

Synchronizers :

→ 2-FF Synchronizer :

→ Toggle Synchronizer :

FIFO [CDC]

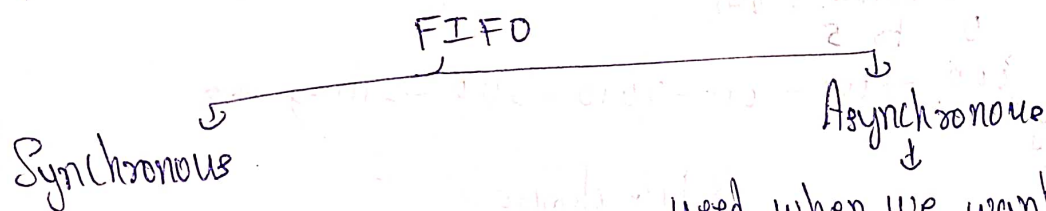
→ First in First Out

→ For single bit Synchronization, we have 2-Flip Synchronizers.

→ For pulse Synchronization with $f_{clk1} > f_{clk2}$ we have Toggle Synchronizer.

→ For multi-bit Synchronization we have, MUX Synchronizer, handshake synchronizers, and FIFO, etc.

→ FIFO is not limited to CDC only. Even in a single clock domain, there can be cases when the read frequency is less than write frequency, even in those cases, we need FIFO to store the data.



Why Gray code?

→ Because there is only 1 bit difference b/w any number and its incremental next number, so we can avoid false full and false empty condition.

FIFO depth:- The no. of slots or memory locations in a FIFO ^{is} ~~are~~ called

FIFO depth.

* If $FIFO\ depth = 8 \Rightarrow 3\ bits\ sufficient\ for\ the\ address\ but\ to\ determine\ full\ and\ empty\ conditions,$
we need ^{MSB} 1 bit extra.

$\rightarrow 2^3$

Binary

$\Rightarrow 000 \quad 001 \quad 010 \quad 011 \quad 100 \quad 101 \quad 110 \quad 111$

Gray

$\Rightarrow 000 \quad 001 \quad 011 \quad 010 \quad 110 \quad 111 \quad 101 \quad 100$

1 bit change.

\rightarrow In Sync FIFO:

9, 10, ..., Anything can be depth.

In Async FIFO:

Only power of 2? [Not mandatory but need to follow some rule].

Ex: For Async FIFO with 6 deep.

\Rightarrow 0 to 5
 \Rightarrow 000 \rightarrow 001 \rightarrow 011 \rightarrow 010 \rightarrow 110 \rightarrow 111 \rightarrow 100

3 bits change
 (not acceptable)

$$\Rightarrow \left(\frac{2^n}{2} - \frac{\text{FIFO Depth}}{2} \right) \text{ to } \left(\frac{2^n}{2} + \frac{\text{FIFO Depth}}{2} - 1 \right)$$

Ex: 520 deep

$$\Rightarrow \left(\frac{2^{10}}{2} - \frac{520}{2} \right) \text{ to } \left(\frac{2^{10}}{2} + \frac{520}{2} - 1 \right)$$

252 to 771.

Gray \Rightarrow 010000010 to 1010000010

1 bit change.

* Binary to hexay code conversion:-

Binary $\rightarrow A$

$$\text{hexay} \rightarrow A_g \Rightarrow \boxed{A_g = A^{\wedge} (CA \gg \gg 1)}$$

overflow in Asynchronous FIFO: Overflow occurs when an attempt is made to write new data in a full FIFO.

underflow: underflow occurs when ~~an~~ there is no data to provide on a read request, because the FIFO is empty.