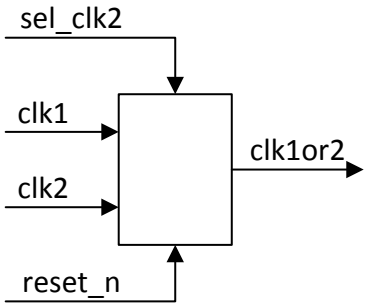


Clock mux

Often a piece of logic has to run of two different clocks. An external signal selects between the two clocks. In doing this the output clock should be 'clean', that is: the low or high period at the output should never be shorter than the high or low period of either of the two input clocks. The module **`clock_mux.v`** takes care of all the difficulties of that process.

Diagram



Description

The module has the following ports:

```
input    clk1,      // Clock 1
input    clk2,      // Clock 2
input    reset_n,   // System reset
input    sel_clk2,  // Select clock2 when high
output   clk1or2    // Selected clock
```

`clk1,clk2`
The two input clocks. No ratio or phase relation between the two clocks is required.
(Although in the test bench the clk2 period must be longer than the clk1 period, this is for testing purposes only. In reality this condition does not apply).

`reset_n`
The system reset, is active low. The reset can be asserted asynchronously. In this circuit the reset may also be released asynchronously. The synchronisation registers will make that no clock glitches will appear.

`sel_clk2`
This signal selects which of the two clocks is to be selected. If it is low **`clk1`** will be selected; if it is high **`clk2`** will be selected.
This signal can be asynchronous to both input clocks. There is also no timing constraints of **`sel_clk2`** against the reset.
This signal is assumed to have a period much, much slower than the clock inputs. As a lower limit, the **`sel_clk2`** must have a high and low duration of at least 2 clock periods of the slowest of clk1 or clk2. If the above rule is not followed, certain pathological patterns of **`sel_clk2`** against the input clocks will produce low period violations.

`cklk1or2`
This is the selected clock. ***During*** a reset the **`cklk1or2`** signal will be equal to **`clk1`**. After a reset the **`sel_clk2`** signal determines if **`clk1`** or **`clk2`** appears at this output. This signal will be low for a short period of time when switching from one clock to the other.

Corner cases

The general behaviour of the module is described above. However there are some corner cases to consider.

If a clock is selected but that clock is not running the **`cklk1or2`** output will remain low. It is possible to switch back to the other clock input.
If the selected clock is not running but then starts running, the **`cklk1or2`** output will track that input clock after one or two clock cycles.

Beware

The module makes extensive use of synchronisation registers. It will only work correctly if those are implemented following the rules for such circuits.

Copyright