www.ietdl.org

Published in IET Circuits, Devices & Systems
Received on 13th February 2013
Revised on 4th July 2013
Accepted on 21st July 2013
doi: 10.1049/iet-cds.2013.0067

ISSN 1751-858X

Timing variation aware dynamic digital phase detector for low-latency clock domain crossing

Faiq Khalid Lodhi[1], Syed Rafay Hasan[2], Naeha Sharif[3], Nadra Ramzan[1], Osman Hasan[1]

[1]*School of Electrical Engineering and Computer Science, National University of Sciences and Technology (NUST), Islamabad, Pakistan*
[2]*Department of Electrical and Computer Engineering, Tennessee Technological University, Cookeville, TN, USA*
[3]*Department of Biomedical Engineering, Kyung Hee University, Yongin, South Korea*
E-mail: shasan@tntech.edu

Abstract: This study presents a digital phase detector-based approach for estimating and synchronising phase variations between clock domains. Instead of waiting for the resolution of metastability (with finite probability of failure), the authors propose a metastability avoidance algorithm, based on a sampling method for asynchronous signals. The results, using 90 nm inovation for high performance microelectronics (IHP) technology, show that the proposed design is about 1.5 times faster and provides a 35% improvement in Energy-Delay Product compared with the state-of-the-art approaches. Moreover, it completely prevents metastability failures.

1 Introduction

With the recent developments in VLSI technology, the transistor count has enormously increased along with the processor speed. This remarkable progress comes with the greater challenge of run-time (dynamic) phase variations because of process, voltage and temperature (PVT) and architectural influence on hot spot shifting [1]. These variations introduce irregular cycle-to-cycle delays in clock distribution networks, which in turn aggravates the clock skew and jitter problems giving rise to unmanageable global synchrony. With the advent of three-dimensional (3D)-integrated circuits (IC) technologies, using through silicon vias, the thermal management problem has further aggravated [2].

To alleviate these problems, physical design engineers divide the entire chip into several modules, such that every module is mutually asynchronous to one another. Such kind of multiple systems with multiple clock domains are commonly known as 'globally asynchronous locally synchronous' systems [3]. International technology roadmap for semiconductors in their 2011 edition [4] endorses this concept and foresees that 25% of the global signals will be asynchronous by the year 2015. However, the impeding factor in this strategy is the synchronisation problem at the clock domain crossing (CDC) interfaces. Several synchronisation solutions have been proposed, but each one has its own limitations, for example, the high latency of the traditional two flop synchroniser can jeopardise its application in high performance chips. Several other techniques utilise two flop synchronisers (asynchronous handshake) or variants of two flops, for example, level synchroniser, edge-detecting synchroniser and pulse synchroniser. They all have the synchronisation latency along with finite possibility of metastability failure. Moreover, these solutions require a separate phase detection mechanism to adjust to the run-time cycle-to-cycle delay variations.

This paper primarily focuses on overcoming the above-mentioned limitations by presenting a timing variation aware and real-time phase adjustable (dynamic) digital phase detector (DPD) circuit for clock domains crossing. This paper proposes a solution which can be applied to systems having any unrelated mutually asynchronous clocks (e.g. ratiochronous systems, where clock frequency ratio is rational number). Fig. 1 further elaborates the utility of the proposed design as it shows that the remote module and local module working at frequency 'F' and '$(M/N)F$', respectively, which can be an example of any ratiochronous systems. Certainly, our design can handle mesochronous systems as well because mesochronous is a special case of ratiochronous system. The proposed DPD exhibits the following two distinguishing features: (i) this design not only acts as a synchroniser, but also acts as a PD for mutually asynchronous clock domains. (ii) With its metastability tolerant sampling technique, it can select the most in-synch clock phase at run time. The experimental results indicate that the proposed DPD provides at least 25% improvement in latency compared with state-of-the-art designs (using the same process technology, IHP 90 nm CMOS). This way the proposed DPD provides functionally correct results at a 1.5 times lower latency than contemporary state-of-the-art synchronisers. The proposed design is comprised of all-digital components and does not presume metastability resolution duration. The paper provides the evidence of the above claims by presenting a comprehensive timing constraint analysis.

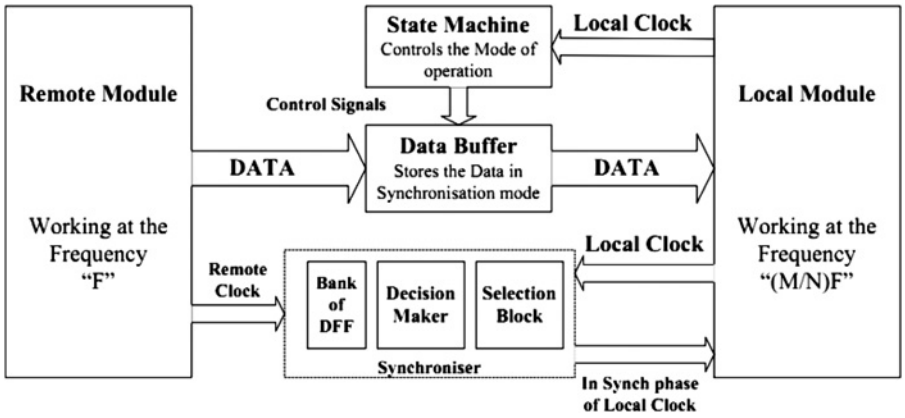

Fig. 1 *Block diagram of mesochronous synchroniser [17]*

2 Literature review

CDC in SoC designs have always been a great challenge for designers. Over the past decade, many synchroniser designs and techniques have been proposed in the literature. Dobkin and Ginosar in [5] have proposed novel synchronisers based on two-phase and four-phase protocols, which can operate as fast as two clock cycles for some cases. The authors have also analysed the performance of synchronisers in the presence of long wire delays and suggested a two-phase synchroniser to minimise this delay. Verbitsky [6] has described a four-stage mesochronous synchroniser that provides lower latency and full throughput. Dobkin and Ginosar [7] also proposed two synchronisers that are based on two-phase protocols. The first one is a low-latency two-flip-flop design and the other one uses a sub-cycle latency locally delayed latch synchroniser. All the above-mentioned synchronisers work fine when the communicating domains are adjacent to one another or are separated by long interconnects. However, they have a latency of approximately 2–4 clock cycles which may not be feasible in the case of high-frequency burst mode data transfer. Dally and Tell [8] proposed an all-digital synchroniser to estimate the phase difference between two periodic clock domains and to provide low-latency synchronisation. The synchroniser is integrated with a first in first out synchroniser to provide flow control. The periodic synchroniser offers latency slightly greater than half a clock cycle, along with a small probability of synchronisation failure. Alshaiekh *et al.* [9] explored the structural technique called wagging to improve the design of flip-flop synchronisers. The total latency is reduced by 15 and 25% compared with the two and three flip-flop synchroniser design, respectively. However, a special control circuit for is an additional requirement to ensure ordered and non-overlapping clock phases for the wagging synchroniser. Chabloz and Hemani [10] proposed novel mesochronous and plesiochronous interfaces, which offer the best and worst-case latency of less than half clock cycle and more than one clock cycle, respectively, for three flip-flops per data line and a single delay line on the strobe signal. Semiao *et al.* [11] presented a solution that permits the communication between multiple clock domains over an asynchronous bus. The proposed bus infrastructure uses an event driven protocol, where the event is associated to a clock. Another proposed solution utilises the concept of pausable clocking [12], where clocks are stopped temporarily when new data arrives. However, this technique requires clock with its inverted phase and thus introduces critical timing parameters, because of clock network delays [13, 14] and additional jitter. Moreover, the clock pause circuit relies on Seitz [15] arbiters which may take an unbounded time to resolve if an internal metastable state arises. Event protocol based [11] or handshake-based [3, 6, 7] techniques are most commonly used CDC interfaces to date. However, such CDC interfaces suffer from high synchroniser latency.

Our work is closely related to a work presented in [16]. Where the authors utilised two phases of the clock to detect the metastability in a mesochronous system. On the other hand, our work has following characteristics, which makes it distinct from [16]:

1. It is assumed in [16] that metastability is resolved in half a clock cycle (so that XOR gate in Fig. 12 of the mentioned paper will always spew out clean signal). This constraint can restrict the maximum speed of an interface. This sets a need for architectures that provide more freedom in the timing budget of mesochronous systems. Our design on the other hand does not assume this constraint, rather it utilises more samples to lead to a decision.
2. Our design provides solution to ratiochronous systems (system where two blocks have frequency ratio of a rational number) or any clock frequency combination (i.e. it is not limited to mesochronous communication).

3 Methodology and architecture

The proposed DPD architecture is primarily inspired by our previously reported mesochronous synchroniser [17], shown in Fig. 1. The remote and local modules in Fig. 1 communicate through a synchroniser and a buffer and the state machine controls the mode of operation in this design. Overall this design has two phases: 'synchronisation phase' and 'data transmission phase'. The different phases of the design are decided by a state machine. Remote module is sending the data at frequency 'F' to a local module that is operating at the frequency $(M/N) \times F$. So in order to receive the proper data, local clock should be synchronous to the remote module. Our novel technique chooses the phase of the local clock, which is in-synch with the remote modules. During this time, a data is held by the buffer module and which is controlled by a state machine.

In this paper, we propose a novel DPD synchroniser, shown in Fig. 2 that not only offers latency around half a clock cycle, but also alleviates the operational frequency

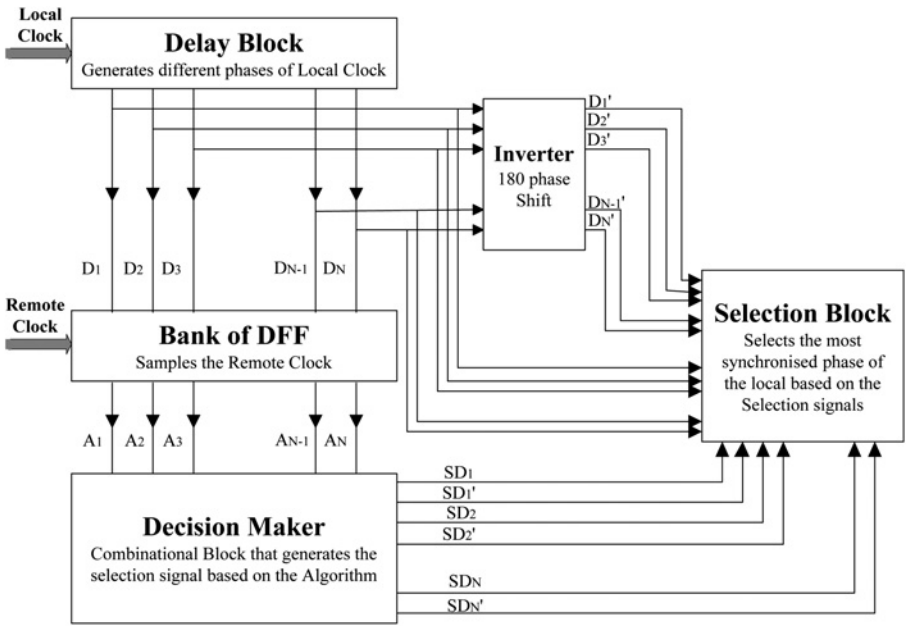

Fig. 2 *Block Diagram of the proposed DPD*

limit of the synchroniser scheme presented in [17]. The proposed synchroniser has four modules, that is, bank of delay flip flop (DFF), delay block, decision maker and the selection block as shown in Fig. 2.

The proposed system works as follows: the remote clock is sampled four times by the bank of DFFs. The number 4 is chosen to obtain a metastability tolerant sampling criterion as will be explained later. The delay block, given in Fig. 2, generates a set of equally delayed local clocks D1, D2, D3 and D4. The sampled signals are analysed by the decision maker to pass certain select signals to the selection block based on the combinational logic as a function of sampled signals from the DFF bank (explained further in Section 3.3). The selection block uses this information to output the clock phase that is most in-synch with the remote one. Next, we explain each individual block of Fig. 2 one by one.

3.1 Sampling technique

In this section, we present the proposed sampling technique. The sampling time interval is calculated based on the prior knowledge of the timing parameters, which includes the setup and hold time information of the DFF, and time period of remote and local clocks. The combined time duration of setup and hold time on each side of the clock edges is termed as the forbidden zone (t_{fz}), which is shown as the shaded region in Fig. 3. The sampling formula is obtained based on the premise that the proposed decision making algorithm can tolerate up to one metastable sample value (decision maker is elaborated in Section 3.3). Our sampling technique successfully withstands this constraint.

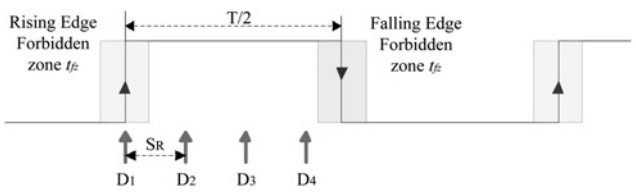

Fig. 3 *One possible set of sampling clock edges*

This constraint along with a priori information of worst-case timing parameters led us to calculate the sampling rate (S_R). In the presented technique, four samples ($N = 4$) per period are taken and all of these samples are taken within the half time-period duration. To avoid having more than one metastable value in the sampled vector, the sampling interval (the time interval between two successive rising clock edges, of the locally delayed clock) must be kept greater than the forbidden zone time. For robustness, we kept it greater than $1.5t_{fz}$.

$$S_R > 1.5t_{fz} \tag{1}$$

Fig. 3 shows that if the first sample falls in the rising edge of the forbidden zone of the clock, then there is a chance that the fourth sample (at a sampling interval of $3S_R$) may also fall in the falling edge of the forbidden zone, which will lead to more than one metastable output. To ensure that this does not happen and the fourth sample falls within the half time period, the following inequality should hold

$$T/2 - t_{fz} < 3S_R \tag{2}$$

$$S_R > (T - 2t_{fz})/6 \tag{3}$$

Generalising the above inequality for n samples leads to the following inequality

$$S_R > (T - 2t_{fz})/2(N - 1) \tag{4}$$

With these criteria, the sampled vectors can at most have one metastable output for any given remote clock cycle period. For example, Fig. 3 shows that D1 leads to a metastable output, whereas D2, D3 and D4 provide valid outputs.

It is to be noted that the choice of number of samples is based on the decision making algorithm. It has been observed that using four samples suffice the need of our solution (explained in subsection C). However, the optimum choice of even or odd number of samples, along with the total number of samples, requires further detailed analysis and is beyond the scope of this paper.

doi: 10.1049/iet-cds.2013.0067

3.2 Delay block and DFF bank

The purpose of the delay block is to generate the delayed versions of the local clock. In our case, the delay block generates a set of four equally delayed clocks: D_1, D_2, D_3 and D_4. These delayed local clocks are used by the DFF bank to sample the incoming remote clock (Fig. 4). The input of all the DFFs is the remote clock, whereas its output is the sampled vector [A_1, A_2, A_3, A_4]. Using the sampling criterion, as described in the preceding section, only one of these vectors ($A1$–$A4$) can be metastable.

3.3 Decision maker

The decision maker is the most vital part of the synchroniser. It is a combinational block that analyses the sampled signals: $A1$–$A4$ as shown in Fig. 4. It resolves the phase of the incoming clock and selects a delayed clock, which provides metastability free output.

The decision-making algorithm is primarily used to analyse the sampled information and selects the most in-synch sample. We have explained the algorithm assuming that there are four samples taken per period.

Columns $A1$–$A4$ of Table 1 provide raw sampling data to the input of the decision maker block in Fig. 4. The *X*s in each row of this table represent the possible metastable inputs to the decision maker. The 'intermediate output' column of Table 1 contains *YN*s (where *N* is an integer from 1 to 8), which provide the minterm representation of the corresponding row while assuming metastable outputs as do not care terms. For example, in row 2 of Table 1, Case 2 contains the logic value 111X, which leads to the Boolean expression '$A_1A_2A_3$'. The last two columns in Table 1 represent the 'selection signal' (output of decision maker) and 'decision' (output of selection block), respectively. These two signals are obtained after processing the intermediate output (*YN*s), based on the information given in Tables 2 and 3.

The 'selection signal' column in Table 1 represents the final decision of the decision maker block, which are represented as SDN and (its prime) SDN′ (where *N* is an integer that varies from 1 to 4). Before discussing the processing of *YN* signals, it is required to understand the need of SDN and its prime signals. For this purpose, consider the second row of Table 1. This case points to one of the local clocks from the delay block that has to be selected. The decision requires further processing, based on the information available in Tables 2 and 3. Scenario I of Fig. 5 illustrates the Case 2 of Table 3. It can be observed that metastability is occurring only once in the sampled vector, that is, D_4 is metastable, and hence the corresponding sample A_4 is metastable.

It is also important to note that the rows in cases 4–7 of Table 1 contain more '0's. This means that the remote clock is almost 180° out-of-phase with respect to the local clock (and most of its phases). For example, Case 6 in Table 1 has the logic value of 000X. So, if the phase difference between the local and the remote clock is more

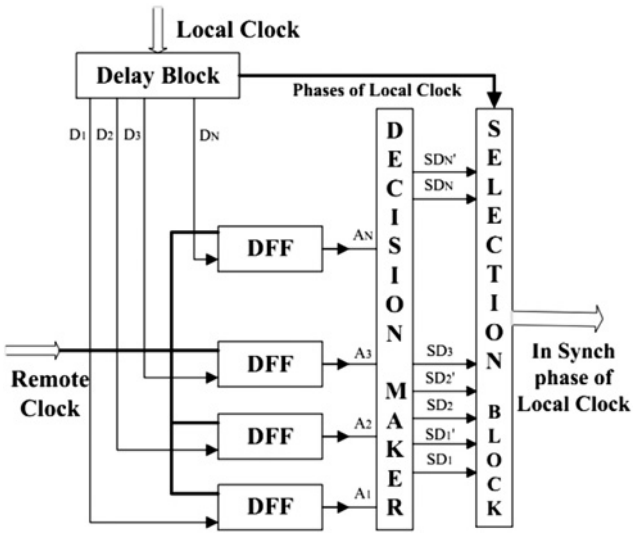

Fig. 4 *Working principle of digital phase detector (novel synchroniser)*

Table 2 Mutually exclusive minterms

Mutually exclusive minterms	Mutually exclusive minterms
$Y_1 = A_2A_3A_4$	$Y_2 = A_1A_2A_3$
$Y_3 = A_1A_2A_4'$	$Y_4 = A_1A_3'A_4'$
$Y_5 = A_2'A_3'A_4'$	$Y_6 = A_1'A_2'A_3'$
$Y_7 = A_1'A_2'A_4$	$Y_8 = A_1'A_3A_4$

Table 3 Reference table for mutually exclusive minterms: to resolve the conflicting minterms

CASE#	Intermediate output (true)	Selection signals	Decision
1	*Y*1, *Y*2	SD2	D2
2	*Y*1, *Y*8	SD3	D3
3	*Y*3, *Y*2	SD1	D1
4	*Y*3, *Y*4	SD4′	D4′
5	*Y*5, *Y*4	SD3′	D3′
6	*Y*5, *Y*6	SD2′	D2′
7	*Y*7, *Y*6	SD1′	D1′
8	*Y*8, *Y*7	SD4	D4

Table 1 Truth table for decision-making algorithm

CASE	A_1	A_2	A_3	A_4	Intermediate output	Selection signals	Decision
1	*X*	1	1	1	*Y*1	SD3	*D*3
2	1	1	1	*X*	*Y*2	SD2	*D*2
3	1	1	*X*	0	*Y*3	SD1	*D*1
4	1	*X*	0	0	*Y*4	SD4′	*D*4′
5	*X*	0	0	0	*Y*5	SD3′	*D*3′
6	0	0	0	*X*	*Y*6	SD2′	*D*2′
7	0	0	*X*	1	*Y*7	SD1′	*D*1′
8	0	*X*	1	1	*Y*8	SD4	*D*4

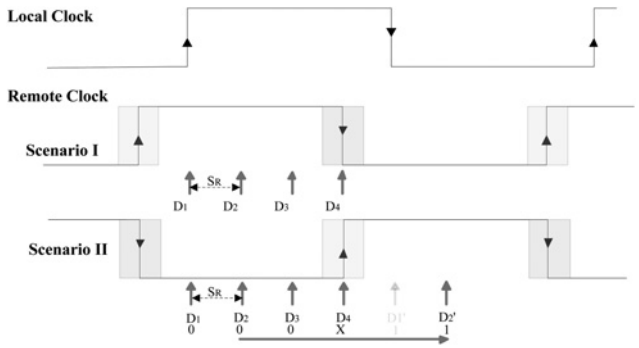

Fig. 5 *D4 is metastable at falling edge*

than 180°, then none of the delayed versions of the local clocks can be selected directly as the best in-synch one with the remote clock. For such conditions, our methodology recommends choosing the inverted versions of the locally delayed clocks. Scenario 2 of Fig. 5 illustrates this condition. It can be observed that the metastability is occurring at D_4 again. However in this case, in contrast to the Case 2, the safest phase that can be chosen in-synch with the remote clock is D_2'. In order to select this D2′ clock phase, decision maker block generates the signal SD2′.

A closer inspection of Table 1 shows that each one of the minterms ($Y1$–$Y8$) has two aliases, that is, more than one minterm can be asserted at the same time. For example, $Y2$ has aliases $Y1$ and $Y3$. To choose a unique solution, we need to avoid these aliases. In order to cope with this situation, we formulated Table 2. It can be observed that Table 1 provides two sets of mutually exclusive minterm expressions, which are $Y1$, $Y3$, $Y5$ and $Y7$ and $Y2$, $Y4$, $Y6$ and $Y8$, as shown in Table 2. Table 2 explains how to reduce this problem into eight new cases, based on the fact that at most only one of the conditions in each mutually exclusive column of Table 2 can be true. We can obtain three possible outcomes from Table 2, during the synchronisation process: (i) only one of the mutually exclusive minterm on the left column is asserted. (ii) Only one of the mutually exclusive minterm on the right column is asserted. (iii) One minterm from each column in Table 2 is asserted. The solution for first two possibilities is trivial. Table 3 shows the possible minterm combinations that can be asserted in the third case. In order to elucidate this problem, we assume that the obtained sample vector ($A1$–$A4$) is [1 1 1 1]. For such a case, both $Y1$ [$X111$] and $Y2$ [$111X$] are asserted. Under normal circumstances (when either $Y1$ or $Y2$ is asserted) assertion of $Y1$and $Y2$ triggers the selection of D3 and D2, respectively. In order to provide a conflict free output (when both $Y1$ and $Y2$ are concurrently asserted), possible metastability scenarios are analysed. Case #1 in Table 3 suggests that if $Y1$ and $Y2$ are concurrently asserted, then the sampled clock outputs are at logic '1' for four successive local clock pulses. These outputs are D1, D2, D3 and D4. Signal D2 is chosen because it is the middle signal, and hence it is most closely and safely in-synch with the remote clock. Same reasoning applies to all the cases in Table 3.

3.4 Selection block

In the proposed technique, eight different phases of the local clock are used to achieve the synchronisation within a complete clock cycle. Therefore this block is composed of eight two-input AND gates for the selection of above-mentioned eight different phases of the local clock. This block selects the particular phase from the given clock phases, generated by the delay block module in Fig. 4, based on the selection signals (SDNs) obtained from the decision maker block. If the selection signal SD1 is set, then this block passes the D1 phase of the local clock and similarly if the selection signal SD1′ is set, then this block passes the inverted phase of D1. Same reasoning applies for all the eight different conditions tabulated in selection signals column of Table 3.

4 Experimental results

We validated the proposed scheme by implementing the whole design (delay block, DFF, DM and selection block) in CADENCE CAD tools using IHP CMOS 90 nm technology. The key point of this analysis is to generate the delayed local clocks under the conditions mentioned in inequalities (1) and (3). Forbidden zone (t_{fz} duration) is obtained by performing the parametric analysis for true single phase clock (TSPC)-based DFF [18] with an arbitrary load of 100 fF, which is approximately 20 ps. In order to incorporate the process and temperature variations as well as variations occurring because of noise in the circuit, a value of 40 ps for t_{fz} is used for sampling rate (S_R). Incorporating this value in the inequalities (1) and (3), a range of sampling rate (60 ps $< S_R <$ 95 ps) was obtained for an operating frequency of 1.5 GHz ($T = 666$ ps). Therefore a sampling rate of 70 ps was chosen for simulation purposes. Based on the above-mentioned values, results for some selected cases from Table 1 are shown in Figs. 6 and 7. Sampling signals A1 to A4 corresponding to the four output signals from the DFF bank are generated after sampling the remote clock at four different clock phases (D1–D4) of the local clock (clock phases are multiples of SR).

4.1 Falling edge is in the forbidden zone

In Fig. 5, these signals generate a minterm $A1A2A3A4'$ under the condition that the falling edge of the remote clock lies in the forbidden zone (t_{fz}). This minterm corresponds to the logical value 111X or 1110, because the falling edge of D4 falls in t_{fz} region and generates a metastable output $A4$. By following the decision making algorithm, this logic value set enables $Y2$ and $Y3$ (as shown in Fig. 6). Thus, under the condition that the minterms are mutually exclusive, given in the truth table (Table 2), it selects the D1 phase of the local clock by enabling SD1.

4.2 Rising edge is in the forbidden zone

Similarly, if the rising edge of the remote clock lies in the forbidden zone, then the sampling output signals generate a minterm $A1'A2'A3'A4$ as shown in Fig. 7. This minterm corresponds to the logical value 000X or 0001 since the rising edge of the remote clock D4 falls at t_{fz} and generates a metastable output $A4$. According to the decision making algorithm, these logic values enable $Y6$ and $Y7$ (as shown in Fig. 7). Thus, under the condition of the mutually exclusive minterms in the truth table (Table 3), it selects the inverted version of the D1 phase (D1′) of the local clock by enabling SD1′ (Fig. 7).

doi: 10.1049/iet-cds.2013.0067

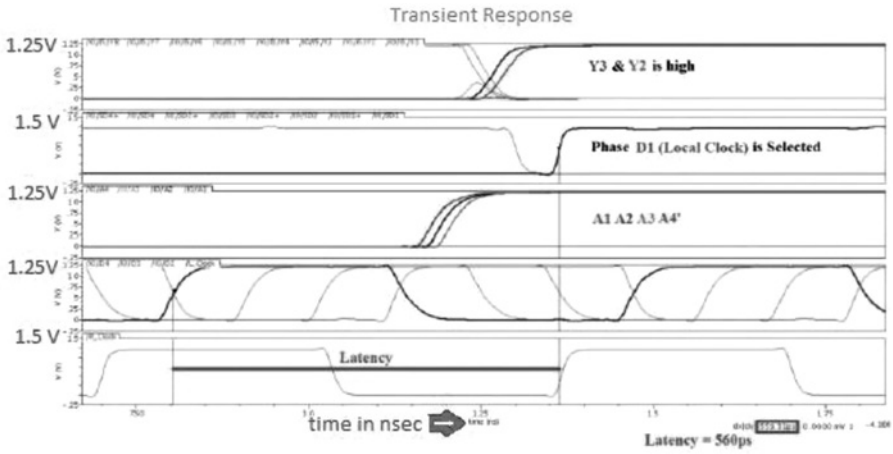

Fig. 6 *Selection signal and selected phase when A4 is metastable (falling edge)*

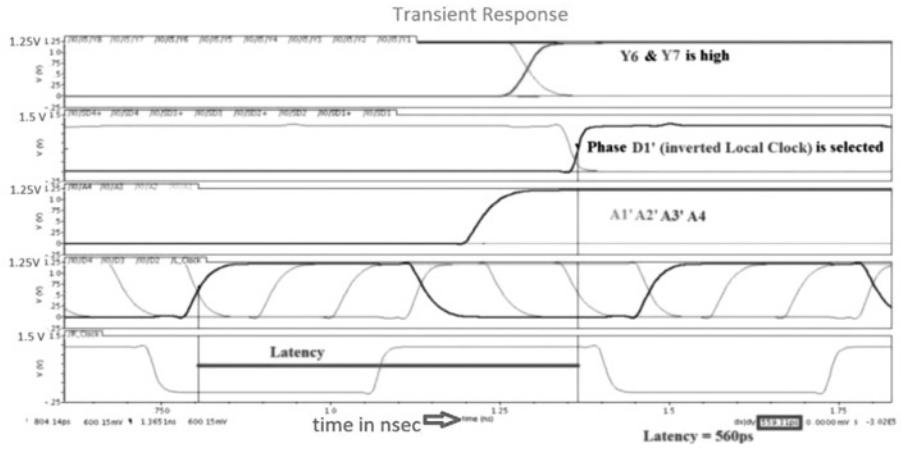

Fig. 7 *Selection signal and selected phase when A4 is metastable (rising edge)*

5 Discussion

The results, presented in the last section, show that the proposed design is capable of tolerating the metastability by generating different phases of the local clock. The main challenge of any synchroniser is to quickly resolve the metastability problem. As far as the proposed design is concerned, it avoids the metastability instead of waiting to obtain it resolved and thus is able to generate the metastable free output. Latency of this design depends on the combinational logic design. Therefore by optimising the combinational logic its latency can be further reduced. As it can be seen in Fig. 6 that this design takes 560 ps (which is less than a clock cycle) at 1.5 GHz to select the phase of the clock that generates the metastable free output.

Its latency is less than all the state-of-the-art synchronisers that are mentioned in Table 4, which also presents the power and energy comparisons. All the synchronisers are simulated at their respective maximum operating frequencies under similar design constraints (same technology, node load and data transfer frequency), as shown in Table 4 so that synchronisation is required at every clock cycle. In Table 4, latency and energy are calculated for five data cycles. The proposed design provides metastable free output after 560 ps, which is superior to the best state-of-the-art design (877 ps). Both consume comparable energy (3.26–3.27 mJ). Thus, the proposed DPD trims down the latency and operating frequency limitations of the existing low latency synchronisers without compromising power and energy.

Under an extreme scenario of phase clock changing in each cycle, our design would require to switch between synchronisation and data sampling mode very regularly. To our knowledge, such an extreme scenario does not occur very often. However, our design is capable of coping with

Table 4 [19] Comparison with the state-of-the-art synchronisers for five data cycles

Synchronisers	Maximum operating frequencies, GHz	Latency (*D*), ps	Energy (*E*), mJ	ED product, pJs
level synchroniser [19]	1.1	890	3.41	3.035
edge-detecting synchroniser [19]	1.0	960	3.50	3.360
pulse synchroniser [19]	1.0	990	3.62	3.584
low latency synchroniser [17]	1.1	877	3.27	2.867
proposed synchroniser (DPD)	1.7	560	3.26	1.825

such a scenario, provided clock period of the faster design is less than the latency of the design.

6 Conclusion

In this paper, a novel solution for CDC is proposed. This design acts as a synchroniser as well as a PD. It is shown that our design avoids metastability and at the same time provides latency lower than one clock cycle. The proposed sampling criteria along with our novel design implementation make our design free from any possible metastability failure. The paper also presents a comparative analysis of our DPD with some of the widely used contemporary synchronisers. It has been shown that, given similar design constraints, our design is about 1.5 times faster than the fastest state-of-the-art design presented in Table 4. The experimental results further show that our solution is at least 35% lower in latency with no added energy consumption. Hence, an improvement of 35% in energy-delay product is also achieved over the state-of-the-art designs. Further investigation on robustness against PVT and corner cases are under investigation.

7 References

1 Hasan, S.R., Pontikakis, B., Savaria, Y.: 'An all-digital skew-adaptive clock scheduling algorithm for heterogeneous multiprocessor systems on chips (MPSoCs)'. ISCAS, 2009, pp. 2501–2504

2 Chen, X., Davis, W.R., Franzon, P.D.: 'Adaptive clock distribution for 3D integrated circuits'. EPEPS, 2011, pp. 91–94

3 Royal, A., Cheung, P.Y.K.: 'Globally asynchronous locally synchronous FPGA architectures'. Field Programmable Logic and Applications, 2003 (*LNCS*, 2778), pp. 355–364

4 ITRS: 'International technology roadmap for semiconductors executive summary'. Available at http://www.itrs.net/Links/2011ITRS/2011Chapters/2011ExecSum.pdf, 2012

5 Dobkin, R., Ginosar, R.: 'Fast universal synchronizers'. PATMOS, 2008, pp. 199–208

6 Verbitskyv, D., Dobkin, R., Ginosar, R.: 'A four-stage mesochronous synchronizer with back-pressure and buffering for short and long range communications'. Technical Report, 2011 Available at http://webee.technion.ac.il/matrics/papers_by_topic.html

7 Dobkin, R., Ginosar, R.: 'Two-phase synchronization with sub-cycle latency', *Integr. VLSI J.*, 2009, **42**, (3), pp. 367–375

8 Dally, W.J., Tell, S.G.: 'The even/odd synchronizer: a fast, all-digital, periodic synchronizer'. Proc. IEEE Symp. Asynchronous Circuits and Systems (ASYNC), 3–6 May 2010, pp. 75–84

9 Alshaikh, M., Kinniment, D., Yakovlev, A.: 'A synchronizer design based on wagging'. Int. Conf. Microelectronics (ICM) 2010, IEEE, December 2010

10 Chabloz, J.M., Hemani, A.: 'Low-latency and low-overhead mesochronous and plesiochronous synchronizers'. Proc. 14th Euromicro Conf. Digital System Design, IEEE, 2011

11 Semiao, J., Varela, J., Freijedo, J., *et al.*: 'Robust solution for synchronous communication among multi clock domains'. APCCAS, 2008, pp. 1107–1110

12 Moore, S.W., Taylor, G.S., Cunningham, P.A., Mullins, R.D., Robinson, P.: 'Using stoppable clocks to safely interface asynchronous and synchronous subsystems'. AINT, 2000, pp. 129–132

13 Dobkin, R., Ginosar, R., Sotiriou, C.: 'Data synchronization issues in GALS SoCs'. ASYNC, 2004, pp. 170–179

14 Mullins, R., Moore, S.: 'Demystifying data-driven and plausible clocking schemes'. ASYNC, 2007, pp. 175–185

15 Seitz, C.L.: 'System timing'. Mead and Conway. Introduction to VLSI Systems' (Addison-Wesley, 1980), Ch. 7

16 Fenghao, M., Svensson, C.: 'Self-tested self-synchronization circuit for mesochronous clocking', *IEEE Trans. Circuits Syst. II, Analog Dig. Signal Process.*, 2001, **48**, (2), pp. 129–140

17 Hasan, S.R., Savaria, Y.: 'Metastability tolerant mesochronous synchronization'. MWSCAS, 2007, pp. 13–16.

18 Rabaey, J.M., Chandrakasan, A., Nikolic, B.: 'Digital integrated circuits; a design perspective' (Prentice-Hall, 2002, 2nd edn.)

19 Sharif, N., Lodhi, F.K., Ramzan, N., Hasan, O., Hasan, S.R.: 'Quantitative analysis of state-of-the-art synchronizers: clock domain crossing perspective'. IEEE Int. Conf. Emerging Technologies (ICET-11), September 2011, pp. 1–6

doi: 10.1049/iet-cds.2013.0067