CDC

Abanoub Emad Hanna

Metastability

Causes

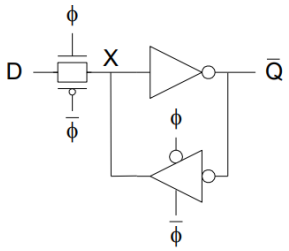
Metastability occurs when input changes during aperture time. This might happen due to:

- **Asynchronous inputs**
- **Clock Domain Crossing (CDC):** Two clocks can be considered different domains or 'Asynchronous' if they are **not from the same source** even if they have same frequency. If the 2 clocks have tha same frequency, despite the fact that the phase difference will be constant during the operation of the system, after each startup the phase difference is different each time.

Explanation

Transitioning during the aperture window causes the master's latch to close while having $\Delta V_{in}(V_x - V_y)$ around V_m.
FF's output will not settle before t_{pcq}, since latch's delay is variable given by:

$$t_d = \tau \ln \frac{V_{DD}}{\Delta V_{in}}, \text{ where } \tau = \frac{C_L}{g_m}$$



Effects

1. **High CLK-to-Q Delay:** This might lead to setup time violation for the receiver FF.
2. **Short Circuit Current:** Since ΔV_{in} is around V_m, the short circuit current is high.
3. **Different Values Propagation:** If the metastable FF drives 2 paths, the 2 paths might have different values.

Probability

- The probability of metastability per input change is given by ($T_a = t_{setup} + t_{hold}$):

$$P(MS) = \frac{T_a}{T_c} = T_a f_c$$

- Fialure Rate = BER = Probability of failure per second is given by:

$$FR = f_{in} \frac{T_a}{T_c} = T_a f_c f_{in}$$

- Mean Time Between Failure (MTTF) is given by:

$$MTBF = \frac{1}{FR} = \frac{1}{T_a f_c f_{in}}$$

Note: Metastability can never be eliminated, but it can be minimized.

Synchronization

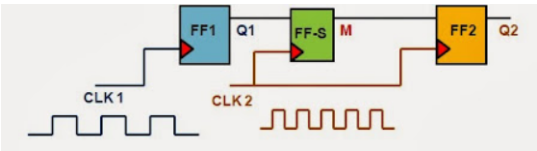
Simple Bit Synchronization

Behaviour

- If ΔV_{in} is exactly V_m, the output should theoritically be stuck at V_m (review inverter curve). However, presence of noise as well as mismatch between inverters' switching thresholds would make it $V_m + \Delta$, so the output will be driven to V_{DD} or GND in the same clock cycle due to the high gain of the inverters in the FF.
- However, this value might be wrong, but the FF will eventually settle to the correct value after an unkonwn number of clock cycles (2 or 3 for double flop synchronizer) depending on the Δ around V_m.
- Therefore, the input should be stable for at least $1.5 * T_{clk}$ of the receiver clock to ensure correct data sampling.

Notes:

- If the difference in clock speeds is huge, a third FF is added.
- Some librarirs provide synchronizer cells called meta-hardened flops.
- $t_{resolve} = T_{clk} - t_{setup}$



Probability

- Probability of synchronizer failure is given by:

$$P(syncfailure) = \frac{\Delta V_{in}}{V_{DD}} = \exp\left(-\frac{t_{resolve}}{\tau}\right)$$

- Probability of system failure is given by:

$$P(systemfailure) = P(MS)P(syncfailure) = T_a f_c \exp\left(-\frac{t_{resolve}}{\tau}\right)$$

- Mean Time Between Failure (MTBF) is given by:

$$MTBF = \frac{1}{T_a f_c f_{in} \exp\left(-\frac{t_{resolve}}{\tau}\right)}$$

Pitfalls

- Do NOT synchronize data lines.
- Do NOT drive two synchronizers with the same signal; The two synchronizers can resolve to different values.
- Do NOT put the sync FFs far apart or put logic between them; The wire or logic delay will eat from your resolution time.
- Synchronizers must have good feedback loops; You cannot use dynamic FFs

Data Bus Synchronization

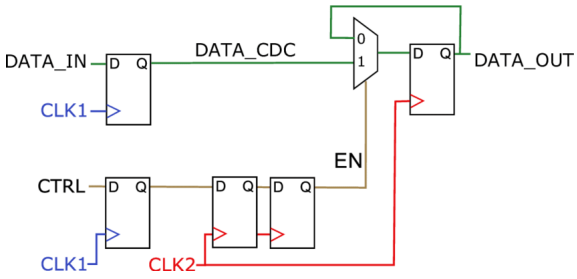
Simple bit synchronizer can not be used for bus synchronization, due to **Different Routing Delays**. Different bits might arrive at different times during the aperture time causing each FF to have different ΔV_{in}, so each FF might rseolve metastability at different clock cycles, so some bits would have an old value, while others have a new one.

Gray Coding Technique

- Gray coding ensures that only one bit changes at a time, so the non-changing bits don't care if value is old or new since they are the same.
- It is only used for data that follows a certain pattern (e.g. counter).

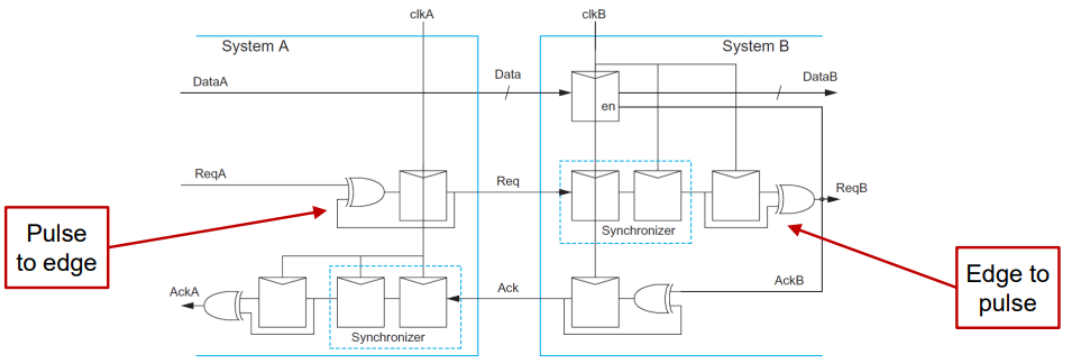
Enable-Based Technique

- Source domain sends an enable signal, which is synchronized by two FFs.
- Data should be constant for 2 cycles, so it is sampled correctly by the receiver.
- The data receiving FFs don't suffer from metastability, since they are only enabled when the data is stable.
- Instead of using FFs that have EN, a MUX can be used to select between the old and new values.



Handshake Technique

- Source domain sends a request signal, which is synchronized by two FFs.
- The receiver domain sends an acknowledge signal, which is also synchronized by two FFs.
- This technique is preferred when data is sent occasionally between subsystems, due to its very high latency.
- Handshake protocol can be:
  - **4-Phase:** Request and Acknowledge are level sensitive.
  - **2-Phase:** Request and Acknowledge are edge sensitive.
- Pulse to edge and edge to pulse converters are used to convert between the two protocols.



Note: Only fast to slow domain crossing causes data loss.

Asynchronous FIFO

FIFO is best suited when transmitter is faster than receiver, since it can store data until the receiver is ready to read it.

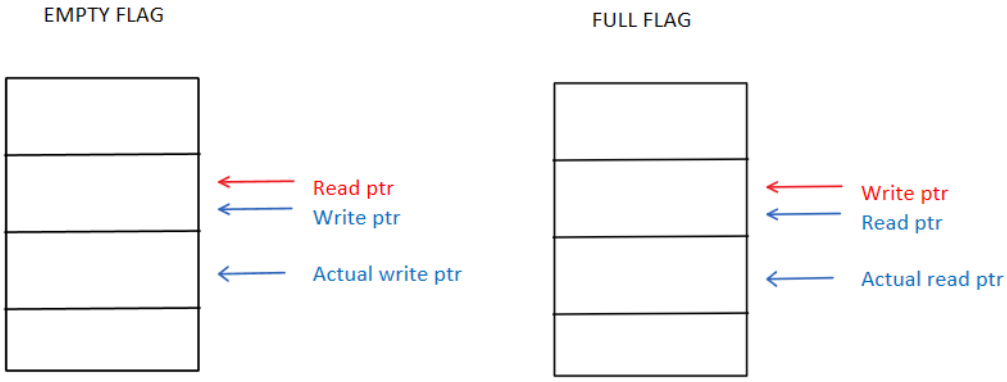
Operation

- **Empty Flag Condition:** Empty flag is set when read pointer is incremented and the new value is equal to the write pointer. Otherwise, we would have read same value twice.
- **Full Flag Condition:** Full flag is set when write pointer is incremented and the new value is equal to the read pointer. Otherwise, we would have overwritten an unread data.
- To differentiate between full and empty flags, an additional bit is added to the pointers.
- Comparing the pointers from different domains would lead to metastability, so double flop synchronizers are used. Since pointers are multi-bit, gray coding must be used to avoid data incoherence.

Conservative Scenarios

These scenarios happen due to the added latency of the pointers' synchronizers. They are not catastrophic as data is not lost, but operation falsely stops for 1 clock cycle. For example:

- When comparing pointers for empty flag, the write pointer could be outdated and equal to read pointer, despite the fact that it has been incremented, so the empty flag is raised until correct value is written.
- When comparing pointers for full flag, the read pointer could be outdated and equal to write pointer, despite the fact that it has been incremented, so the full flag is raised until correct value is read.



Depth Calculation

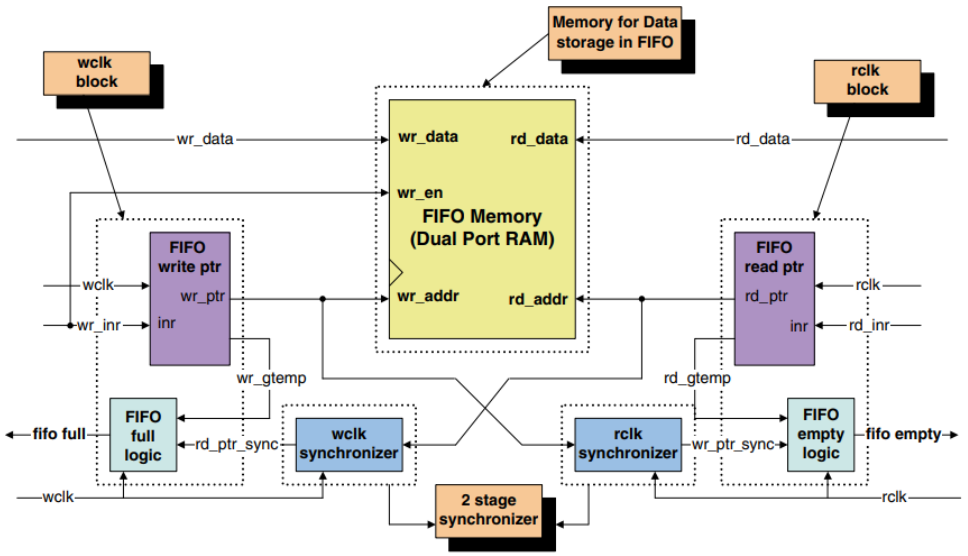
Definitions:

- **Effective frequency:** Since there might be constatnt idle cycles during writing and reading, the write/read frequency is reduced. For example, if there is 3 idle clock cycles between successive writes/reads, the effective frequency is $\frac{1}{4}f_{write}$.
- **Burst Length:** Number of data items to be transferred.

Example:

- **Givens:** $f_{write} = 80MHz$, $f_{read} = 50MHz$, Burst Length = 120, Write idle cycles = 1, Read idle cycles = 3.
- **Solution:**
  - **Effective frequencies:** $f_{write} = 80/2 = 40$MHz, $f_{read} = 50/4 = 12.5$MHz.

- **Write Time:** $120 * (1/40) = 3\mu s$
- **Number of items that can be read:** $3\mu * 12.5M = 37.5 \simeq 37$
- **Remaining data to be stored:** $120 - 37 = 83$
- Therefore, the FIFO depth should be at least 83.



Reset Synchronization

- A double flop synchronizer is used to synchronize the reset signal, because reset signal could be de-asserted during the aperture time causing metastability (violating recovery time).
- Therefore, designs should have asynchronous reset assertion and synchronous reset de-assertion.
- The number of reset synchronizers is equal to number of clock domains.
- If Reset signal is not input to second FF, the reset assertion will be synchronous. Also, if there is clock gating, the system will not reset.