Mohit Arora

The Art of Hardware Architecture

Design Methods and Techniques for Digital Circuits

Springer

The Art of Hardware Architecture

Mohit Arora

The Art of Hardware Architecture

Design Methods and Techniques for Digital Circuits

Springer

Mohit Arora
Freescale Semiconductor
Faridabad
India
mohit.arora@me.com

ISBN 978-1-4614-0396-8 e-ISBN 978-1-4614-0397-5
DOI 10.1007/978-1-4614-0397-5
Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2011934353

This book is dedicated to my wife Pooja and my daughter Prisha.

Preface

I started my Industry career in year 2000 in the field of Chip Design. My work involved lot of research that gave me opportunity to write technical papers, participate in various conferences and share practical experiences. During this journey I got lot of positive feedback on my publications. Readers have often asked me forcing me to think if I should write a book compiling all the practical experiences. The book's aim is to highlight all the complex issues, tasks and skills that must be mastered by an IP designer to design an optimized and robust digital circuit to solve a problem. The techniques and methodologies prescribed in the book, if properly employed, can significantly reduce the time it takes to convert initial ideas and concepts into right-first-time silicon.

The book is intended for a wide audience. Though it may be used in an undergraduate or graduate course, book is mainly intended for those in semiconductor industries who are directly involved with chip design and requires deeper understand of the subject.

This book is distinguished from others by its primary focus on real problems rather than theoretical concepts with its emphasis on design techniques across various aspects of chip-design.

The book covers aspects of chip design in a consistent way, starting with basic concepts in Chap. 1 and gradually increasing the depth to reach advanced concepts, such as EMC design techniques or sophisticated low power techniques like DVFS (Dynamic Voltage and Frequency scaling).

Chapter 1 covers "*metastability"* to help user understand more clearly the issues related to metastability, how it can be quantified and necessary techniques to minimize its effort.

Chapter 2 covers general set of recommendations around "*clocks and resets*" intended for use by designers while designing a block or Intellectual Property (IP). The guidelines are independent of any CAD tool or silicon process and are applicable to any ASIC designs.

Chapter 3 goes beyond synchronous clock designs and covers asynchronous clocks or "*handling multiple clocks*" in design, problems faced and solutions in order to get a robust designs that works on multiple clocks.

Chapter 4 covers all about "*Clock Dividers*" that a typical SoC may require generating number of phase related clocks. Apart from synchronous division where required clocks are generated by dividing the master clock by a power of two, chapter also covers odd division (Divide by 3, 5 etc.) as well as non-integer dividers (Divide by 1.5, 2.5 etc.).

Chapter 5 covers all about "*Low Power Design techniques*". In recent times, power consumption has become a significant design constraint with shrinking technology as well as to meet power targets for energy efficient applications. This Chapter describes various design methodologies and techniques at various levels of design abstraction to reduce dynamic and as well as static power consumption.

Chapter 6 covers the concept of "*Pipelining*", the way it applies to processor design to increase the throughput in terms of calculations per clock cycle. The chapter extends the scope of pipelining beyond microprocessor to cover typical circuits so as to increase performance.

Chapter 7 covers "*Endianess issues*" in design that may include several third-party IPs with different Endianess and the way it can be handled in the design in an optimal way.

Chapter 8 covers several hardware as well as software "*Deboucing Techniques*" to eliminate unwanted noise or glitch in the circuit caused by an external input (usually some kind of switch).

Chapter 9 covers deep details on EMC/EMI issues, the way it applies to digital circuits and design guidelines that can be followed at various level of abstraction for *"better EMC performance*".

Theoretical part has been intentionally kept to the minimum that is essentially required to understand the subject. The guidelines explained across various chapters are independent of any CAD tool or silicon process and are applicable to any ASIC designs and can help designers to plan and to execute a successful System on Chip (SoC) with a well-structured and synthesizable RTL code.

There are few chapters that include Verilog Hardware Description Language (HDL) code for beginner's who are learning digital circuits, however the same can be skipped by more advanced engineers who are already exposed to the fundamentals.

Some of the more advanced chapters like "*Design Guidelines for EMC performance*" have been thoroughly researched and have taken months to write in a way to make topics more relevant to digital designers.

Every possible effort was made to make the book self-contained. Any feedback/comments are welcome on this aspect or any other related aspects. Comments can be sent to me at the following mails: mohit.arora@me.com or mohit.arora@freescale.com.

Faridabad, India Mohit Arora

Acknowledgements

The original idea behind "*The Art of Hardware Architecture*" was to link my years of experience as a design architect with the extensive research I have conducted. However, achieving the final shape of this book would not have been possible without many contributions.

My sweet wife *Pooja* was so patient with my late nights, and I want to thank her for her faithful support & encouragement in writing this book. Most of the work occurred on weekends, nights, while on vacation, and other times inconvenient to my family. I like to thank my parents for allowing me to follow my ambitions throughout my childhood.

I am grateful to *Prashant Bhargava,* my colleague from Freescale Semiconductor for his careful reading of drafts of this book and his constructive suggestions for improvement based on his years of experience. He also helped in book editing, formatting as well contribution to some of the sections in Chap. 7.

Special thanks to *Rakesh Pandey* and *Sudhi Ranjan Proch* from Freescale Semiconductor for their early help and feedback on some of the sections of the book.

I thank *Charles Glaser*, Senior Editor from Springer for providing me opportunity to publish with Springer and entire publication team at *Springer* who helped turn this book into excellent professional manuscript.

Mohit Arora

Contents

Chapter 1
The World of Metastability

1.1 Introduction

In a synchronous system, the data always has a fixed relationship with respect to the clock. When that relationship obeys the setup and hold requirements for the device, the output goes to a valid state within its specified propagation delay time. In synchronous systems, the input signals always meet the flip-flop's timing requirements; therefore, metastability does not occur. However, in an asynchronous system, the relationship between data and clock is not fixed; therefore, occasional violations of setup and hold times can occur. When this happens, the output may go to an intermediate level between its two valid states and remain there for an indefinite amount of time before resolving itself or it may simply be delayed before making a normal transition.

This Chapter is intended to help understand more clearly the issues relating to the metastability, how it is quantified, and how to minimize its effort.

1.2 Theory of Metastability

Metastability arises as a result of violation of setup and hold times of a flip flop. Every flip-flop that is used in any design has a specified setup and hold time, or the time during which the data input is not legally permitted to change before and after a rising clock edge, respectively. If the signal does change during this time window, the output will be unknown or "metastable". This propagation of unwanted state is called Metastability. As a result the output of a flip-flop can produce a glitch or remain temporarily in metastable state, thus taking longer to return to stable state.

When a flip-flop is in a metastable state, the output hovers at a voltage level between high and low, causing the output transition to be delayed beyond the specified clock-to-output delay (t_{co}). The additional time beyond t_{co} that a metastable output takes to resolve to a stable state is called the settling time (t_{MET}). This has

M. Arora, *The Art of Hardware Architecture: Design Methods and Techniques for Digital Circuits*, DOI 10.1007/978-1-4614-0397-5_1,

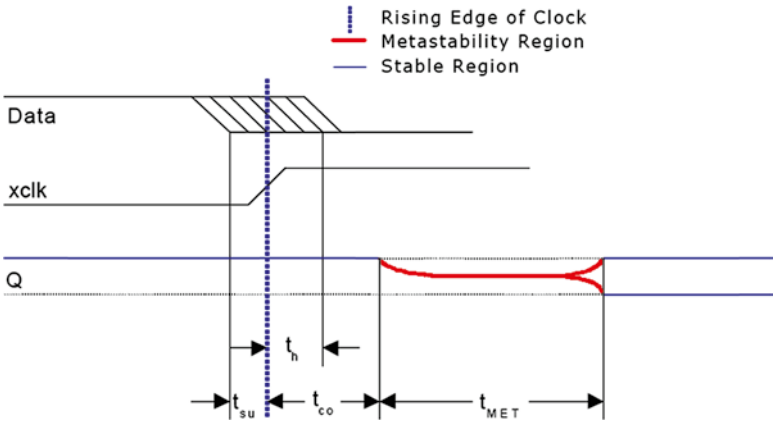

Fig. 1.1 Metastability timing parameters

been shown in Fig. 1.1. Not every transition that violates the setup or hold times results in a metastable output. The likelihood that a flip-flop enters a metastable state and the time required to return to stable state depends on the process technology used to manufacture the device and on the ambient conditions. Generally, flip-flops will return to a stable state within one or two clock cycles.

The operation of a Flip-Flop is analogous to a ball rolling over a frictionless hill, as shown in Fig. 1.2. Each side of the hill represents a stable state (i.e. high or low) and the top represents a metastable state. Suppose the ball is in a stable state (i.e. either 1 or 0) and a push (state transition) is given to the ball that is sufficient (no setup or hold time violations) enough to make the ball cross over to the other stable state, the ball crosses to the other stable state within the specified time.

However, if the push is less (i.e. violation of setup and hold time), the ball shall travel to the top of the hill (i.e. output metastable), stay there for some time and return to either stable state (i.e. output becomes stable eventually). It may also happen that the ball may rise partially and come back (i.e. output may produce some glitches). Either condition increases the delay from clock transition to a stable output.

Thus, in simple words, when a signal is changing in one clock domain (*src_data_out*) and is sampled in another clock domain (*dest_data_in*), then this causes the output to become metastable. This is known as Synchronization Failure (Shown in Fig. 1.3).

1.3 Metastability Window

Metastability Window is defined as the specific length of time, during which both the data and clock should not change. If both signals do occur, the output may go metastable. As shown in Fig. 1.4, the combination of the Setup and Hold time determine the width of the Metastability window.

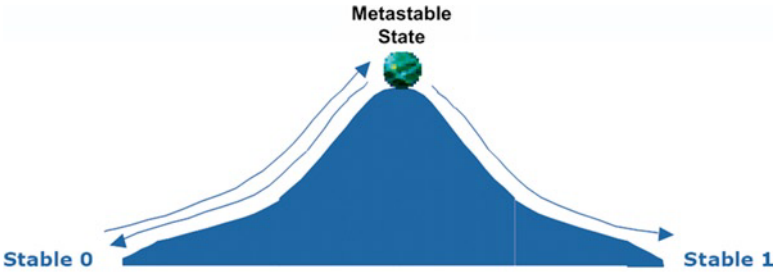

Fig. 1.2 Metastable behavior of flip flop

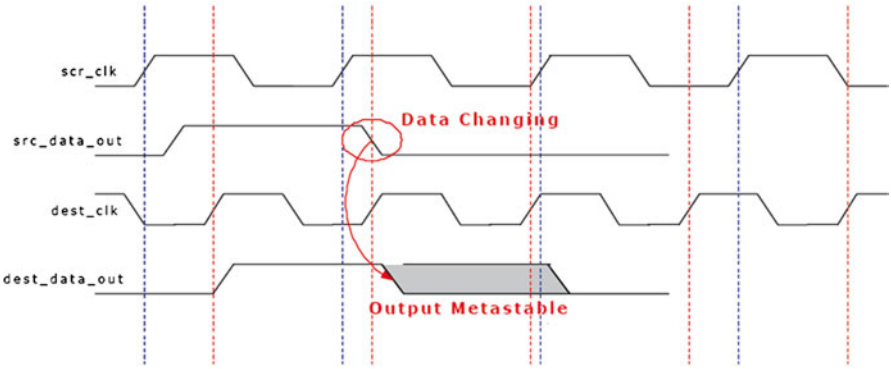

Fig. 1.3 Metastability in flip flop

The larger the window, the greater the chance the device will go Metastable. In most cases, newer logic families have smaller Metastability windows which reduce the chance of the device going Metastable.

1.4 Calculating MTBF

Mean (Average) Time Between Failures or MTBF of a system, is the reciprocal of the failure rate in the special case when the failure rate is constant. This gives the information on how often a particular Flip Flop will fail.

For a single-stage synchronizer with a given clock frequency and an asynchronous data edge that has a uniform probability density within the clock period, the rate of generation of metastable events can be calculated by taking the ratio of the setup and hold time window to the time between clock edges and multiplying by the data edge frequency.

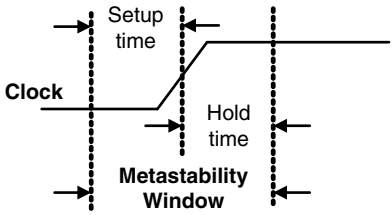

Fig. 1.4 Metastability window

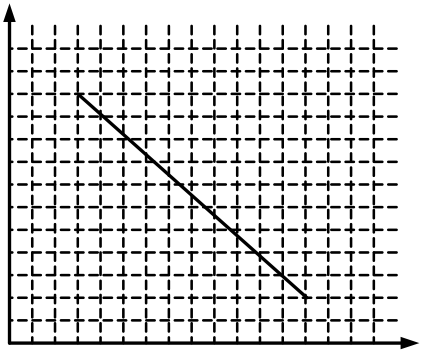

Fig. 1.5 Failure rate vs. time (log scale)

$$\frac{1}{FailureRate} = \text{MTBF}_1 = \frac{e(t_r/\tau)}{W \times f_c \times f_d} \tag{1.1}$$

where

t_r = resolve time allowed in excess of the normal propagation delay time of the device
τ = metastability (resolving) time constant for a flip-flop
W = Metastability Window
f_c = Clock frequency
f_d = Asynchronous data edge frequency

The constants W and τ are related to electrical characteristics of the device and may vary according to the process technology node. Therefore, different devices manufactured with the same process have similar values for W and τ.

If the failure rate of a device is measured at different resolve times and plotted, the result is an exponentially decaying curve. When plotted on a semi logarithmic scale, as shown in Fig. 1.5, this becomes a straight line the slope of which is equal to τ; therefore, two data points on the line are sufficient to calculate the value of τ using Eq. 1.2.

$$\tau = \frac{t_{r2} - t_{r1}}{\mathrm{In}(N1/N2)} \tag{1.2}$$

where

t_{r1} = resolve time 1
t_{r2} = resolve time 2
N1 = numbers of failures at tr1
N2 = numbers of failures at tr2

Based on Eqs. 1.1 and 1.2, MTBF for a two-stage synchronizer can be calculated by Eq. 1.3 below

$$\mathrm{MTBF}_2 = \frac{e(t_{r1}/\tau)}{W \times f_c \times f_d} \times e(t_{r2}/\tau) \tag{1.3}$$

where

t_{r1} = resolve time allowed for the first stage of synchronizer
t_{r2} = resolve time in access of normal propagation delay

The first term in the Eq. 1.3 calculates the MTBF of the first stage of the synchronizer, which in effect becomes the generation rate of the metastable events for the next stage. The second term then calculates the probability that the metastable event will be resolved based on the value of t_{r2}, the resolve time allowed external to the synchronizer. The product of the two terms gives the overall MTBF for the two-stage synchronizer.

In quantitative terms, using Eq. 1.3 above , if the Mean Time Between Failure (MTBF) of a particular Flip-Flop in the context of a given clock rate and the input transition rate is 40 s then MTBF of two such flip-flops used to synchronize the input would be 40 × 40 = 26.6 min.

1.5 Avoiding Metastability

As shown in Sect. 1.2, metastability occurs whenever setup or hold time is violated. So signals may violate the timing requirements under the following conditions:

- When the input signal is an asynchronous signal.
- When the clock skew/slew (rise/fall times) is higher than the tolerable limit.
- When signals cross the domains working at two different frequencies or with same frequency but different phase and skew.
- When the combinational delay is such that the Flip Flop data input changes in the Metastability Window.

Metastability can cause excessive propagation delays and subsequent system failures. All Flip Flops and latches exhibit metastability. The problem cannot be eliminated. But it is possible to make metastability less likely to occur.

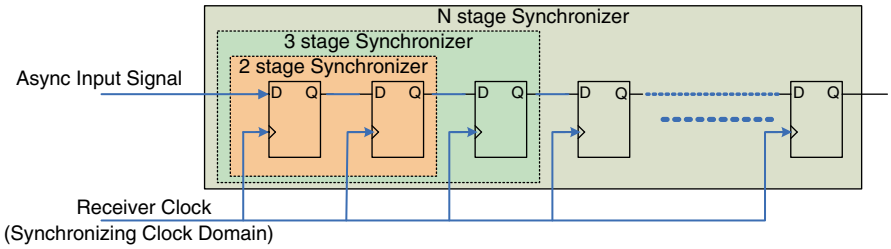

Fig. 1.6 N-stage synchronizer

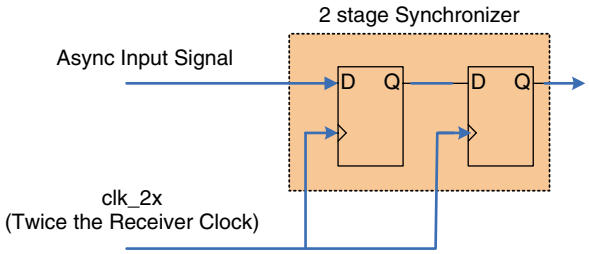

Fig. 1.7 Multi-stage synchronizer with clock boost circuitry

In the simplest case, designers can avoid metastability by making sure the clock period is long enough to allow for the resolution of quasi-stable states and for the delay of whatever logic may be in the path to the next flip-flop. This approach, while simple, is rarely practical given the performance requirements of most modern designs. The other approach is to use Synchronizers.

1.5.1 Using a Multi-stage Synchronizer

The most common way to avoid metastability is to add one or more synchronizing flip-flops at the signals that move from one clock domain to the other as shown in Fig. 1.6. This approach allows for an entire clock period (except for the setup time of the second flip-flop) for metastable events in the first synchronizing flip-flop to resolve itself. This does however; increase the latency in the synchronous logic's observation of input.

1.5.2 Multi-stage Synchronizer Using Clock Boost Circuitry

One limitation of the multiple-stage synchronizer is that it takes longer for the system to respond to an asynchronous input. A solution to this problem is to use the output of a clock doubler to clock the two synchronizing flip-flops. Altera's FPGA exhibit this technique as Clock Boost or Clock Doubler (Fig. 1.7).

This approach allows the system to respond to an asynchronous input within one system clock cycle, while still improving MTBF. Although the Clock Boost clock could decrease the MTBF, this effect is more than offset by the two synchronizing flip-flops.

Neither of these approaches can guarantee that metastability cannot pass through the synchronizer; they simply reduce the probability of occurrence of metastability.

1.6 Metastability Test Circuitry

Whenever a flip-flop samples an asynchronous input, a small probability exists that the flip-flop output will exhibit an unpredictable delay. This happens not only when the input transition violates setup and hold time specifications, but also when the transition actually occurs within a small timing window during which the flip-flop accepts the new input. Under these circumstances, the flip-flop can enter a metastable state.

The test circuit described in Fig. 1.8 is used to determine metastability characteristics of a Flip-Flop. Figure 1.8 shows an Asynchronous Input "*async_In*" to the Flip Flop "FF_A" triggered on positive edge of Clock "*clk*". As shown both the Flops "FF_B" and "FF_C" are triggered on negative edge of the clock in order to capture the metastable event on "FF_A".

As complementary signals are passed on the input of Flip Flops "FF_B" and "FF_C", the output of the XNOR gate goes HIGH whenever a metastable event occurs on "FF_A". This conditions is captured on output of Flip Flop "FF_D" indicating that a metastable event has been detected.

The timing for all the nodes in this test circuit is shown in Fig. 1.9.

Because the resolving flip-flops ("FF_B" and "FF_C") are clocked by the falling clock edge, the required settling time can be controlled by changing the clock high time (Δt). The settling time t_{MET} can be determined with the equation below

$$t_{MET} = \Delta t - t_{ACN} \tag{1.4}$$

where t_{ACN} is the minimum clock period which is equal to t_{CQ} (clock to Output delay of FF_A) + setup time t_{su} of the resolving Flip Flop (FF_B or FF_C).

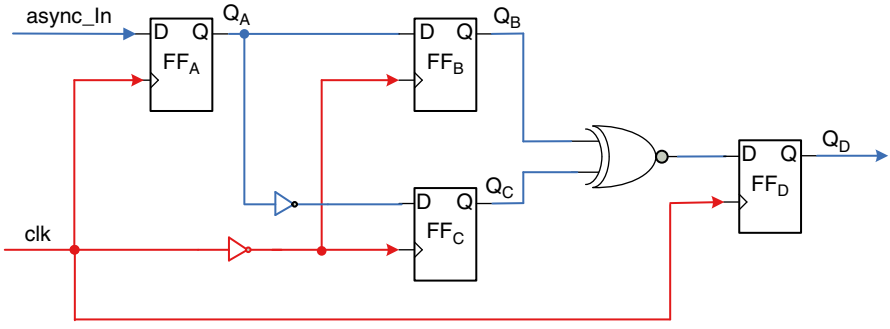

Fig. 1.8 Metastability test circuitry