Presented by:

Mohamed Adel Melad Abd_Elhameed Youssef

Subject: Synopsys Spyglass Tool

Under supervising of:

Eng: Mohamed Zaiton

Q1: Mention the major violations linting tool can detect.

ANS:-

- Combinational Loops
- Unused and Undriven Signals
- Inferred Latch

Q2: Mention the error appeared after running linting goals for these codes and mention how will you fix them:

ANS:

a- Blocking assignment 'a = (a & b);' used inside a 'FlipFlop' inferred sequential block solution:

```
always@(posedge clk)
a <= a & b;
```

b- No Errors

c- Signal c is being assigned multiple times (previous assignment at line 11) in same always block solution:

```
always@(posedge clk)
begin
if(a>b) c <= a ;
else c <= c ; end
```

d- No Error

e- Input 'B' declared but not read solution:

assign C= A;

endmodule

f- RHS of the assignment contains 'X'(Reason : Assigned with 'x' in assignment) solution: give A value 0 or 1

g- Procedural assignment statement cannot drive a net : x Latch inferred for signal 'x' in module 'q2_partg' Detected undriven input terminal q2_partg.v[1:0]

solution:

always@(*)

case(v)

2'b00: x=1;

2'b10: x=0;

default : x=0;

endcase

h- Combinational loop exists at solution: reg temp1, temp2,temp3; assign out = en? in : temp1; assign a = en? temp2 : c; assign z = en? temp3 : y;

```
always@(posedge clk)
begin
 temp1 <= out;
temp2 <= z;
temp3 <= a;
end
```

i- Asynchronous reset signal used as non-reset/synchronous-reset at instance Reserved name 'in' used Flip-flop 'out2' has neither asynchronous set nor asynchronous reset.

solution:

```
always @(posedge clk or posedge rst)
begin
if (rst) out1 <= 0;
else out1 <= in_1; end
always @(posedge clk or posedge rst)
begin
if (rst) out2 <= 0;
else out2 <= in_1; end
```

Q3: Run your previous RISCV design code on spyglass and mention the errors got appeared and fix them.

Server closed while solving Errors.

1- Mention at least 3 CDC problems and their solutions.

Data Metastability:

- **Problem:** When a signal crosses from one clock domain to another, it can encounter a metastable state if the receiving flip-flop samples the signal at an unstable time, leading to unpredictable behavior.
- **Solution:** Use synchronizers to mitigate metastability. A two-stage synchronizer, also known as a dual flip-flop synchronizer, is commonly employed. This involves two flip-flops in series, each clocked by the destination clock domain. It helps reduce the risk of metastability by providing a double-sampling mechanism.

Asynchronous Assertion:

- **Problem:** An asynchronous assertion occurs when an input signal from one clock domain is used directly as an asynchronous control signal in another clock domain without proper synchronization.
- **Solution:** Synchronize the asynchronous control signal using a synchronizer before it is used in the destination clock domain. This ensures that the signal is stable during the entire clock period and avoids potential setup and hold time violations.

Clock Domain Crossing Setup and Hold Violations:

- **Problem:** Violations of setup and hold times can occur when signals crossing clock domains are not properly synchronized, leading to incorrect data capture or setup/hold time violations in the receiving flip-flop.
- **Solution:** Analyze and adjust the synchronization paths to ensure that the setup and hold time requirements of the receiving flip-flop are met. This may involve adjusting the delay or using more sophisticated synchronization techniques. Simulation and formal verification tools can help identify and address these violations.

2- Are synchronous clock domain crossings safe?

Synchronous clock domain crossings (CDCs) are generally considered safer than asynchronous crossings, but they still require careful consideration and proper design practices to ensure reliable operation. In a synchronous CDC, the clock domains involved have a known and fixed relationship, typically through a common clock or a fixed phase relationship between clocks.

Here are some points to consider when dealing with synchronous clock domain crossings:

- **Common Clock Source:** same clock
- **Fixed Phase Relationship:** same clock phase
- **Synchronization Elements:** flip/flop pairs
- **Setup and Hold Times:** must be met
- **Use of Gray Coding:** for asynchronous reset signal to reduce glitches

3- Do metastability problems depend on the frequency of the destination clock?

» Metastability problems are not directly dependent on the frequency of the destination clock. Metastability is a phenomenon that occurs when a signal transitions near the sampling edge of a flip-flop in the destination clock domain. It is primarily influenced by the setup and hold times of the flip-flop, the uncertainty in the arrival time of the signal, and the characteristics of the signal itself.

4- Can static timing tool like Synopsys Prime Time detect asynchronous clock domain crossings?

» Static timing analysis (STA) tools like Synopsys PrimeTime are primarily designed to analyze and verify the timing characteristics of a digital design under the assumption of synchronous operation. Asynchronous clock domain crossings (CDCs), where signals transition between different clock domains with independent clock sources, present challenges for traditional STA tools. STA tools are not inherently designed to detect or analyze asynchronous CDCs.

» To address asynchronous CDC issues, designers typically use specialized CDC analysis tools or techniques. Some common approaches include:

- **Synchronization Elements:** Inserting synchronization elements (e.g., double flip-flops or gray-code-based synchronizers) at the boundary of asynchronous domains to reduce the risk of metastability.
- **Cross-Clock Analysis Tools:** Specialized tools designed for CDC analysis can identify potential issues related to asynchronous domain crossings. Examples include SpyGlass CDC and VCS CDC.
- **Simulation and Formal Verification:** Performing detailed simulations and formal verification to identify and address potential issues arising from asynchronous CDCs.

5- Name the two key problems caused by asynchronous clock domain crossings

Asynchronous Clock Domain Crossings (CDCs) can introduce several challenges in digital designs, but two key problems associated with them are:

1. **Metastability:**
    - **Description:** Metastability is a phenomenon that occurs when a signal transitions near the sampling edge of a flip-flop or latch in the destination clock domain. It leads to unpredictable behavior and can result in the output being temporarily caught in an indeterminate state.
    - **Cause:** Asynchronous signals transitioning between clock domains may arrive at the destination flip-flop at times that violate the setup and hold time requirements, leading to the potential for metastability.
    - **Consequences:** Metastability can cause incorrect system behavior, data corruption, and potentially propagate errors throughout the design. Resolving metastability typically involves adding synchronization elements (e.g., double flip-flops or synchronizers) to reduce the risk.

2. **Data Transfer Synchronization:**
    - **Description:** Asynchronous CDCs can result in data transfer between different clock domains, and ensuring the synchronized and correct transfer of data is challenging.
    - **Cause:** The lack of a common clock reference between the clock domains makes it difficult to guarantee synchronized data transfer without proper synchronization mechanisms.
    - **Consequences:** Without adequate synchronization, data may be sampled incorrectly, leading to data corruption and potential functional errors. Synchronization techniques, such as dual-clock FIFOs or handshake protocols, are often employed to address this issue.

6- Do metastability problems depend on the frequency of the destination clock? mention why if(yes or no).

While the frequency of the destination clock doesn't directly cause metastability, the following factors can influence metastability risk:

1. **Clock-to-Q Delay (tCOQ):** The clock-to-Q delay is the time it takes for a flip-flop to produce a stable output after the clock edge. This delay affects the window of vulnerability to metastability. Higher frequencies may result in shorter clock-to-Q delays, reducing the metastability window.
2. **Signal Transition Timing:** The timing of signal transitions relative to the clock edge is critical. Faster transitions or signals changing close to the clock edge increase the risk of metastability.
3. **Clock Domain Asynchronicity:** The level of asynchronicity between clock domains affects the likelihood of metastability. Synchronizing signals between domains reduces the risk.
4. **Number of Synchronization Stages:** Adding more synchronization stages (flip-flops) in a synchronizer chain increases the time available for metastability resolution but introduces additional clock-to-Q delays.

9- Write a rtl code for this architecture and it's SGDC file and run it to spyglass with fixing errors will appear:

RTL and CDC_Constraint.sgdc concatenated with my mail.

Running errors history:

Design Read (4):

STX_VE_361 (4) : A procedural assignment statement cannot drive a net other than reg type.(wire and reg missing) => I fixed it.

Cdc/cdc_verify_struct has an error:

Ac_conv04 (2): Checks all the control-bus clock domain crossings which do not follow gray encoding(Control destination bus 'CDC_Check.qualifier1[0:7]' has a clock domain crossing. Gray encoding check:DISABLED) => Ifixed it.

Other error was data loss when we lunch data by clk1_fast and capture that data by clk2_slow => I fix it by fifo or serial in serial out technique.