Basics of multi-cycle & false paths

Nitin Singh , Neha Agarwal & Arjun Pal Chowdhury -August 07, 2014

1 Introduction

One of the significant challenges to RTL designers is to identify complete timing exceptions upfront. This becomes an iterative process in complicated designs where additional timing exceptions are identified based upon critical path or failing path analysis from timing reports. This approach leaves a significant number of timing paths which may not be real, but these never get identified, since they may not come up in the critical path report. However, synthesis and timing tools will continue to expend resources optimizing these paths when it is not needed. At the same time, it can also impact area and power consumption of the device.

The intent of this document is to provide examples of false and multi cycle path exceptions that are easily missed by even experienced designers, and are identified through iterations on timing reports.

1.1 False Path

False Paths are those timing arcs in design where changes in source registers are not expected to get captured by the destination register within a particular time interval. False Paths can be categorized in the following design topologies.

1.1.1 Static False Path

Static false path are those timing arc in design where excitation of source register will not have any impact or change in destination register. The timing path in these topologies can't be sensitized by any input vector even if both source and destination flops are running on same clock domains.

- EXAMPLE – 1

  The following topology depicts an example of static false path. Here change in D1.Q will never cause any change in D4.D. The value of D4 flop will always be governed by the value of D2. Hence for this particular circuit D1->D4 can be treated as false path.

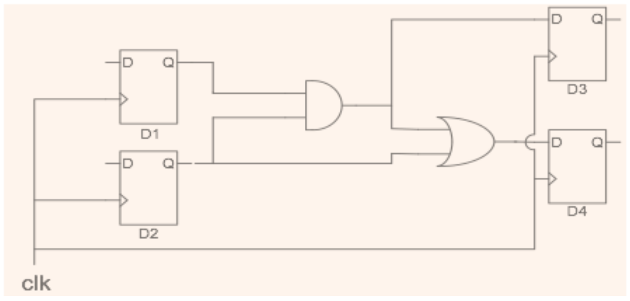

Figure 1: Static timing path

- EXAMPLE - 2

In a system a control bit is defined such that if the bit is SET, the clock going to DMA controller will remain enabled in Low Power (STOP) mode, otherwise the clock will be gated in STOP mode.

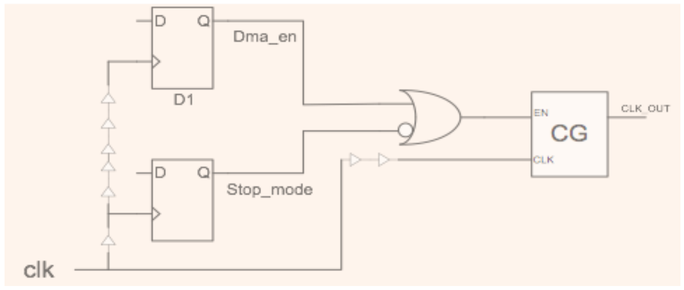

Figure 2: DMA Clock gating

In the above circuit, when the system is in RUN mode, the clock gating cell will always remain enabled and any change in dma_en control register won't affect the clock gating enable generation logic. User is supposed to write/change to this control register in RUN mode prior to enter low power STOP mode. Once the system enters stop mode the pre programmed value of dma_en register will govern the state of the clock gating cell.The single cycle timing path is the one generated from "stop_mode_reg -> CG_cell". The path "dma_en_reg -> CG_cell" is a false path. The clock skew and cell placement in this topology could cause timing problems even if there is very small combinational delay between the flip flops dma_en_reg->CG cell. Giving the right exceptions to the timing tool will help optimize the cell placement with in the first iteration.

- EXAMPLE-3

IO output timing arcs are usually critical and is a limiting factor to decide the maximum baud rate support of a synchronous protocol interface. It becomes important to identify the proper REG->OUT timing arc and isolate the invalid REG->OUT path by constraining them as False Path. It can save lot of iteration time and efforts put by timing and placement tools for meeting the IO-Spec of synchronous interfaces.

Usually, before starting any data transaction the protocol modules need to be configured properly. These configuration registers are considered to be static during any data transaction. However timing arc could exist from these configuration registers to output pads. This is another example where upfront design constraints can help ease the timing tool's task to meet the target frequency.

In the topology given below, the configuration registers (config1, config2 and config3) are supposed to be programmed with a particular value prior to making any data transaction. When the data transaction has been initiated, the valid timing arc would be shift register->combo3->IO buffer-> PAD. However the timing arc originated from config1/2/3 and terminated at PAD can easily be disabled by putting false path exceptions.

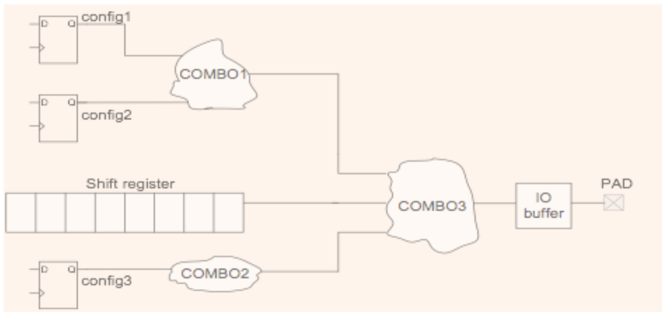

Figure 3: I/O path from configuration registers

However the above exceptions need to be reviewed properly to avoid any uncovered corner case. For example: In a half duplex communication or memory interface the data direction can change on the fly. So it may require to meet the timing from “data direction register-> PAD” and the path can’t be considered as false path.

1.1.2 False Reset Timing Arc

During device start-up it is not required all the application modules of a device to remain enabled. Hence by default, the clock to those modules is gated. During system reset de-assertion sequence the reset de-assertion to those modules happened in absence of the clock. As there is no clock, so no chance of metastability due to reset de-assertion timing violation. Hence asynchronous reset recovery/de-assertion paths to those modules can be considered as false path.

1.1.3 Asynchronous False Path (CDC path)

If the clock domain of the source register is asynchronous to the clock domain of the destination register then the path is considered as asynchronous or Clock Domain Crossing path. It is not possible to have any timing correlation in these paths because there is no defined relationship between the clock edges of the launching and capturing domain. These paths can be treated as false path for timing analysis. In this case, it is the responsibility of the designer to avoid any occurrence of setup/hold violation at capturing domain registers. In this section some popular synchronization techniques are discussed which helps to avoid the occurrence of metastability in designs with asynchronous clocks. Designers should provide these exceptions to the timing tool before synthesis. Sometimes it may happen that an asynchronous design is used with synchronous clocks in SOC integration. In such cases, these exceptions are particularly useful since timing tool doesn’t really need to optimize these timing paths where synchronization topologies are used.

- Two Flop Synchronizer :

When a signal crosses from one clock domain to another it needs to be synchronized first before it is used in destination domain. The destination register can experience setup/hold violation since there is no relationship between the launching and capturing clock domain edges. Metastability problem in first flop of synchronizer is blocked using second flop with an assumption that the first flop will settle down to a value before the next clock edge arrives on second flop. The probability of metastability occurrence on both the flops directly depends on the frequency of the clock source. The settling time in case of a violation depends on the flop characteristics of a particular technology library and has to be lesser than the fastest clock supported in the design.

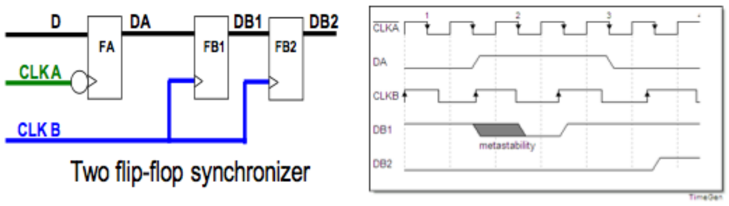

Figure 4: Two Flop Synchronizer

Please note that it is not necessary that DB1 signal will settle down to a high value after the metastability period. The design has to ensure the stability of DA signal for more than one cycle of the destination clock. If the design can't ensure the stability then the same can cause the change of the signal unregistered in destination domain.

Two flops synchronizer is generally used to synchronize the control signal across the domain.

- Mux Synchronizer

  Mux synchronizer based design topology is generally used when designer has to transmit a multi bit data bus across any clock domain. In this topology, the clock domain crossing of the data bus is controlled by the mux select signal which gets enabled when the input data becomes valid at mux input. This ensures that the destination flop will never be meta-stable due to change in data input.

  Designer has to ensure here that the input data should remain stable when the mux enables its input path and it should only change when the mux selects the feedback path.

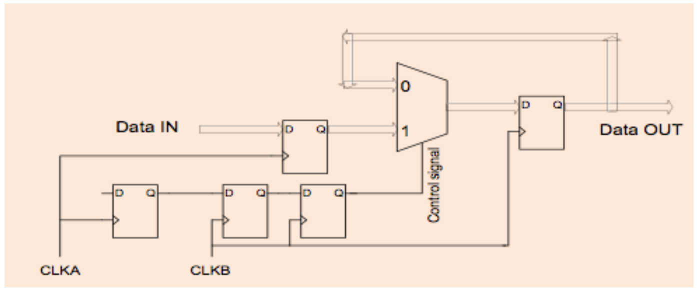

Figure 6: Mux Synchronizer

- FIFO

FIFOs are often used to safely pass data from one clock domain to another asynchronous clock domain. Using a FIFO to pass data from one clock domain to another clock domain requires multi-asynchronous clock design techniques.

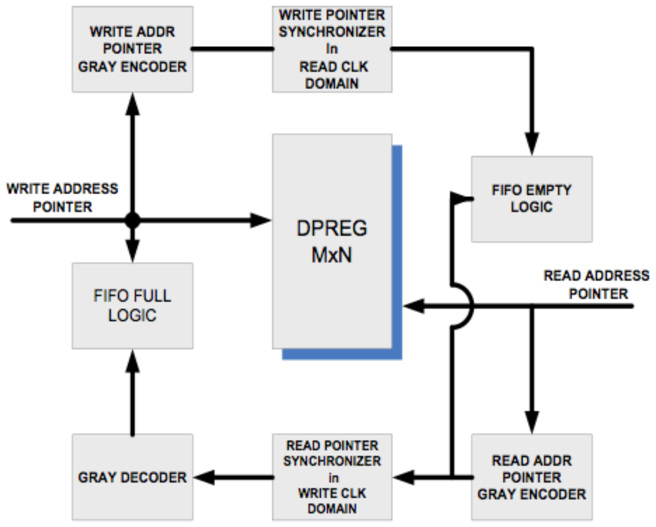
WRITE ADDR POINTER GRAY ENCODER
WRITE POINTER SYNCHRONIZER In READ CLK DOMAIN
WRITE ADDRESS POINTER
DPREG MxN
FIFO EMPTY LOGIC
READ ADDRESS POINTER
FIFO FULL LOGIC
GRAY DECODER
READ POINTER SYNCHRONIZER in WRITE CLK DOMAIN
READ ADDR POINTER GRAY ENCODER

Figure 7: Asynchronous Fifo

The write Pointer points the next memory location for Data Write. The read pointer points the current FIFO location to be read. Write pointer value changes with write clock and read pointer value changes with read clock, however both the pointers cross clock domain to determine fifo full/empty logic. Hence it is important to synchronize them in destination clock domain before using in FIFO full/empty logic. Gray Encoding is preferred before crossing clock domain for multi bit signals.

- Handshaking
  One of the popular methods to send data from one clock domain to another is using hand shaking protocol, where sender sends the data with request and the receiver acknowledge the data by sending acknowledgment. Both the request and acknowledgement signal gets synchronized in destination clock domain before being used inside destination domain state machine.

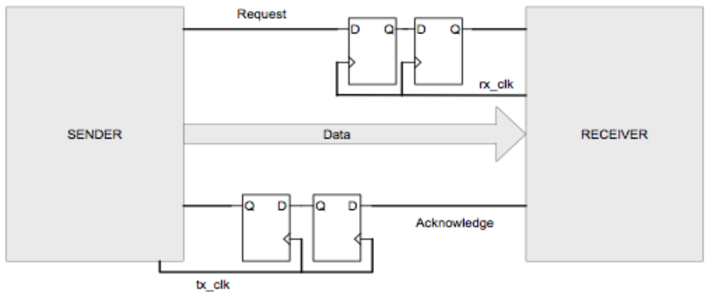

Figure 8: Handshaking scheme

1.1.4 Pseudo Asynchronous False Path

Pseudo asynchronous false paths are those where source and destination register both are clocked by same clock or a clock derived out of the same source but still they are designed such a way that meeting the timing specification is impossible and can be ignored. The difference in clock and data skews between source and destination registers makes it virtually impossible to meet timing. In such cases, designers need to ensure that there should not be any stringent timing requirement for these paths.

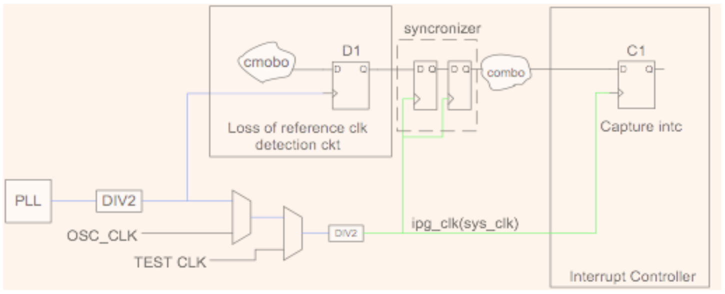

Figure 9: Pseudo Async False Path

In the above design PLL clock is used to generate system_clk along with other on chip/off chip clock sources. Although system_clk is derived from PLL, there is a huge uncommon path between PLL output and system clock due to functional and test clock mux, postscaler and clock gating cell present down the line in clock path.

The circuit working on PLL output can communicate with logic present in system clock.Since system clock is derived from PLL , hence both the clock are treated synchronous. Tool will try to meet timing for the same. However the huge uncommon clock path sometime becomes nightmare for timing tool to meet. This kind of pseudo asynchronous path can be treated as false path. However treating false path can't prevent the "capture_intc_reg" to become meta-stable as no timing is ensured which can lead to any functional failure.

In the above design a 2 flop synchronizer has been placed which blocks the metastability to propagate to the whole system. Now, simply "D1_reg->Sync1_reg" path can be treated as false path.

Note: In the above circuit, the designer should ensure the delay because of two-flop synchronizer won't cause any protocol or functional issue in design.

Next: Multi Cycle Path

< Previous **Page 1 of 2** **Next >**

Decode a quadrature encoder in software

Choosing a mobile-storage interface: eMMC or UFS

Gate level simulations: verification flow and challenges

Floorplanning: concept, challenges, and closure

MBIST verification: Best practices & challenges

Power integrity domains

Moving averager rejects noisy outlier values

Modern ADCs improve CMOS image sensors

Design planning for large SoC implementation at 40nm - Part 2

FEATURED RESOURCES

Subscribe to RSS: or

DESIGN CENTERS

5G
Analog
Automotive
Components & Packaging
Consumer
DIY
IC Design
LEDs
Medical
PCB
Power Management
Sensors
Systems Design
Test & Measurement

MORE EDN

Blogs
Design Ideas
Tech Papers
Courses
Webinars
EDN TV
Events
About Us

ASPENCORE NETWORK

EBN EDN EE Times EEWeb Electronic Products Electronics-Tutorials Embedded Planet Analog ElectroSchematics
Power Electronics News TechOnline Datasheets.com Embedded Control Europe Embedded Know How Embedded News
IOT Design Zone Motor Control Design Electronics Know How

GLOBAL NETWORK

EE Times Asia EE Times China EE Times India EE Times Japan EE Times Taiwan EDN Asia EDN China EDN Taiwan
EDN Japan ESM China

Working With Us: About | Contact Us | Media Kits