Performance Analysis of Two Synchronizers

Zhen Zhang, Jim Garside

School of Computer Science, University of Manchester,
Oxford Road, Manchester. M13 9PL United Kingdom
{zhangz,jgarside}@cs.man.ac.uk

***Abstract*—Synchronizers are necessary when importing signals into any clocked domain. As multiple different clocks become increasingly common on chips, synchronizers also proliferate. To achieve high performance it is important that the system designer is aware of the timing characteristics of different synchronizers -which are non-deterministic by nature – and can choose a design to meet their system requirements. This paper presents a method for analysing and depicting behaviour of synchronizers and applies this to two recognised designs. A detailed analysis of timing boundaries of the two synchronizers is presented. The probabilistic behaviour of data cycle is then investigated. Analytical expressions for the average data cycle are also derived.**

I. INTRODUCTION

Future VLSI systems will integrate an increasing number of modules and operate at faster clock frequencies. These developments complicate the design style where a single clock is distributed on an entire chip dramatically [1]. An alternative is to provide each module with its own clock. This approach requires integrating modules with different operating frequencies on a System on Chip (SoC) device. This also allows clock frequencies and voltages to change dynamically during chip operation [2] – [4].

A promising solution to these developments is the *Globally Asynchronous Locally Synchronous (GALS)* system [7]. A SoC device with GALS architecture typically consists of multiple modules, each operating with a potentially unrelated local clock. Communication between different modules is *asynchronous*, where no common clock is used to implement sequencing. The asynchronous solution eliminates complicated clock distribution problem. It also exhibits more flexible performance with voltage and temperature variations [5]-[9].

Data synchronization and communication across clock domains is a major challenge for designing GALS system. Circuits that provide such interfaces are called *synchronizers*. Synchronizers are expected to achieve correct data transfer while providing low latency and high data rate.

The mutual relationships of a pair of clock domains are analysed and classified in [10] and [11]. The most general case is where two corresponding modules are asynchronous. In this case, there is no known frequency or phase relation between them. This enforces the need for synchronization during each communication and universal synchronizer circuits are required. The family of two-flop synchronizers [12][13] and two-clock FIFOs [14] can serve this purpose. However, performance overhead is possible because universal synchronizers are designed handle all possible clock relationships and they can not be optimized for certain clock relationships.

Clock relationships, required reliability and throughput can influence designer's choice of synchronizers. Because synchronizer performance is influenced by these factors, it is difficult to predict its properties by cursory inspection. This paper introduces a systematic analysis applied to two synchronizers with varying clock relationships.

This paper is organized as follows. In Section 2, operation principle and timing boundaries are presented in detail. In Section 3, average data cycle for two-flop synchronizer is derived. In Section 4, the same analysis is applied to a fast four-phase synchronizer [12]. Conclusions and future work are presented in Section 5.

II. TIMING BOUNDARIES OF THE TWO-FLOP SYNCHRONIZER

Synchronizer performance is evaluated in terms of *data cycle* in this paper. The data cycle refers to the time between to consecutive writings of the first register of the receiver.

Standard two-flop synchronizers are widely used in industry [15]. The circuit diagram is shown in Fig 1. In this work, it is assumed that the REG flop on the receiver side is always ready to receive data from the sender.

A. Principles of Operation

The *forward cycle* is defined as the time from assertion of request signal (REQ) to de-assertion of it, i.e. REQ+→REQ-. The *backward cycle* refers to the time from de-assertion of REQ to the next assertion of it, i.e. REQ-→REQ+. Therefore, a complete data cycle consists of one forward cycle and one backward cycle. Both forward and backward cycles are on integral number of transmitter cycles.

Fig 2 shows the timing relation for signal transitions of forward cycle. When data is ready to be sent at the transmitter, it raises the REQ signal. On the next rising edge of receiver clock, REQ is sampled and R1 rises. The time difference between REQ rising and R1 rising is $\phi 1$. The acknowledge signal (ACK) and R2 rise consequently one receiver clock later. On the next rising edge of transmitter clock, A1 goes high. The time difference between assertions of ACK and A1 is $\phi 2$. It then takes another transmitter clock before A2 rises, which brings REQ down. Following the same sequence, R1, ACK, A1 and A2 consequently fall to zero. The falling of A2 will activate SNT to indicate a complete and acknowledged transfer of a single word.

B. Timing Boundaries of Two-Flop Synchronizer

The length of the data cycle can be seen as two successive assertions of REQ in the two-flop synchronizer. Because the forward and backward cycles are symmetric for two-flop synchronizer, only the forward cycle is analysed here. The complete data cycle can be obtained by doubling the forward cycle.

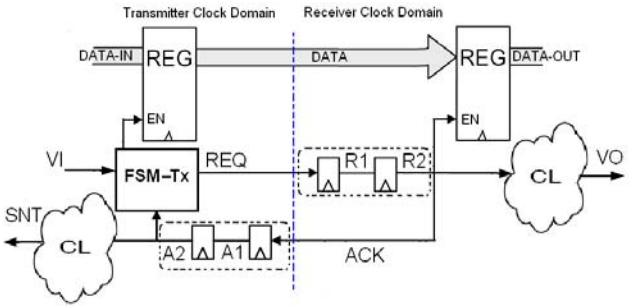

Fig. 1 A simple two-flop synchronizer

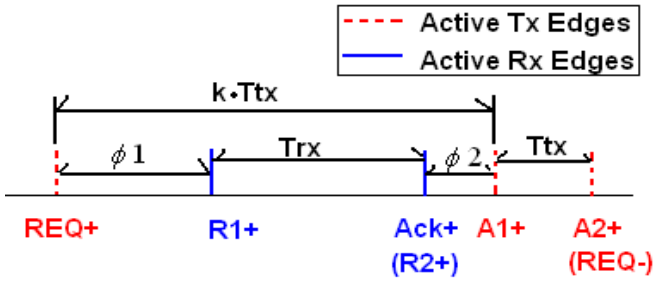

Fig. 2 Timing relation of forward cycle

From Fig 2, the forward cycle (REQ+→REQ-) is (k+1)Ttx, where k is a positive integer and Ttx is transmitter clock cycle. Equation (1) is obtained:

$$kTtx = \phi 1 + Trx + \phi 2 \qquad (1)$$

In Fig 2, $\phi 1$ represents phase shift between sender clock and receiver clock and it can be no greater than one receiver clock. Similarly, $\phi 2$ can be no longer than one transmitter clock, i.e.:

$$0 < \phi 1 \le Trx \qquad (2)$$

$$0 < \phi 2 \le Ttx \qquad (3)$$

$$(2) + (3), 0 < \phi 1 + \phi 2 \le Trx + Ttx \qquad (4)$$

Combining (4) into (1), we get

$$Trx < kTtx \le 2Trx + Ttx \qquad (5)$$

Let $x = \frac{Trx}{Ttx}$, then (5) becomes

$$x < k \le 2x + 1 \qquad (6)$$

The forward cycle is (k+1) times Ttx according to Fig 2. Inequality (7) can represent the timing boundaries for forward cycle in terms of Ttx:

$$x + 1 < k + 1 \le 2x + 2 \qquad (7)$$

The shaded area in Fig 3 shows the forward cycle timing boundaries. If the receiver clock is significantly faster than transmitter clock, it may take only two transmitter clocks to complete a forward cycle. As Trx/Ttx increases, the boundaries of forward cycle vary linearly and become wider. For a given ratio of Trx/Ttx, for example, Trx/Ttx=3, the possible value for forward cycle spans from 3 to 8. The reason is that although clock ratio is fixed, phase shift between the two clocks varies. The best and worst cases are exemplified in Fig 4. In the best case, the phase shift is so small that the receiver clock rises immediately after the rising edge of transmitter clock. REQ is sampled on this rising edge of receiver clock and R1 is raised. The worst case is that REQ signal just misses the rising edge of receiver clock and is delayed for almost an additional receiver clock cycle.

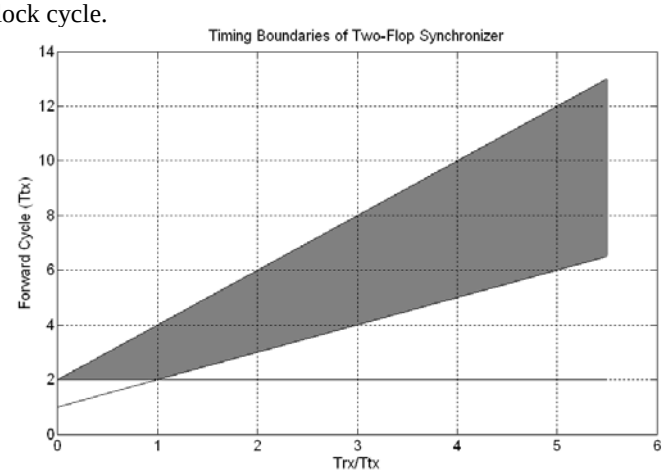

Fig. 3 Timing Boundaries of Forward Cycle

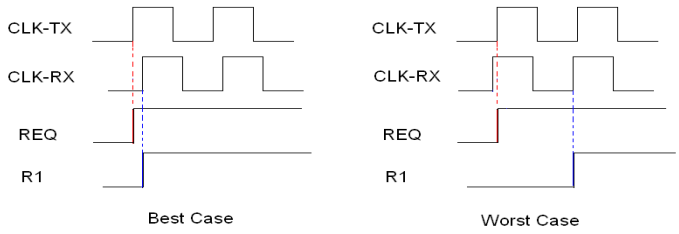

Fig. 4 Best and Worst Case for Sampling REQ

III. Average Data Cycles of the Two-Flop Synchronizer

Analysis of timing boundaries indicates that in the case when Trx/Ttx is greater than 0.5, different forward cycle values can result due to different phase shifts between transmitter and receiver clocks. Therefore, for a fixed Trx/Ttx ratio, it is not possible to tell the exact forward cycle time unless the phase relation is known between the two clock domains. However, with unrelated clocks, the phase shift varies. Another reason that makes it impossible to predict phase shift is that during burst-mode data transfer, the synchronization of one word may affect the synchronization of the next one. The phase shift may vary for synchronization of each word. These facts would add great difficulty in calculating exact forward cycle and hence data cycle for the two-flop synchronizer.

The *average* data cycle is however a good way of predicting cycle time. For a fixed Trx/Ttx ratio, the average data cycle is obtained from calculation of each possible data cycle and its probability of occurrence. It is assumed that the phase difference is random and evenly distributed. For example, when Trx/Ttx is 3.5, from Fig 4, forward cycle varies from 5 to 9. The probability of getting a 5 for forward cycle is 1/7, and this probability is denoted as P_5. Similarly, P_6, P_7 and P_8 have

the same probability of 2/7. P_9 is infinitesimally small and can be regarded as 0. For a different Trx/Ttx ratio, similar analysis can be applied. Table 1 summarizes the results of probability of forward cycle with Trx/Ttx ratio ranging from 0 to 4. In this table, x represents Trx/Ttx. The interval of x is incremented in steps of 0.5, because from the timing boundaries plots in Fig 4, the upper limit for forward cycle is *2x+2*, which reaches a new integer value at step of 0.5. The following analysis derives general expressions for probabilities of forward cycle and the average forward cycle.

TABLE I
PROBABILITY AND AVERAGE FORWARD CYCLE FOR TWO-FLOP SYNCHRONIZER

M/N=x	P2	P3	P4	P5	P6	P7	P8	P9	Aver
(0,0.5]	1								1
(0.5,1]	1/x-1	2-1/x							3-1/x
(1,1.5]		2/x-1	2-2/x						4-2/x
(1.5,2]		2/x-1	1/x	2-3/x					6-5/x
(2,2.5]			3/x-1	1/x	2-4/x				7-7/x
(2.5,3]			3/x-1	1/x	1/x	2-5/x			9-12/x
(3,3.5]				4/x-1	1/x	1/x	2-6/x		10-15/x
(3.5,4]				4/x-1	1/x	1/x	1/x	2-7/x	12-22/x

Let n be a positive integer. If $x \in [\frac{n}{2}, \frac{n+1}{2}]$, then $k \in [\frac{n+1}{2}, n+1]$, given n is odd or $k \in [\frac{n}{2}+1, n+1]$, given n is even.

Let P_m be the probability of getting a forward cycle of $m\,Ttx$. Then

$$P_{n+2} = 2 - \frac{n}{x}$$

If n is odd, $P_{\frac{n+3}{2}} = \frac{n+1}{2x} - 1$, and

$$P_{\frac{n+3}{2}+1} = P_{\frac{n+3}{2}+2} = \cdots = P_{\frac{n+3}{2}+\frac{n-1}{2}} = \frac{1}{x}$$

$$Aver = \frac{n+3}{2} P_{\frac{n+3}{2}} + (\frac{n+3}{2}+1) P_{\frac{n+3}{2}+1} + (\frac{n+3}{2}+2) P_{\frac{n+3}{2}+2} + \cdots + (n+2) P_{n+2}$$

$$Aver = \frac{3}{2}n + \frac{5}{2} - (\frac{3}{8}n^2 + \frac{1}{2}n + \frac{1}{8})\frac{1}{x} \qquad (8)$$

If n is even, $P_{\frac{n}{2}+2} = \frac{\frac{n}{2}+1}{x} - 1$, and

$$P_{\frac{n}{2}+3} = P_{\frac{n}{2}+4} = \cdots = P_{\frac{n}{2}+(\frac{n}{2}+1)} = \frac{1}{x}$$

$$Aver = (\frac{n}{2}+2) P_{\frac{n}{2}+2} + (\frac{n}{2}+3) P_{\frac{n}{2}+3} + \cdots + (\frac{n}{2}+\frac{n}{2}+1) P_{\frac{n}{2}+(\frac{n}{2}+1)} + (n+2) P_{n+2}$$

$$Aver = \frac{3}{2}n + 2 - (\frac{3}{8}n^2 + \frac{n}{4})\frac{1}{x} \qquad (9)$$

The above analysis requires that n is greater or equal to 1. The resultant intervals expressed in terms of n excludes (0, 0.5). Within this region, k has a uniform value of one, as is shown in Fig 3. The average is therefore one throughout this region. Equations (8) and (9) are expressions of average forward cycle (multiples of Ttx) at an arbitrary ratio of Trx/Ttx, provided this ratio is greater or equal to 0.5.

The average forward cycles in Fig 5 show certain non-linearity when two clocks have similar frequencies. This indicates that variations of phase shift can significantly affect the cycle time. However, as the Trx/Ttx ratio becomes larger, the average tends to be linear. On the other extreme, when transmitter clock is much faster than the receiver clock, the resultant forward cycle is constant. This indicates that phase shift has no impact on cycle time for large Trx/Ttx ratios. Therefore, when one clock is significantly slower than the other, the slower clock dominates. The impact of phase shift consequently decreases.

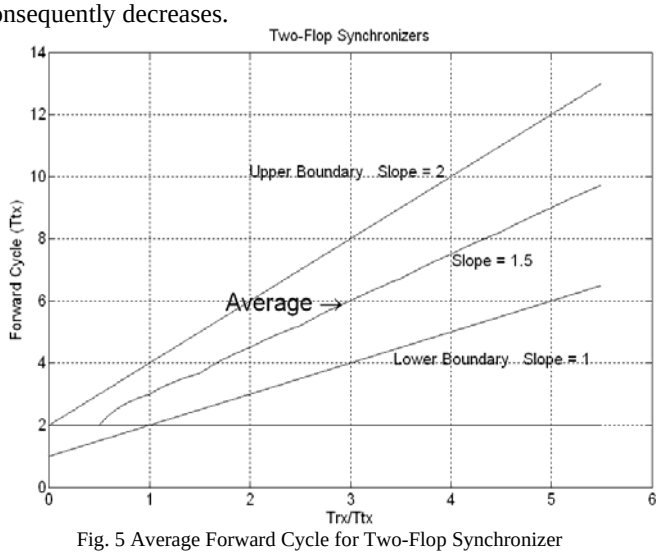

Fig. 5 Average Forward Cycle for Two-Flop Synchronizer

The linearity of average forward cycle can also be seen from Table 1. As the Trx/Ttx ratios increases, the number of the in-between probabilities (boxes with probability of 1/x) increases. When the ratio becomes large enough, these probabilities become dominant and are the main factors influencing the average value. Also, because these probabilities all have the same value, the resultant average will tend to be linear. This effect can also be seen from manipulations of Equations (8) and (9) as shown on the next page. The result indicates that limit of average forward cycle tends to be a straight line with the slope of 3/2, as is shown in Fig 5.

$$\text{Since } x \in [\frac{n}{2}, \frac{n+1}{2}],$$

$$\text{as } x \to \infty, n \approx 2x \text{ and } n \to \infty.$$

$$\lim_{\substack{x \to \infty \\ n = odd}} Aver = \lim_{x \to \infty} \{\frac{3}{2} 2x + \frac{5}{2} - [\frac{3}{8}(2x)^2 + \frac{1}{2} \cdot 2x + \frac{1}{8}] \frac{1}{x}\}$$

$$= \frac{3}{2}x + \frac{3}{2}$$

$$\lim_{\substack{x \to \infty \\ n = even}} Aver = \lim_{x \to \infty} \{\frac{3}{2} \cdot 2x + 2 - [\frac{3}{8}(2x)^2 + \frac{1}{4} \cdot 2x] \frac{1}{x}\}$$

$$= \frac{3}{2}x + \frac{3}{2}$$

IV. Timing Boundaries and Average Data Cycle of Fast Four-Phase Synchronizer

The standard two-flop synchronizer enables reliable communication between two clock domains [15]. However, it is limited to long data cycle. A fast four-phase synchronizer [12] (Fig.6) has been proposed to reduce data cycle by implementing asynchronous reset. This section analyses the timing boundaries and average data cycle of this synchronizer.

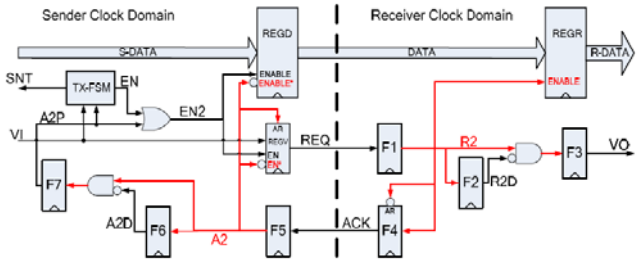

Fig. 6 Fast Four-Phase Synchronizer [15]

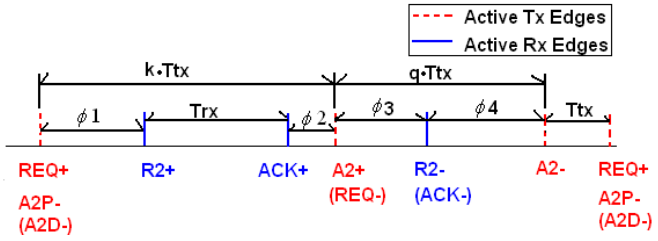

Fig. 7.Timing Relation of Forward and Backward Cycles

Unlike the two-flop synchronizer where REQ falls one transmitter clock later, the assertion of A2 asynchronously reset REGV flop and brings REQ down immediately. Therefore, one transmitter clock is saved before the backward cycle starts. Similarly, the falling edge of R2 triggers an asynchronous de-assertion of ACK during backward cycle. This saves one receiver clock.

Derivation of timing boundaries for forward cycle is similar to that of the two-flop synchronizer. From Fig 7, the following relationships can be obtained:

$$x < k < 2x + 1 \qquad (10)$$

Unlike the two-flop synchronizer where backward cycle and forward cycle are symmetric, the implementation of asynchronous reset in the fast four-phase synchronizer breaks this symmetry. The asymmetry leads to separate analysis for backward cycle. From Fig 7, the time boundary for backward cycle is *(q+1)Ttx*. It can be shown that inequalities (11) and (12) are the timing boundaries for fast four-phase synchronizer. The grey area in Fig 8 represents forward cycle timing boundaries and the area inside the thick lines represents backward cycle.

$$x < q + 1 < x + 2 \quad (x \geq 2) \qquad (11)$$

$$2 \leq q + 1 \leq x + 2 \quad (0 < x < 2) \qquad (12)$$

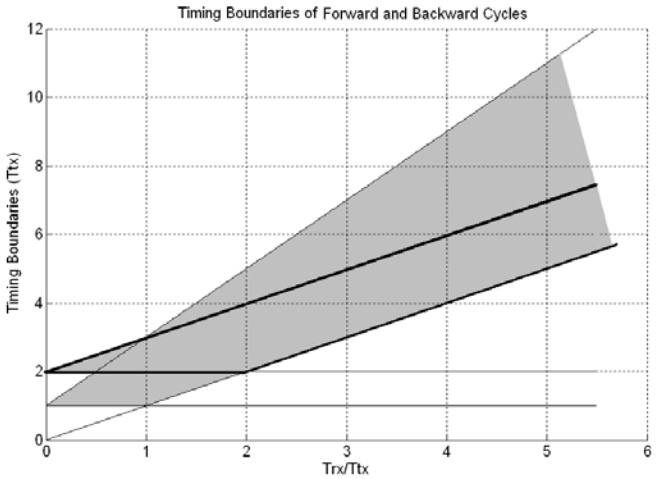

Fig 8 Timing Constraints of Fast Two-Flop Synchronizers

The analysis of the average data cycle for the fast two-flop synchronizer is done for the forward cycle and backward cycle separately. The forward cycle is similar to that of the two-flop synchronizer. This can be seen from the two timing relation diagrams in Fig 2 and Fig 7. The forward cycle of fast four-phase synchronizer is one transmitter clock faster than that of the two-flop synchronizer. Therefore, if the forward cycle of two-flop synchronizer is *(k+1)Ttx*, then the corresponding one for the fast four-phase synchronizer is *kTtx*. Similar analysis is applied here to calculate the average forward cycle for fast four-phase synchronizer as shown below.

$$\underset{n = odd}{Aver} = \frac{3}{2}(n+1) - (\frac{3}{8}n^2 + \frac{1}{2}n + \frac{1}{8})\frac{1}{x} \qquad (13)$$

$$\underset{n = even}{Aver} = \frac{3}{2}n + 1 - (\frac{3}{8}n^2 + \frac{n}{4})\frac{1}{x} \qquad (14)$$

$$where\ n \in Z^+ \ and\ x \in [\frac{n}{2}, \frac{n+1}{2}].$$

Again, the analysis assumes that *n* is greater than or equal to 1. The average within (0, 0.5) region is a constant value of one. Fig 9 shows the average of forward cycle time. As the Trx/Ttx ratio becomes larger, the average tends to be more linear. The slope of the line for average is 3/2.

Unlike forward cycle where general expressions can be derived for probabilities and average, the backward cycle does not show regular pattern. Fortunately, the variations for the backward cycle are limited to a small range and remain almost constant as the ratio Trx/Ttx increases. Intensive simulation is carried out to obtain average of backward cycle. Fig 9 shows the average forward and backward cycles. Bumps occur on the backward cycle curve where Trx/Ttx equals 1.5, 2.5, 3.5 etc. These fluctuations are gradually smoothed out as Trx/Ttx ratio increases, and the average tends to a straight line with slope of 1.

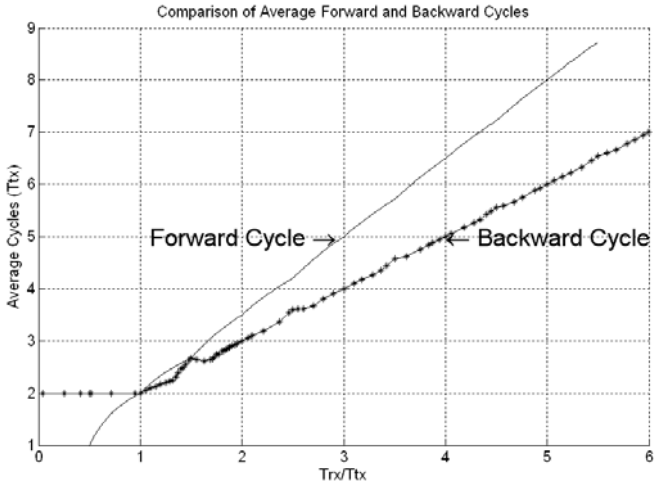

Fig. 9 Forward and Backward Data Cycles for Fast Four-Phase Synchronizer

V. Conclusions and Future Work

This paper presented a new method in characterizing data cycles for two-flop and fast four-phase synchronizers. It introduces systematic analysis on the probabilistic behaviour of the said synchronizers under various clock domain relationships. Timing boundaries for data cycles of the two synchronizers are derived. Analytical expressions for the average forward data cycles are also derived.

Metastability effect has been neglected in the analysis. The synchronizer designs allow for metastability by introducing a delay, which is one clock cycle for the two synchronizers analysed in this paper. The synchronizers are assumed to be impervious to the effects. The performance degradation caused by metastability can be added as a constant to the analysis.

Future work involves incorporating interconnection delay into the analysis. Also, it is expected that this analysis can be expanded to other types of synchronizers.

References

[1] E.G.Friedman, “Clock distribution networks in synchronous digital integrated circuits,” Proc.IEEE, vol.89, no.5,pp.665-692,May 2001.

[2] G. Semeraro, D.H. Albonesi, S.G. Dropsho, G. Magklis, S. Dwarkadas, M.L. Scott, "Dynamic frequency and voltage control for a multiple clock domain microarchitecture," Proc. of IEEE/ACM International Symposium on Microarchitecture, pp. 356-367, 2002

[3] L. S. Nielsen, C. Niessen, J. Sparsø, and C. H. van Berkel. “Low-power operation using self-timed and adaptive scaling of the supply voltage”, IEEE Transactions on VLSI Systems, 2(4):391-397, December 1994.

[4] W.R. Daasch, C.H. Lim, G. Cai, "Design of VLSI CMOS Circuits Under Thermal Constraint," IEEE Transactions on VLSI Systems, 49(8), 589-593, Aug. 2002.

[5] D.M.Chapiro, “Globally-Asynchronous Locally-Synchronous Systems,” PhD Dessertation, Stanford University, 1984.

[6] J.Bainbridge and S.Furber, "Chain: a Delay-Insensitive Chip Area Interconnect," IEEE Micro, 22(5):16-23,2002.

[7] E. Beigne, F.Clermidy, P.Vivet, A.Clouard, M. Renaudin, "An Asynchronous NOC Architecture Providing Low Latency Service and Multi-Level Design Framework," Proc. of ASYNC, 54-63, 2005.

[8] T Felicijan, S.B. Furber, "An Asynchronous On-Chip Network Router with Quality-of-Service (QoS) Support," Proc. of IEEE Int. SOC Conf., 274-277, 2004.

[9] T. Bjerregaard, J. Sparsø, "A Scheduling discipline for latency and Bandwidth Guarantees in Asynchronous Network-on-chip", Proc. ASYNC, 34-43, 2005.

[10] N. H.E. Weste and D. Harris, “CMOS VLSI Design: A circuit and systems perspective”, 3rd Edition, Pearson Addison-Wesley, 2004

[11] D. Messerschmitt, “Synchronization in digital system design,” IEEE J. Selected Areas Communications, vol.8,no.8,Oct.1990, pp.1404-1419

[12] R.Dobkin and R. Ginosar, “Zero latency synchronizers using four and two phase protocols”, Technical Report, 2007

[13] R. Dobkin and R.Ginosar, "Fast Universal Synchronizers," PATMOS, Sep. 200

[14] T. Chelcea, S. Nowick, “Robust Interfaces for Mixed-Timing Systems with Application to Latency-Insensitive Protocols”, DAC 2001

[15] R.Ginosar, “Fourteen Ways to Fool Your Synchronizer,”ASYNC, 89-96,2003