The Solution of Metastability in Asynchronous System Design Based on FPGA

Lu Lin
Dept. of Mechanical and Electrical Engineering
Beijing Institute of Technology
Beijing, 100081, P. R. China
lulin10902063@bit.edu.cn

Wu Fei
Dept. of Ammunition Professional
Navy in Shenyang Military Representative Office
Shenyang, 110045, P. R. China

Abstract--- In this paper, we will illustrate the mechanism of metastability issues in ASICs designs that are driven by multiple asynchronous clocks. In order to solve this problem, effective synchronization method is described first. Using the proposed design techniques and optimization methods, metastability could be controlled and system reliability could be improved. We provide the structure of the system to show the process of the workflow, and then analyzed the high and low impact on the system. We divided the system into different parts, and designed the schematic for each part to realize the system. After all modules have been completed, we designed an experiment to test the performance of the system. The test results showed that the system is running well which has some practical value.

Keywords- Metastability; Synchronizer; Handshake protocol; Asynchronous FIFO

I. INTRODUCTION

Metastability is unavoidable in asynchronous systems. Whenever asynchronous data is registered by a clocked flip-flop, there is a probability of setup or hold time violation on that flip-flop. In applications such as FPGAs, have defined signal-timing requirements that allow each register to correctly capture data at its input ports and produce an output signal. To ensure reliable operation, the input to a register must be stable for a minimum amount of time before the clock edge (setup time or t_{SU}) and a minimum amount of time after the clock edge (hold time or t_H). The flip-flop output is available after a specified clock-to-output delay (t_{CO}). If the data violates the setup or hold time requirements, the output of the register might go into a metastable state. In a metastable state, the voltage at the register output hovers at a value between the high and low states, which means the output transition to a defined high or low state is delayed beyond the specified t_{CO}. Different destination registers might capture different values for the metastable signal, which can cause the system to fail. In synchronous systems, the input signals must always meet the register timing requirements, so that metastability does not occur. Metastability problems commonly occur when a signal is transferred between circuitry in unrelated or asynchronous clock domains, because the signal can arrive at any time relative to the destination clock.

Although metastability is unavoidable, the probability of occurrence of this issue can be calculated and managed. The MTBF is an estimate of the average time between instances when metastability causes a design failure. The calculated MTBF due to metastability indicates whether designers should take steps to reduce the chance of such failures. This paper explains how MTBF is calculated from various design and device parameters, and how both FPGA vendors and designers can increase the MTBF. System reliability can be improved by reducing the chance of metastability failures with design techniques and optimizations.

In order to minimize the failures due to metastability in asynchronous signal transfers, this paper details the method of using signal synchronization. Synchronization register is typically used for resynchronizing the signal from the destination clock domain to the new clock domain. These registers allow additional time for a potentially metastable signal to resolve to a known value before the signal is used in the rest of the design.

Past studies have shown that the synchronization register has very good results in the solution of signal synchronization. But in today's high speed digital system design, more and more complicated and multiple bus always needs to be used to communicate with more than one other systems simultaneously. For example, TI'DSP TMS320DM642 has UART port, Host Parallel Interface port(16bit or 32 bit), USB port, Ethernet port etc. to exchange data with different peripherals. Therefore only use synchronization register method to eliminate the chances of metastability in high speed circuit is very difficult. To solve this problem, we presents an effective way of using dual synchronization and gray code FIFO to transmit data between different clock domains.

The paper is structured as follows: First, section Ⅱ introduce the reason and hazards during the process of communication between different clock domains, metastability is likely to happen. And then from the study of the key factor of impact metastability----section Ⅲ details the synchronizer failure rate----MTBF. Although the occurrence of metasbility in integrated circuit is unavoidable, we can gain some understanding of what different parts of the brain do. the After a brief introduction of our DARTS architecture in the following section, we will then investigate this circuit's potential for metastability generation and propagation. Next, Section IV will be concerned with the derivation of an analytic model for the circuit and its metastable decay. The propagation of metastability will be studied by means of simulation in Section V. Finally, Section VI concludes the paper and presents future prospects.

During the process of communication between different clock domains, metastability is likely to happen. This paper discusses several applications of different synchronizers in IC asynchronous design. Asynchronous FIFO is applied to the design of asynchronous data buffer between the interface and the core of ATM communication chip. The results of simulation show that the method can increase the reliability as well as realize the desired function.

II. METASTABILITY

When a signal crosses a clock domain, it appears to be an asynchronous signal to the circuitry in the new clock domain. The circuit that receives this signal needs to synchronize it. Synchronization prevents the metastable state of the first storage element (flip-flop) in the new clock domain from propagating through the circuit.

Metastability is the inability of a flip-flop to arrive at a known state in a specific amount of time. When a flip-flop enters a metastable state, a designer cannot predict the output voltage level or when the output will settle to a correct voltage level. During this settling time, the flip-flop's output is at some intermediate voltage level or may oscillate and can cause a cascade of failures when the flip-flops further down the signal path capture the invalid output level.

A Metastable causes

For any flip-flop, there is a small window of time where the input must be stable (Figure 1: Stable Window). This window of time is a function of the design of the flip-flop, the implementation technology, operating conditions and the load on the output for outputs not buffered. Also sharp edge rates on the input signal minimize the window of time. The probability of a flip-flop entering a metastable state is also a function of the data and clock frequencies. There are more windows of vulnerability as the clock frequency goes up and there is greater probability of hitting the window as the data frequency goes up.

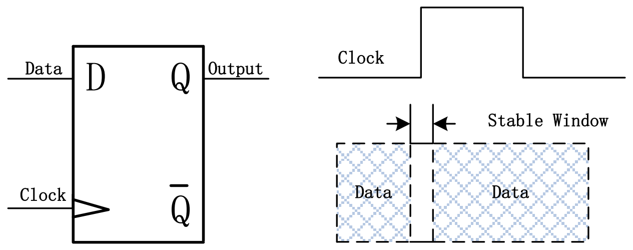

Figure 1. Stable Window

Past studies have shown that the flip-flop delay in the metastable region is exponential in nature where two parameters can be extracted from simulation to model the behavior of the delay in the metastable region [5]. Metastability window, given by Equation (1), is a common metric used to analyze metastability.

Here's a brief analysis. When the circuit is in a metastable state, enter '1 'or '0' may be sentenced to '0 'or '1', this state is called a flip (upset).The mathematical expression of the flip probability P as follows:

$$P = T_0 \exp \frac{-T_\tau}{\tau_C} \qquad (1)$$

It represents the period where data transition will not be resolved within a given settling time (s), which is related to the clock frequency and represents the amount of time given for the output to settle to a stable state. T_0 is the width of the metastability window with no resolution time, and is the resolution time constant that represents the inverse of the gain-bandwidth product of the feedback element in the flip-flop. As seen from Equation (1), the time of metastability has the greatest impact on due to the exponential relationship. A small value results in fast resolving time from the metastable region and thus decreases.

As we have seen that whenever setup and hold violation time occurs, metastability occurs, so we have to see when signals violate this timing requirement:

- When the input signal is an asynchronous signal.
- When the clock skew/slew is too much (rise and fall time are more than the tolerable values).
- When interfacing two domains operating at two different frequencies or at the same frequency but with different phase.
- When the combinational delay is such that flip-flop data input changes in the critical window (setup+hold window)

B Mean Time Between Failures

FPGA manufacturers and IC foundries claimed that they have qualified their flip-flops and determine their characteristics. However, if the restriction of a minimum separation between input edges is lifted, then two edges very close together will not be seen by the circuit and two edges much further apart will count as two. In this situation metastable behavior is unavoidalle due to the continuum of edge interarrival times possible between no effect and two counts.

Since the occurrence of metastable behavior is inevitable, We can effectively control probability of this phenomenon from the analysis above mean time between two successive failures. MTBF(Mean Time Between Failures) describes the metastability characteristic of a flip-flop using statistics to determine the probability of a flip-flop failure. The MTBF is based in part on the length of the time window during which a change in the input signal causes the flip-flop to become unstable. In addition, MTBF calculation uses the frequency of the input signal and the frequency of the clock driving the flip-flop. The industry standard formula for Mean Time Between Failures (MTBF) for a metastable flip-flop is given by

$$\mathrm{MTBF} = \frac{1}{\mathrm{P} f_{clock} f_{data}} = \frac{\exp\frac{T_\tau}{\tau_C}}{T_0 f_{clock} f_{data}} \qquad (2)$$

where:

- exp = 2.718281828...
- τ_C= time delay for the metastability to resolve itself
- f_{clock} = the clocking frequency
- f_{data} = the data frequency
- T_0 = a constant representing the metastability catching setup time window
- T_τ = a constant describing the speed with which the metastable condition is resolved

The variables in the expression are functions of the flip-flop design, its process technology, the clocking rate, and the data switching speed, which are discussed following sections.

III. METASTABLE AND SYSTEM RELIABILITY

Synchronization circuit, the metastable still could happen, connected with this is the mean time between failures MTBF (mean time between failure), the probability of occurrence of the metastable and independent of the clock frequency, but closely related to the MTBF and clock. Article provides an example of a 20MHz clock work under MTBF is about 50 years, but the clock frequency to 40MHz, MTBF only 1 minute! Seen to reduce the clock frequency can be greatly reduced metastable lead to system errors appear, the reason is that, if the clock cycle than the resolution time can be reduced to the metastable state is passed to the next level of opportunity to improve system MTBF shown in Fig .2.

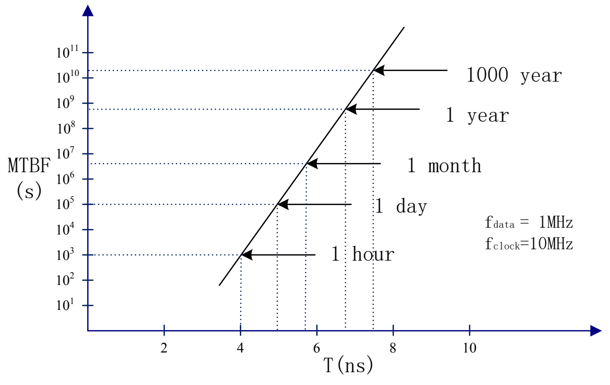

Figure 2.

IV. COMMON SOLUTION

IC foundries help with signal synchronization by providing specially designed synchronizer cells. Usually this synchronizer cells consists of a flip-flop with a very high gain that uses more power and is larger than a standard flip-flop. This flip-flop has reduced setup and hold time requirements for the input signal and is resistant to oscillating when input signal causes a metastable condition. Another type of synchronizer cell contains two flip-flops, which eases the layout engineerís job by meeting the requirement of placing the flip-flops close to each other and prevents the designer from placing any combinational logic between the flip-flops.

Thus, the synchronizer for high-speed digital circuits, usually take the strategy is to trigger cascade time buffer to provide time for the circuit to recover from the metastable state, that the time delay to reduce the metastable impact on the circuit has occurred.

A Flip-Flop Synchronizer

Synchronizing signals begins by protecting downstream logic from the metastable state of the first flip-flop in a new clock domain. A simple synchronizer consists of two flip-flops in series without combinational circuitry between them (see Figure 1-4: A Simple Synchronizer). This design ensures the first flip-flop exits its metastable state and its output settles before the second flip-flop samples the first oneís output. Besides the circuit design, there is another requirement to make a successful synchronizer. The layout engineer needs to place the flip-flops close to each other. This guarantees the shortest signal wire between the output of the first flip-flop and the input of the second one and ensures the smallest possible clock skew between the flip-flops. There are many different designs for synchronizers and each has specific uses because one type does not work well in all applications. All synchronizers fall into three basic categories: edge-detect and pulse.

B The Edge Synchronizer

As shown in Fig .3, the edge detection synchronous increase in the level synchronizer output a trigger. New trigger output by the output of the inverting and level synchronizer operation. This circuit detects the rising edge of the synchronous input signal, resulting in a synchronized clock cycle wide active high pulse. If the two input AND gate is used interchangeably, can constitute a detection of the falling edge of the input signal synchronizer. AND gate NAND gate instead, you can build a circuit to generate an active-low pulse.

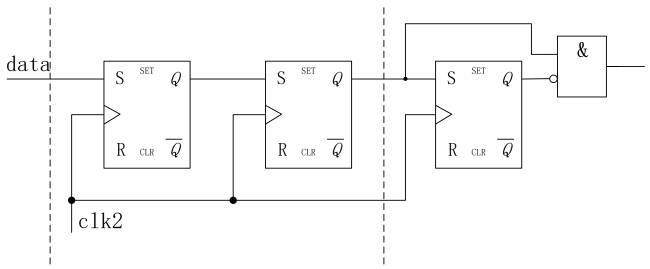

Figure 3. Edge-Detect Synchronizer

The edge-detect synchronizer’s main application is synchronizing a pulse going to a faster clock domain. This circuit produces a pulse that indicates the rising (or falling) edge of the input signal. A restriction on the application of this synchronizer is the width of the input pulse must be greater than the period of the synchronizer clock plus the required hold time of the first synchronizer flip-flop. The safest pulse width is twice the synchronizer clock period. This synchronizer does not work if the input is a single clock-wide pulse going to a slower clock domain; however, the pulse synchronizer solves this problem.

C The Pulse Synchronizer

As shown in Fig .4, the pulse synchronizer input signal is a single clock width pulse, which triggers a flip circuit of the original clock domain. Whenever the flip circuit receives a pulse, it will be converted in the high, low, and then through thelevel synchronization to reach the input of the XOR gate, while another signal by one clock cycle delay into different conversion time state of the door the other side, flip the circuit, the output of the synchronizer to produce a single clock pulse width.

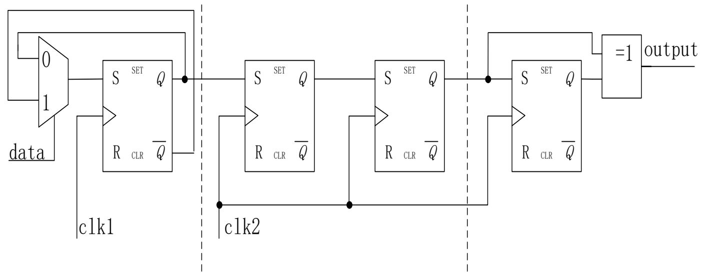

Figure 4. Pulse Synchronizer

The basic function of a pulse synchronizer is to take a single clock wide pulse from one clock domain and create a single clock wide pulse in the new domain. One restriction on this synchronizer design is the input pulses must have a minimum time between them. This minimum spacing between the pulses is equal to two synchronizer clock periods. If the input pulses are closer, the output pulses in the new clock domain are adjacent to each other resulting in an output pulse that is wider than one clock cycle. This is a more severe problem when the clock period of input pulse is greater than twice the synchronizer clock period. In this case, if the input pulses are too close, the synchronizer does not detect every one.

D FIFO synchronizer

If we use two Flip-Flop synchronizers to synchronize the system, this probability can be reduced to an acceptable range. Another problem is the design of the FIFO address generator. Although binary counters work fine for addressing the memory trying to synchronize binary counters into a new clock domain is problematic. The binary counter changes more than one bit at the same time and every bit change at different time. The address which is sampled by the synchronous clock may be different with its true value. If we use this pointer to do the comparison work and produce full and empty flag which is unadvisable. A better approach for passing pointers between clock domains is to use a gray-code counter for the two FIFO pointers.

The Gray Code counter is a binary adder with converters from and to Gray Code before and after the adder. Since converting to and from Gray Code is a XOR operation, this counter design has only two more levels of logic than a binary counter. A design can use the same technique to compare Gray Code pointer values by adding converters between the pointers and binary comparison logic.

This FIFO status technique gives pessimistic status for both reads and writes. The status on the write side indicates full when the FIFO fills and continues to indicate full after it is read since synchronization delays the read pointer to the write-side comparison logic. This is also true for the empty status on the read side since synchronization delays the write pointer to the read-side comparison logic.

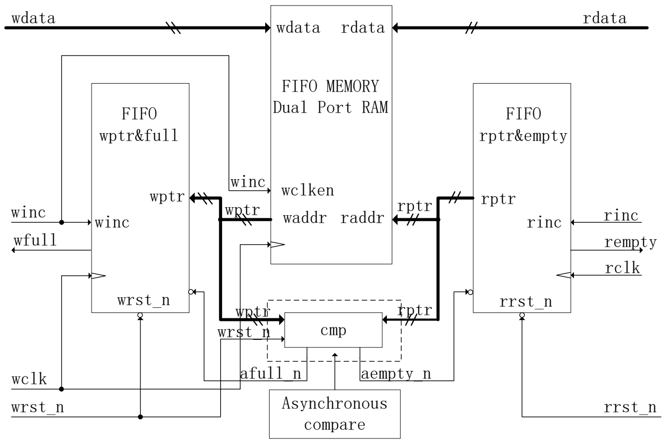

Figure 5. Pulse Synchronizer

Fig .5 shows the general junction asynchronous FIFO Institutions. How to correctly determine the FIFO empty / full status is the key to the design of the FIFO. In order to effectively improve the empty / full flag generation logic reliability in set in total, can be read / write address is divided into counts and internal address by the former to produce the empty / full flag, the latter as a dual-port RAM read / Write address, thereby reducing the synchronization signal, reduce the number of signals Asia Steady state production. To count the address of the synchronization, it is recommended using Gray Code instead of binary code. Gray code is a time to transform a significantly drop Low number of address signal transition to eliminate the count of the address bus Bit different synchronization when the competition. Its synchronization, at most An address data into the metastable state, and therefore further reduce the probability of the metastable For the circuit delay consideration in the design of the empty full flag to generate Logic, it is best to let the state of the FIFO circuit in the FIFO is about to read Send empty or filled with empty / full flag in order to effectively guarantee the FIFO not low or overflow. Asynchronous FIFO design by raising the empty / full The flag logic reliability, can inhibit the metastable state.

V. THE SIMULATION RESULTS

Verilog-HDL and Altera Cyclone EP2C35F484 FPGA are used to design the circuit. QuartusII and Modelsim SE are used as developing enviament. In the experiment we set the data width as 32 bit and the address depth 4bit.

As shown in Fig .6,Fig .7, respectively, run under Modelsim simulation of the two sets of edge detection synchronization and pulse synchronizer. The following brief account. clk1 is the clock of the primary circuit; clk2 synchronization clock; pulse_din input pulse synchronizer pulse_dout its output; edge_din input synchronizer edge detection, edge_dout its output. Figure 4 shows the synchronization input and output of the edge detection synchronization and pulse under normal conditions. In Figure 5, the pulse synchronizer input (pulse_din), pulse interval is too small, the synchronizer can not distinguish between the two input pulses, can only output (pulse_dout the) a synchronization clock cycle the width of the pulse.

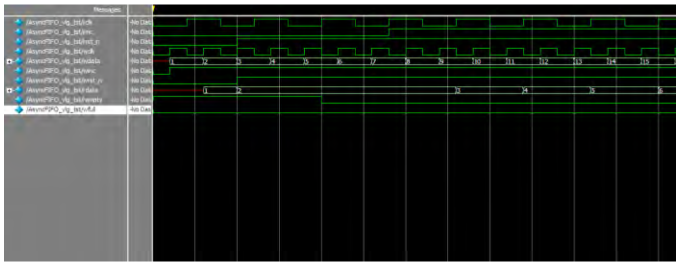

Figure 6. Pulse Synchronizer

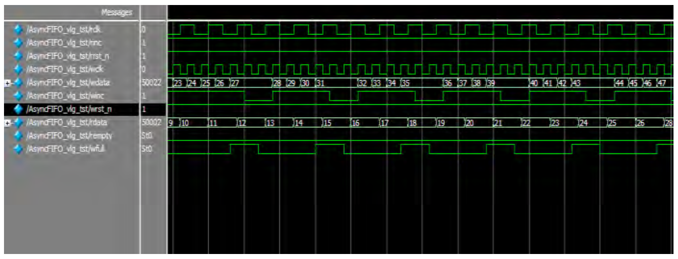

Figure 7. Pulse Synchronizer

We can see that only if the empty flag is invalid the reading address will increase. Otherwise the reading pointer keeps itself invariable until the empty flag is invalid. The reading enable will valid as soon as the full flag is invalid. Fig .7 shows the simulation results when the writing clock is faster than the writing clock. If the full flag is valid, the writing pointer has no change and the writing enable will be put down.

VI. CONCLUSION

The main causes of metastability, timing devices within the sampling window, and can not guarantee that the input signal is always maintained at a stable level. Therefore, we need by reducing the sampling window to increase the sampling success rate (using the edge of the trigger for level trigger, but also a way to reduce the sampling window), or by the stability window of the sampling window or input data move to ensure that sampling to resolve the metastability issue. But the participation of the asynchronous signal, the solution will become extremely complex. Therefore, there is not a universal, effective and feasible solution. However, the mechanism of the formation of metastable, allows us to ease from the engineering point of view to solve specific metastability may occur in the actual project.

REFERENCES

[1] Clifford E. Cummings, “Synthesis and Scripting Techniques for Designing Multi-Asynchronous Clock Designs,” SNUG 2001 (Synopsys Users Group Conference, San Jose, CA, 2010) User Papers, March 2001, Section MC1, 3rd paper. Also available at www.sunburst-design.com/papers.

[2] Managing metastability with the Quartus II Software[Z]. Altera, Quartus II Handbook Version 9. 1Volume: Design and Synthesis, 2009:1-7-15.

[3] KILTS S. Advanced FPGA Design: Architecture,Implementation, and Optimization [M] . New Jersey:John Wiley & Sons, Inc, 2007: 84 -97.

[4] Dinesh Tyagi, former CAE Manager for Synopsys DesignWare product, personal communication Edward Paluch, personal communication Frank Gray, "Pulse Code Communication." United States Patent Number 2,632,058. March 17, 1953.

[5] John O’Malley, Introduction to the Digital Computer, Holt, Rinehart and Winston, Inc., 2011, pg. 190.

[6] L. Kleeman and A. Cantoni, “Metastable behavior in digital systems.” IEEE Design and Test of Computers. Vol. 4, No.6, pp. 4-19, Dec. 1987.

[7] A.Scheibe, W.A.Krauss, “two-transistor SIMOS EAROM cell”. IEEE Journal of Solid-State Circuits. Vol.15,No.3,pp. 353-357 , Nov,2010.

[8] S.Brown, J .Rose, “FPGA and CPLD architectures. A Tutorial”. IEEE Design and Test of Computers, Vol.12, No.2, pp. 42-57,May.2011.

[9] J.Birkner, A.Chan, “A very-high-speed field-programmable gate array using metalto-metal antifuse programmable elements”. Microelectronics Journal. Vol.23, No.7, pp. 561-568, Dec,2010.

[10] L .O .Chua, “Memristor—the missing circuit element”. IEEE Transactions on Circuit Theory Vol.18, No.5, p.507-519, Feb,2009.