V voltage supply. The execution time is 9 us so that the multiplier could calculate 1000 pairs of input vectors at the speed. It is obvious that the proposed asynchronous multiplier reaps big power benefits compared with synchronous approach as the result shown in Fig. 5(b). As discussed in previous Section 2, our proposed architecture iterates different times for different input numbers, reducing useless iteration activities, hence the smaller the input effective dynamic range is, the more useless iteration the asynchronous multiplier reduces, in the meantime, the less time the complete multiplication spends. It is easily discovered that the proposed multiplier harvest a more significant power reduction with smaller effective range of the input vectors.

Table I shows different time the multiplier needs to finish the computation for different effective dynamic range of input vectors. Since the input vectors with smaller effective dynamic range spend less time to finish the computation, we could get a much faster average speed

Table I. Time overhead for different effective dynamic range of input vectors

Effective Dynamic Range	2	4	6	8	10	12	14	16
Time (ns)	**2.9**	**3.8**	**4.7**	**5.5**	**6.4**	**7.3**	**8.2**	**9.0**

than the worst case of 110M, solving the low speed problem of the sequential multipliers to a certain extent.

As for the difference between the power dissipation of two synchronous approaches, multiple clock cycles are needed to finish the multiplication, but the input and output registers are worked only once during several clock cycles, so many useless switching activities of the registers will waste so much energy if clock-gating technology is not applied.

Table II illustrates the comparison between our proposed multiplier with some existed low-power multipliers. Once the power consumption is related to the effective dynamic range of input vectors, we list the benchmark characteristics of each multiplier. For comparison, we normalize the power consumption and area using the formula based on [29, 30].

$$Norm.energy = \frac{Energy}{(tech./55\,\mathrm{nm}) * 3.7} \quad Norm.area = \frac{Area}{(tech./55\,\mathrm{nm})2 * 1618} \tag{1}$$

The proposed design shows a small area cost far less than the cost of other non-iterative multipliers. As is well-known that iterative multiplier usually consumes more energy than non-iterative multiplier due to the registers for intermediate data storing, but the proposed iterative multiplier could eliminate most useless switching activities to get a conspicuous power benefit compared with other exiting low power multipliers.

Table II. Comparison of different multipliers in terms of energy per multiplication (sync-synchronous; async-asynchronous)

Design	Huang [1]	Kuang [8]	Liu [26]^{*1}	H. [27]^{*1}	Proposed
Feature	32 b sync	16 b sync	32 b async	Scalable^{*3} async	16 b async
Tech.	0.18 µm	0.13 µm	0.18 µm	0.18 um	55 nm
Energy (pJ)	196.5	27.7	64.7	170	3.7
					7.9
Norm. energy	16.23	3.17	5.34	14.04	1
					2.14
Benchmark Feature^{*2}	12 b	5 b	10 b	8 b	4 b
					9 b
Area (µm^2)	74598	19995	N.A.	N.A.	1618
Norm. area	4.30	2.21	N.A.	N.A.	1

[*1] based on pre-layout simulation;
[*2] average effective dynamic range of input vectors;
[*3] use 16 bit for comparison.

4. Conclusion

In this paper, a low-power low-area asynchronous iterative multiplier has been presented. Our proposed asynchronous multiplier combines asynchronous control strategy and dynamic-range detecting method to eliminate useless switching activities. The post-layout simulation based on SMIC 55 nm CMOS process shows that the asynchronous multiplier consumes 408 µW at a speed of 110M multiplications per second for condition5 shown in Fig. 5(a), offers a 74% power reduction compared with synchronous design.

By replacing some registers with latches, the area of asynchronous multiplier is similar with synchronous approach even though extra controllers are needed in asynchronous design. Benefited from the iterative architecture, the proposed asynchronous multiplier shows a significant area reduction compared with other non-iterative multipliers.

Additionally, the asynchronous multiplier could provide a much faster average speed than synchronous multiplier as shown in Table I. Therefore, our proposed asynchronous multiplier is an excellent choice to provide fast, low power and low area operations, especially for the computations with smaller input vectors.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61474135).

References

[1] Z. Huang and M. D. Ercegovac: "High-performance low-power left-to-right array multiplier design," IEEE Trans. Comput. **54** (2005) 272 (DOI: 10.1109/TC.2005.51).

[2] A. K. Sahu and L. Kumre: "Low-power less-area bypassing-based multiplier design," ICICI (2017) 522 (DOI: 10.1109/ICICI.2017.8365186).

[3] X. Zhang, *et al.*: "32 bit × 32 bit multiprecision Razor-based dynamic voltage scaling multiplier with operands scheduler," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **22** (2014) 759 (DOI: 10.1109/TVLSI.2013.2252032).

[4] K.-H. Chen and Y.-S. Chu: "A low-power multiplier with the spurious power suppression technique," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **15** (2007) 846 (DOI: 10.1109/TVLSI.2007.899242).

[5] G. Challa Ram, *et al.*: "Design of delay efficient modified 16 bit Wallace multiplier," RTEICT (2016) 1887 (DOI: 10.1109/RTEICT.2016.7808163).

[6] H. Lee: "A power-aware scalable pipelined booth multiplier," Proc. IEEE Int. SOC Conf. (2004) 123 (DOI: 10.1109/SOCC.2004.1362373).

[7] Z. Liu, *et al.*: "An efficient floating-point multiplier for digital signal processors," IEICE Electron. Express **11** (2014) 20140078 (DOI: 10.1587/elex.11.20140078).

[8] S.-R. Kuang and J.-P. Wang: "Design of power-efficient configurable booth multiplier," IEEE Trans. Circuits Syst. I, Reg. Papers **57** (2010) 568 (DOI: 10.1109/TCSI.2009.2023763).

[9] S. Balamurugan, *et al.*: "Design of low power fixed-width multiplier with row bypassing," IEICE Electron. Express **9** (2012) 1568 (DOI: 10.1587/elex.9.1568).

[10] O. Gustafsson: "Lower bounds for constant multiplication problems," IEEE Trans. Circuits Syst. II, Exp. Briefs **54** (2007) 974 (DOI: 10.1109/TCSII.2007.903212).

[11] B. Parhami: *Computer Arithmetic Algorithms and Hardware Designs* (Oxford Univ. Press, Oxford, 2010) 2nd ed. 179.

[12] P. A. Beerel and M. E. Roncken: "Low power and energy efficient asynchronous design," J. Low Power Electron. **3** (2007) 234 (DOI: 10.1166/jolpe.2007.138).

[13] M.-C. Chang, *et al.*: "Low-power asynchronous NCL pipelines

with fine-grain power gating and early sleep," IEEE Trans. Circuits Syst. II, Exp. Briefs **61** (2014) 957 (DOI: 10.1109/TCSII.2014.2362639).

[14] N. El-meligy, *et al.*: "130 nm low power asynchronous AES core," Proc. ISCAS (2017) 1 (DOI: 10.1109/ISCAS.2017.8050832).

[15] I. Obridko, *et al.*: "Minimal energy asynchronous dynamic adders," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **14** (2006) 1043 (DOI: 10.1109/TVLSI.2006.884056).

[16] R. O. Ozdag, *et al.*: "An asynchronous low-power high-performance sequential decoder implemented with QDI templates," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **14** (2006) 975 (DOI: 10.1109/TVLSI.2006.884049).

[17] J.-H. Lee, *et al.*: "A low-power implementation of asynchronous 8051 employing adaptive pipeline structure," IEEE Trans. Circuits Syst. II, Exp. Briefs **55** (2008) 673 (DOI: 10.1109/TCSII.2008.921589).

[18] F. Akopyan, *et al.*: "Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst. **34** (2015) 1537 (DOI: 10.1109/TCAD.2015.2474396).

[19] B. Marr, *et al.*: "Scaling energy per operation via an asynchronous pipeline," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **21** (2013) 147 (DOI: 10.1109/TVLSI.2011.2178126).

[20] B. Ghavami and H. Pedram: "Design of dual threshold voltages asynchronous circuits," Proc. ISLPED (2008) 185 (DOI: 10.1145/1393921.1393970).

[21] B. Z. Tang and F. Lane: "Low power QDI asynchronous FFT," Proc. ASYNC (2016) 87 (DOI: 10.1109/ASYNC.2016.17).

[22] M. Rusci, *et al.*: "An event-driven ultra-low-power smart visual sensor," IEEE Sensors J. **16** (2016) 5344 (DOI: 10.1109/JSEN.2016.2556421).

[23] K.-S. Chong, *et al.*: "Sense amplifier half-buffer (SAHB) A low-power high-performance asynchronous logic QDI cell template," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **25** (2017) 402 (DOI: 10.1109/TVLSI.2016.2583118).

[24] W.-G. Ho, *et al.*: "Low power subthreshold asynchronous QDI 32-bit ALU based on autonomous signal-validity half-buffer (ASVHB)," IET Circuits Dev. Syst. **9** (2015) 309 (DOI: 10.1049/iet-cds.2014.0103).

[25] T.-T. Liu and J. M. Rabaey: "A 0.25 V 460 nW asynchronous neural signal processor with inherent leakage suppression," IEEE J. Solid-State Circuits **48** (2013) 897 (DOI: 10.1109/JSSC.2013.2239096).

[26] Y. Liu and S. Furber: "The design of a low power asynchronous multiplier," Proc. ISLPED (2004) 301 (DOI: 10.1109/LPE.2004.241065).

[27] J. Hensley, *et al.*: "A scalable counterflow-pipelined asynchronous radix-4 booth multiplier," Proc. ASYNC (2005) 128 (DOI: 10.1109/ASYNC.2005.6).

[28] P. A. Beerel, *et al.*: *A Designer's Guide to Asynchronous VLSI* (Cambridge Univ. Press, Cambridge, 2010) 162.

[29] Y.-W. Lin, *et al.*: "A dynamic scaling FFT processor for DVB-T applications," IEEE J. Solid-State Circuits **39** (2004) 2005 (DOI: 10.1109/JSSC.2004.835815).

[30] Y. Chen, *et al.*: "A 2.4-Gsample/s DVFS FFT processor for MIMO OFDM communication systems," IEEE J. Solid-State Circuits **43** (2008) 1260 (DOI: 10.1109/JSSC.2008.920320).