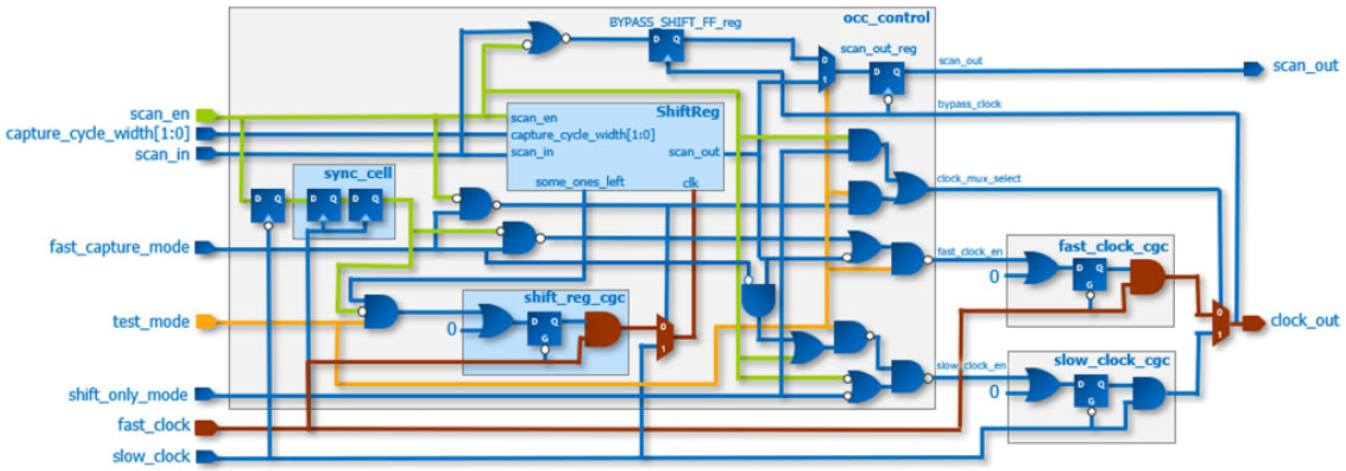
tessent occ



1. operation mode

1.1 function mode

functional mode (test_mode = 0) that enables the fast clock gater to supply a fast clock to the design.

1.2 shift mode

shift mode (scan_en = 1) that uses the slow_clock to load and unload the scan chains, including the condition bits in ShiftReg.

1.3 shift only mode

Shift-only mode disables the OCC (test_mode = 0) and activates shift (scan_en = 1). It enables the slow clock clock gater to use the slow clock path for shift, and it enables bypass shift.

这个信号默认需要连接到 DFT信号ext_ltest_en

1.4 slow capture mode

slow capture mode (fast_capture_mode = 0) that uses the slow_clock to capture data into the scan cells and to shift the condition bits in ShiftReg.

1.5 fast capture mode

fast capture mode (fast_capture_mode = 1) that uses the fast_clock to capture data into the scan

cells and to shift the condition bits in ShiftReg.

1.6 kill clock mode

kill clock mode (kill_clock_en = 1) that enables you to block propagation of fast_clock input to clock_out output. Asserting the kill_clock_en signal is effective in the two operating modes when the OCC is not active:

2. 结构伪代码

```
   clock_mux21 tessent_persistent_cell_inject_tck_mux (
     .A0                                        ( slow_clock_buf_out
),
     .A1                                        ( ijtag_tck
),
     .S0                                        ( inject_tck
),
     .Y                                         ( slow_clock_tck_injected
)
   );


   cgand tessent_persistent_cell_cgc_fast_clock (
     .CK                                        ( fast_clock_buf_out
),
     .FE                                        ( fast_clock_en
),
     .TE                                        ( fast_clock_en
),
     .GCK                                       ( fast_clock_gated
)
   );

   cgand tessent_persistent_cell_cgc_slow_clock (
     .CK                                        ( slow_clock_tck_injected
),
     .FE                                        ( slow_clock_en
),
     .TE                                        ( slow_clock_en
),
     .GCK                                       ( slow_clock_gated
)
   );

   clock_mux21 tessent_persistent_cell_clock_out_mux  (
     .A0                                        ( fast_clock_gated
),
```

```
      .A1                                       ( slow_clock_gated
),
      .S0                                       ( clock_mux_select
),
      .Y                                        ( clock_out
)
    );


    assign kill_clock_en_gated = kill_clock_en & ~test_mode;
    assign fast_clock_en    = ((ShiftReg_SCAN_OUT & fast_capture_mode &
(SCAN_EN_inv_sync || active_upstream_parent_occ)) | (~test_mode)) &
(~kill_clock_en_gated);
    assign slow_clock_en    = (((ShiftReg_SCAN_OUT & (~fast_capture_mode)) |
scan_en) & test_mode) | inject_tck | (shift_only_mode & scan_en);
    assign clock_mux_select = ((scan_en | (~fast_capture_mode)) & test_mode)
| inject_tck | (shift_only_mode & scan_en);
```

3. Parent mode

Parent-mode operation is one of the operating modes of the standard OCC.

This operating mode bypasses the OCC clock gating logic and performs clock selection only.
It requires a downstream child-mode OCC to perform the clock gating and chopping.

Parent-mode, 父OCC只做时钟function clk 与shift clk之间进行切换，不做OCC的pluse gateing(这部分由child-OCC来实现)

tile parent mode OCC + child OCC, 类似于OCC复制

非SSN场景
top parent mode OCC + n个tile内部child OCC,无需单独shift clock连线到tile,tile内部也不用单独check shift timing

如果多个top parent mode OCC(频率不同) + tile child OCC, function模式下不同频率时钟为异步关系，切换到shift模式下，全为shift时钟,scan register单chain存在skew,需要插入latch

SSN场景
shfit clock都是走tile内部(由SSH产生),intest时不考虑tile与tile之间的shift clock skew差异，这时用标准OCC是最好的。

4. Child OCC

SSN好像是不支持child OCC

Child OCC有两种实现：

- 普通OCC的child mode

- child-only的OCC

```
#级联OCC会报R27问题，配置成Parent OCC mode + child OCC mode就不会报这个问题。
add_core_instances -inst [get_inst top_rtl2_tessent_occ_parent_*] \
                          -parameter {parent_mode on fast_capture_mode on}
set_static_dft_signal_values retargeting3_mode 1
//set_drc_handling R27 warning
set_core_instance_parameters -instances \
            coreb_i1/coreb_rtl2_tessent_occ_clka_inst -parameter_values \
            {active_upstream_parent_occ on}
set_core_instance_parameters -instances \
            coreb_i1/coreb_rtl2_tessent_occ_clkb_inst -parameter_values \
            {active_upstream_parent_occ on}
set_core_instance_parameters -instances \
            coreb_i1/coreb_rtl2_tessent_occ_clkc_inst -parameter_values \
            {active_upstream_parent_occ on}
```

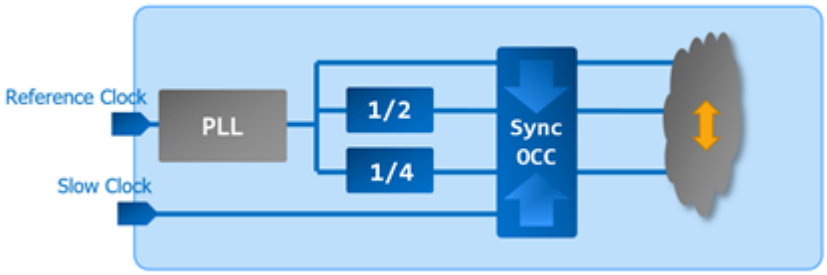
5. sync OCC

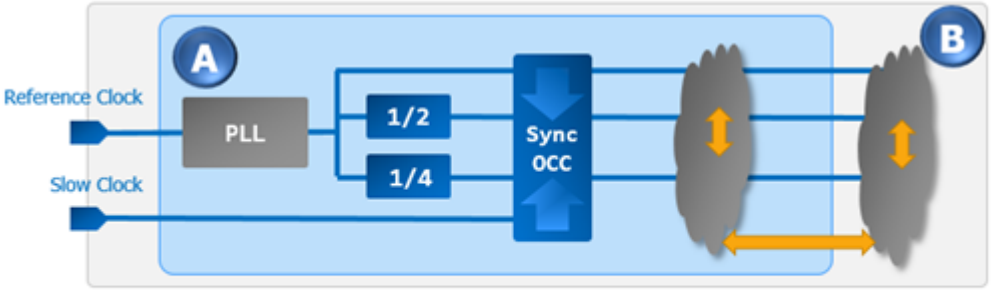
5.1 使用场景

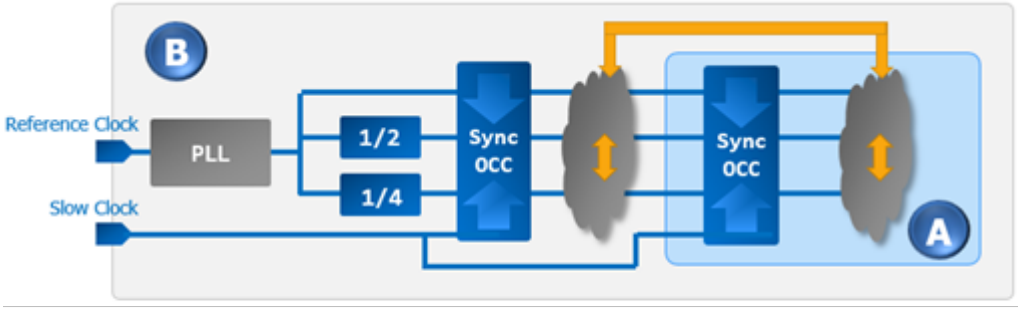
对于分频时钟和不分频时钟之间path当做同步处理的情况，需要使用同步OCC。

在任何场景下。intest/extest, 分频只能发生在第一级sync-occ。后级的sync occ只能是使用前级sync-occ的时钟输出。

1. intest时，模块sync-occ工作，顶层sync-occ穿通，测试模块内部的path
2. extest时，模块sync-occ穿通，顶层sync-occ工作，测试模块之间的path

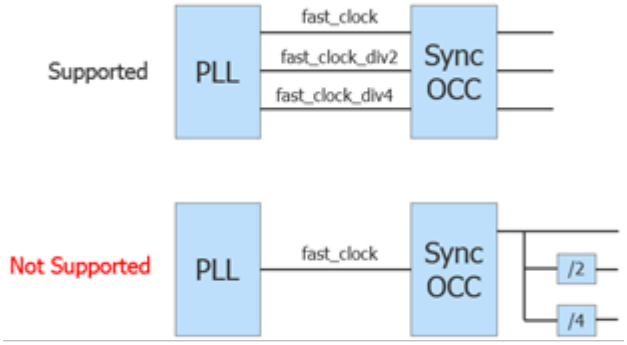






5.2 限制

sync occ后面不能接分频时钟。



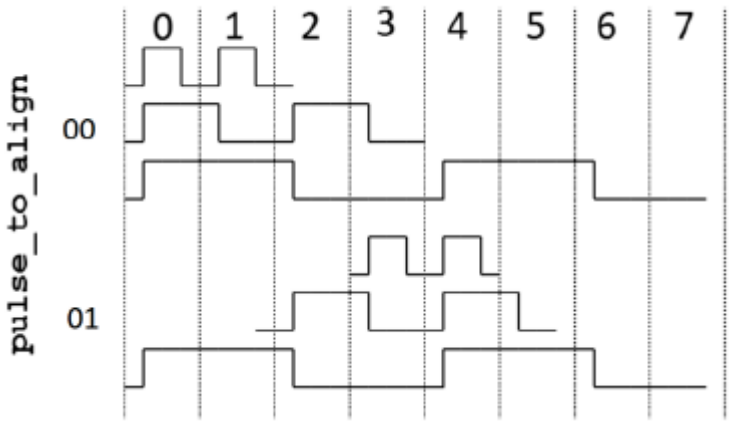
原则是任何测试模式的情况下都只能有一级OCC在工作，如果非得用本地的分频，基本上只有在intest情况下才有机会。

extest情况下，只能使用root处的分频，因为同步OCC的限制是pulse是用低频来shift的，必须保证低频shift一致。

5.3 波形

有两种配置， 我理解分别cover 低频到高频，高频到低频。

出capture pulse, 是使用低频时钟来出pulse的，高频在对应的位置点采用gating的方式，保证也只出2个pulse


0
1
2
3
4
5
6
7
pulse_to_align
00
01