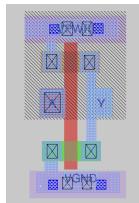


To open file:

- cd vsdstdcelldesign
- magic -T sky130A.tech sky130_inv.mag &



- Now in: (A = input, Y =output)(lef file only has input,output,power,ground - protection)
- Rules to follow in design:
 - Input + output must lie on intersection of horizontal + vertical design tracks
 - Width of standard cell should be in whole multiples of track pitch, and height augmented vertical track
- Go to
`/home/beaver/Desktop/work/tools/openlane_working_dir/pdkssky130A/libs.tech/openlane/sky130_fd_sc_h`

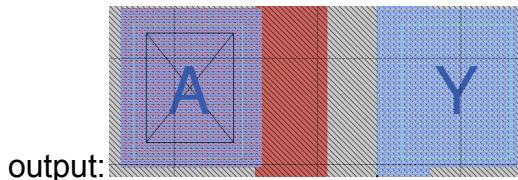
```
l1x X 0.23 0.46
l1y Y 0.37 0.34
net1 X 0.17 0.34
net1 Y 0.17 0.34
net2 X 0.23 0.46
net2 Y 0.23 0.46
net3 X 0.34 0.68
net3 Y 0.34 0.68
net4 X 0.46 0.92
net4 Y 0.46 0.92
net5 X 1.70 3.40
net5 Y 1.70 3.40
tracks.info (END)
```

- Less tracks.info : (inside) (for first input, x= horizontal, offset of 0.23, 0.46 in between each track in horizontal direction (pitch). Based on second line, vericle displacement=0.17, 0.34 in between each)
- Every metal layer x + y direction
- Tracks (metal places) used in routing - what to connect
- To check if follows track intersection rule:
 - In command line, grid on
 - Info on arguments needed in grid:

```
% help grid

Global Commands
-----
Layout Commands
-----
grid [xSpacing [ySpacing [xOrigin yOrigin]]]
      toggle grid on/off (and set parameters)
scalegrid a b    scale magic units vs. lambda by a / b
snap [internal|lambda|user]
      cause box to snap to the selected grid when moved
      by the cursor
```

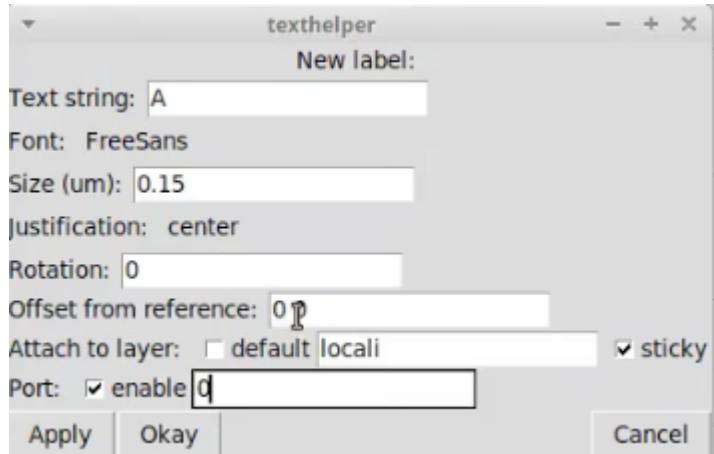
- 'Grid 0.46um 0.34um 0.23um 0.17um' — now new dimensions of tracks - adds tracks, we can see horizontal and vertical to intercept below input +



- To check second rule (width of cell should be in odd multiples of x pitch) – width = 3.5 boxes,



- Same rule for height
- To convert label to port:
 - Select label (where you want port)



- Edit -> text (lower level)
 - enable = sooner on lef file)(power (port use power) + gnd(port use ground) should be attached to metal 1, input+output locali)
 - If you want to see this info, select label and type what:

```
% what
Selected mask layers:
  locali  ( Topmost cell in the window )
Selected label(s):
  "Y" is attached to locali in cell def my_sky_inv
% port class output
% port use signal
```

To extract lef file:

- Once port labels set
- Name cell 'save sky130_vsdinv.mag'
- In vsdstdcelldesigns dir, magic -T sky130A.tech sky130_vsdinv.mag
- 'Lef write' in magic consul

- New item has been added:

```
-rw-r--r-- 1 root root 1321 Jul 23 11:07 sky130_vsdinv.lef
```



- Inside lef (changes + modifications I've made): (shows ports + info we declared)
- 'cp sky130_vsdinv.lef /home/beaver/Desktop/work/tools/openlane_working_dir/openlane/designs/ci/picorv32a/src' — makes sure synthesis maps to cells, has cell defs for synthesis, full characterization (cell rise, cell transition)(has both slow(max delay) and fast)
- Touch cong.tcl, copy in

```
# Design
setenv DESIGN_NAME "picorv32a"
setenv DESIGN_VOBLOC "file://"`designs/picorv32a/sky130_fd_sc_hd_typical.lib`
setenv DESIGN_VOBLOC "file://"`designs/picorv32a/sky130_fd_sc_hd_max.lib`
setenv CLK_PERIOD "12.000"
setenv CLOCK_PORT "clk"

setenv CLOCK_NET $::env(CLICK_PORT)

setenv(VERILATOR_SYNTH) "$::env(OPTIONAL_ROOT)/designs/picorv32a/sky130_fd_sc_hd_typical.lib"
setenv(VERILATOR_MAX) "$::env(OPTIONAL_ROOT)/designs/picorv32a/sky130_fd_sc_hd_slew.lib"
setenv(VERILATOR_DC) "$::env(OPTIONAL_ROOT)/designs/picorv32a/sky130_fd_sc_hd_dc.lib"
setenv(VERILATOR_DC_DC) "$::env(OPTIONAL_ROOT)/designs/picorv32a/sky130_fd_sc_hd_dc_dc.lib"

setenv(EXTEND_LEFS) "glob $::env(OPTIONAL_ROOT)/designs/$::env(DESIGN_NAME)/ver.vlef"
setenv FILENAME_LIMES($::env(OPTIONAL_ROOT)/designs/$::env(DESIGN_NAME)/$::env(VERILATOR).lime,vconfig.tcl
if ($::env(FILENAME_LIMES)) {
    source $FILENAME_LIMES
}
source $FILENAME_LIME
```

- In launcher:

- ./flow.tcl -interactive
- Package require openlane
- prep -design designs/ci/picorv32a
- If want continue old run:

```
0.9
% prep -design picorv32a -tag 15-09_06-28 -overwrite
```

- set lefs [glob \$::env(DESIGN_DIR)/src/*.lef]
- add_lefs -src \$lefs

```
Endpoint: 33890 (rising edge-triggered flip-flop clocked by clk)
Path Group: clk
Path Type: C2C
-----
```

Endpoint	Cap	Slew	Delay	Time	Description
0.00	0.00				Clock net (rise edge)
0.00	0.00	0.00	0.00		Lock net (fall edge)
0.00	0.00	3.3027	/CLK (sky130_fd_sc_hd_dfxtpl4)		
4	0.02	0.14	0.72	3.3027	count_cycle[8] (net)
0.14	0.02	0.12	0.00	3.3027	count_cycle[8] (net)
3	0.01	0.09	0.05	3.3027	count_cycle[8] (net)
0.11	0.07	1.32	0.00	3.3027	count_cycle[8] (net)
3	0.01	0.11	0.00	3.3027	count_cycle[8] (net)
0.10	0.08	1.32	0.00	3.3027	count_cycle[8] (net)
3	0.01	0.10	0.00	3.3027	count_cycle[8] (net)
0.10	0.08	1.30	0.00	3.3027	count_cycle[8] (net)
3	0.01	0.10	0.00	2.28	24742 /X (sky130_fd_sc_hd_or2_4)
0.10	0.08	2.28	0.00	24742 /X (sky130_fd_sc_hd_or2_4)	
3	0.01	0.10	0.00	2.28	24742 /B (sky130_fd_sc_hd_or2_4)
0.10	0.08	2.76	0.00	24742 /B (sky130_fd_sc_hd_or2_4)	
3	0.01	0.00			24993 (net)

- output:
- Displays slew, delay
- Report from opensd. Highlighted (vnxs)is maximum slack

```
tns -1598.21  
wns -15.96  
[INFO]: Synthesis was successful
```

- tns= total negative slack
- To repair high slack:

- Balance delay+area:

```
sky130_fd_sc_hd_8  
sky130_vsdinv 2201
```

```
Chip area for module '\picorv32a': 228590.486400
```

- Synth strategy description:

```
'SYNTH STRATEGY' | The main optimization came (when) from trying to turn off cell to high fan-out inputs and try to use an synthesis strategy (Default: calculated at runtime)  
'SYNTH STRATEGY' | Strategies for abc logic synthesis and technology mapping <br> Possible values are 0, 1 (delay), 2, and 3 (area)<br> (Default: '2')  
'SYNTH BUFFERING' | Enables abc cell buffering (Default: '1')
```

- (now set to 2, 0=most attention on delay)set \$::env(SYNTH_STRATEGY) 1 (area will increase, but delay decrease)
 - Synth-buffering description: (good bc reduces wire delay by inserting buffers) (right now 1)

```
'SYNTH BUFFERING' | Enables abc cell buffering <br> Enabled = 1, Disabled = 0 <br> (Default: '1')  
'SYNTH BUFFERING' | Enables abc cell buffering (Default: '1')
```

- Synth-sizing description (right now 0) (size cell for delay minimization)

```
'SYNTH SIZING' | Enables abc cell sizing (instead of buffering) <br> Enabled = 1, Disabled = 0 <br> (Default: '0')  
'SYNTH READ BLACKBOX LTR' | A file that enable reading the full/untrimmed library file as a blackbox for synthesis
```

- set \$::env(SYNTH_SIZING) 1
 - Synth_driving_cell description (right now sky130_fd_sc_hs_inv_8) (more fan-outs = need for more drive-strength)

```
'SYNTH DRIVING CELL' | The cell to drive the input ports. <br>(Default: 'efs8hd_inv_8')
```

- Running synthesis again, new area:

```
Chip area for module '\picorv32a': 320762.636800
```

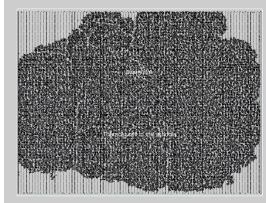
```
tns -36.81
```

```
wns -3.17
```

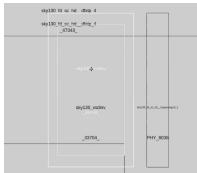
new slack: ~~tns -36.81~~ - much better

- In merged.lef in latestrun/tmp, vsd there, confidence that in placement cell will be present
 - Run_placement
 - Go to results/placement in run, start up magic:

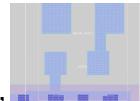
- Magic - T /...(etc)..../sky130A.tech lef read
/...(etc).../merged.lef def read picorv32a.def &



- To find our cell: cat picorv32a.placement.def | grep sky130_vsdinv
- Pick any instance, and in consul of magic
- select cell xyz
- Apartment - makes sure ground + power connected to cell



(our cell)



- In consul: 'expand' (see layers of our cell)
- In openlane folder, new file pre_sta.conf:

```
#include "lib/sky130_fd_sc_hd_inv_8.sdc"
#read_sdc("my_base.sdc")
#read_sdc("slow.lib")
#read_sdc("fast.lib")
#read_sdc("my_base.sdc")
```

(slow, fast, where read verilog)

- In certain_run/results/synthesis, one file, where logic mapped to gates
- In future stages will see clock tree in synthesis files, but not yet, created w/ netlists
- Inside my_base.sdc(referenced in pre_sta.conf): (define clock port,

```
#include "my_base.sdc"
#read_sdc("my_base.sdc")
#read_sdc("slow.lib")
#read_sdc("fast.lib")
#read_sdc("my_base.sdc")
```

clock period)

- Go in vi to change, update SYNTH_DRIVING_CELL to sky130_fd_sc_hd_inv_8
- SYNTH_DRIVING_CELL_PIN to Y

- Vi sky130_fd_sc_hd__typical.lib,(in openlane) search fir inv_8 to get cap load, which need change in my_base.sdc

- Get capacitance = 0.017653, modify in my_base.sdc
 - Switch to my_base.sgc in read line so accessing right settings
 - More skew + output capacitance = more delay
 - SYNTH_MAX_FANOUT description:

SYNTH MAX FANOUT | The max load that the output ports can drive.
 (Default: '5' cells) | , set to 4

(too big fanout might cause delay – net delay w/ 5 fanouts):

- Run_synthesis
 - Report_net -connections _14635_ (the net) – gives driver, Buffer , load pins – fanout increasing slew

```
report_net -connections _14635_
et _14635_
river pins
_43695/_X output (sky130_fd_sc_hd_buf 1)
bad pins
_43697/_A input (sky130_fd_sc_hd_xor2 4)
_43867/_A input (sky130_fd_sc_hd_xor2 4)
_44022/_A input (sky130_fd_sc_hd_xor2 4)
_44032/_A input (sky130_fd_sc_hd_buf 1)
_44186/_A input (sky130_fd_sc_hd_xor2 4)
```

- New area: (slightly bigger)

Chip area for module '\picorv32a': 325893.808000

```
tns -17.52  
wns -1.87  
[TNE0]: Synt
```

- New slew (much smaller) 
 - Clack now -1.87 (desired time to arrive - actual time)
 - Sta pre_sta.conf - where all info of individual pieces slew is stored
 - In this file, we see buffer 1 adding a ton of delay: (line of increasingly bad slew - ability to change output based on input)



- To fix: replace_cell_41882_sky130_fd_sc_hd_buf_4 (area grows, but should be less delay)

- Report_checks -fields {net cap clew input_pins}(fields we want to see) -digits 4(expand after decimal)
 - Slack goes down to 1.4421 (upsizing cell decreases delay)
 - Go through replacing long delay buffers w/ bigger buffers (signif. Timing betterment - just one more (44332) and -1.053 slack)
 - Violation unless + value for slack, so keep upsizing
 - Have to make sure openlane flow has new netlist post these buffer changes so,

```
write_verilog [-sort] [-remove_cells cells] filename  
write_verilog /home/nickson/Desktop/work/tools/openlane_working_dir/openlane/designs/picorv32a/runs/15-09_06-28/results/synthesis/picorv32a.synthesis.v
```

- Run_floorplan (no more synthesis, or will undo all we did)
 - Run_placement
 - Area (increased):

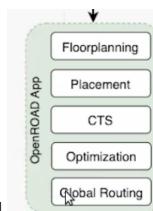
5] 20.000000 : 10880.000000 - 1229120.000000 : 927520.000000
DieArea:

- Params in cts step: (target skew sets global skew, root buffer makes greater drive force, clock tree synth actually does it, tolerance int representing balance of QoR (quality of results) + runtime

```
### CTS
Variable      | Description
-----|-----
[CTS TARGET SKEW] | The target clock skew in picoseconds. <br> (Default: '20' ps)
[CTS ROOT BUFFER] | The name of clock inserted at the root of the clock tree. <br> (Default: 'efsBhd_clkbuf_16')
[CLOCK TREE SYNTH] | Enable clock tree synthesis for tirtonCTS. <br> (Default: '1')
[CTS TOLERANCE] | an integer value that represents a tradeoff of QOR and runtime. Higher values will produce smaller runtime but worse QoR <br> (Default: '100') |
## Routing
```

- Run_cts
 - In cts clock buffers get added, modifies netlist (cts file in synthesis)
 - In openlane/scripts/tcl_commands, one for each status we've done so far
 - Inside cts, procs (functions) name of proc + procedure

- In run_cts, runs openroad (IDE tool in openlane), passes to script in openroad folder to cts.tcl



- All parts of ASIC flow done by openroad
 - $\text{Max_slew} = 10\% \text{ clock cycle time}$
 - $\text{Max_gap} = \text{max gap of output buffer}$

- CTS-CLK_BUFFER_LIST - all clock buffers
- root-buffer
- Max cap - max capacitance of root-buffer
- ‘Openroad’ to launch openroad (note timing diff., db made from lef + def files)
- Steps To create db file:
 - read_lef (place of merged lef –specific_run/tmp/merged.lef)
 - read def file from the CTS stage:

```
% read_def designs/picorv32a/runs/19-03_16-40/results/cts/picorv32a.cts.def
```

- Create db: write_db pico_cts.db
- Read :
 - db
 - verilog files
 - Library
 - Sdc
- Set propagated delay for all clocks’ delay, w/ set_propogated_clock [all_clocks]
- Check timing:

```
% report_checks -path_delay min_max -format full_clock_expanded -digits 4
```

```
Endpoint: mem_data[0].ibus_port[0] clocked by clk
Endpoint: mem_data[1].ibus_port[0] clocked by clk
Path Group: #11
Path Type: min
Delay Time Description
.....
```

- Min:

```
Endpoint: mem_data[0].ibus_port[0] clocked by clk
Endpoint: mem_data[1].ibus_port[0] clocked by clk
Path Type: max
Delay Time Description
.....
```

- Max:

- TritonCTS only optimizes once corner, but other libraries previously used do timing analysis based on min + max corners –inaccurate
- So need to leave + relaunch openroad to check timing for typical corner

```

% exit 1
%
% openroad 2
OpenROAD 0.9.0 1415572a73
This program is licensed under the BSD-3 license. See the LICENSE file for details.
Components of this program may be licensed under more restrictive licenses which must be honored.
%
% read_db pico_cts.db 3
%
% read_verilog designs/picorv32a/runs/19-03_16-40/results/synthesis/picorv32a.synthesis_cts.v 4
%
% read_liberty $::env(LIB_SYNTH_COMPLETE) 5
1
%
% link_design picorv32a 6
[WARNINg ORD-1000] LEF master sky130_fd_sc_hd_tapvprvgnd_1 has no liberty cell.
%
% read_sdc designs/picorv32a/src/my_base.sdc 7
[INFO]: Setting output delay to: 2.4000000000000004
[INFO]: Setting input delay to: 2.4000000000000004
[INFO]: Setting load to: 0.01765
%
% set_propagated_clock [all_clocks] 8

```

- Steps to do this:
- Run check :

```
% report_checks -path_delay min_max -fields {slew trans net cap input_pin} -format full_clock_expanded -digits 4

```

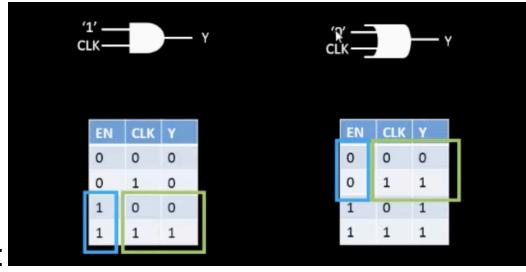
	library hold time	library setup time
1.7707	1.7707 data required time	13.2596 data required time
.....
1.8506	1.8506 data arrival time	9.3159 data arrival time
0.0799	slack (MET)	3.9437 slack (MET)

- Hold + slack time:
- During cts, buffers from CTS_CLK_BUFFER_LIST are added to try and meet skew (can be modified based on requirements)
- As TritonCTS builds clock tree, tried to use all buffers in \$::env(CTS_CLK_BUFFER_LIST) (sky130_fd_sc_hd_clkbuf_1
sky130_fd_sc_hd_clkbuf_2 sky130_fd_sc_hd_clkbuf_4
sky130_fd_sc_hd_clkbuf_8 from smallest to biggest, until skew is met (target = \$::env(CTS_TARGET_SKEW))
- sky130_fd_sc_hd_clkbuf_1 is the most used buffer, can switch this val (\$::env(CTS_CLK_BUFFER_LIST)) (we can do it via lreplace (ex below)) to see effect on STA + area

```
% echo $::env(CTS_CLK_BUFFER_LIST)
sky130_fd_sc_hd_clkbuf_8 sky130_fd_sc_hd_clkbuf_4 sky130_fd_sc_hd_clkbuf_2
%
% set ::env(CTS_CLK_BUFFER_LIST) [lreplace $::env(CTS_CLK_BUFFER_LIST) 0 1]
sky130_fd_sc_hd_clkbuf_2
%
```

- Want to set \$::env(CURRENT_DEF) used by CTS in DEF file as placement's def file rather than that of previous cts. So, set \$::env(CURRENT_DEF) to placement def and run_cts again
- Now only buf_2 clock buffer (before was buf_1)
- Hte + wns are much better bc bigger buffers

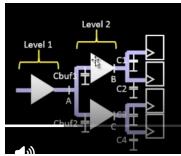
Power Aware CTS:



- Clock in and and or gate:
- And - only when clock on could go through gate
- Or - only opens when clock on – save power

Specific example:

- Level one buffer drives load of 2 buffers, put load of four flops onto one buffer



```

Let us assume C1 = C2 = C3 = C4 = 25fF
How about creating a buffer tree at node 'A'?
Let us assume Cbuf1 = Cbuf2 = 30fF
Therefore, total Cap at node 'A' => 60fF
Therefore, total Cap at node 'B' => 50fF
Therefore, total Cap at node 'C' => 50fF
  
```

- assumptions:

Observations
• 2 levels of buffering
• At every level, each node driving same load
• Identical buffer at same level

- Observations: (2 – level one same to self, level two each same)(3 – all at one level same size)
- Output capacitance for buffer is not constant for whole clock – load at output can vary
- Varying input transition + output load
- Delay tables:
 - To combat these issues
 - Specific range of inputs + outputs tp buffer put on one table – becomes timing model of specific size + threshold voltage:

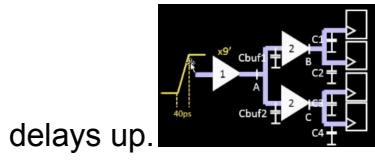
Delay Table for CBUF '1'		Output Load					
		10fF	30fF	50fF	70fF	90fF	110fF
Input Slew	20ps	x1	x2	x3	x4	x5	x6
	40ps	x7	x8	x9	x10	x11	x12
	60ps	x13	x14	x15	x16	x17	x18
	80ps	x19	x20	x21	x22	x23	x24

Level 2 Observations
• 2 levels

(values = delay numbers)

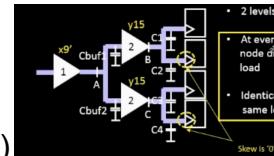
- One for each type of cell, based on sizes + ration of pmos and nmos size (more = less resistance = varying delay) 1(smallest)-10(biggest)
- Build equation from table, not just vals on table

- With input 40ps, what will be clock latency (delay) at end? We must add all



Input Pulse	Output Load				
	200p	300p	500p	700p	900p
200p	x1	y2	y3	y4	y5
400p	y7	y8	y9	y10	y11
600p	y13	y14	y15	y16	y17
800p	y19	y20	y21	y22	y23

- Then take output to put input next buffer:
- Although there are 2 buffers w/ y15 delay, on one branch only one factors into delay

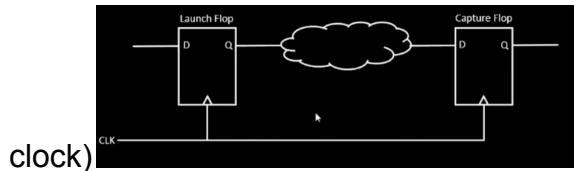


- Skew = 0 at these end points(at same time)
- So $x9' + y15 = \text{latency}$
- Last flipflops only on under certain conditions

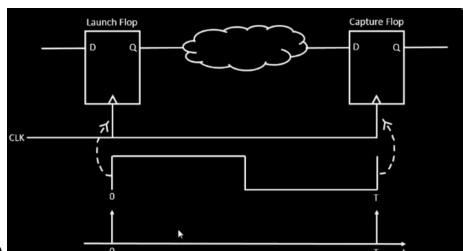
Setup timing analysis (w/ ideal clocks)(single clock):

Specifications:
Clock Frequency (F) = 1GHz
Clock Period (T) = 1/F
 $= 1/1\text{GHz}$
 $= 1\text{ns}$

- specs:
- Launch flop \rightarrow combinational logic \rightarrow capture flop (connected by ideal



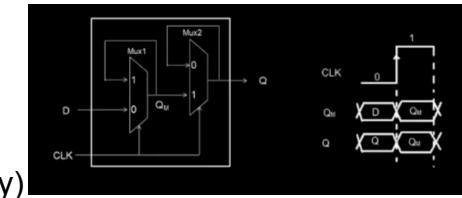
- In one clock cycle, send posedge to launch flop. On 'T'th time, reaches to



capture flop (bottom line on scale of time)

- Assume combinational logic delay of theta. This is less than clock cycle time. as time period goes up, frequency goes down (inverse)
- Image inside of flop. Mux1 delay of D to Q (when clock=0) output moves back to input of mux2 –output doesn't change until clock goes to logic 1

(output mux1 goes back in as input, and goes to second Q – this is a finite

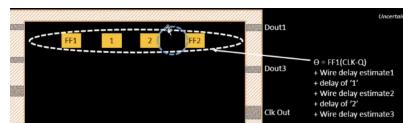


delay)

- Time it takes D to get to first Q subtracted from T (some amount of time before T needed for flop to transmit data within = mux1 delay, setup time) now theta < T-setup
- Some signal (phase lock loops) that supplies clock, has bias, might not provide edges at exactly 0 or T
- There is a window within which clock edge can arrive – temporary variation (jitter)



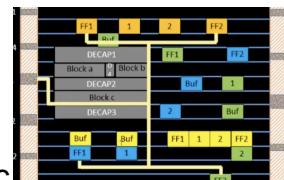
- theta < T - setup - setup_uncertainty (jitter)



- Ex. combinational delay calculations:

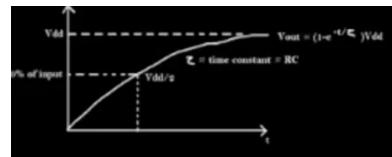
Clock Tree Synthesis:

- Example of skew: , want to be as little as possible
- H-Tree: take clock route, goes midpoint from distances to all next pieces, split in



half and go to midpoint, etc etc – hopefully reduces skew a lot

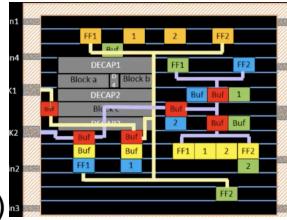
- Due to distances (capacitances + resistances), input to flops might not be same



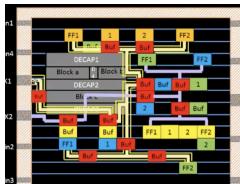
as output from clock

(get slope of waveform, not crisp edges)

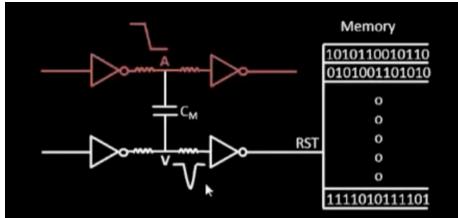
- Add repeaters to mimic + maintain signal (for clocks must be equal rise + fall times) (no skew)



- Protect from cross talk via Clock net shielding - protect from outside world



- Danger of glitch (unwanted transfer of electrical energy between two conductors through electric field made by their capacitance) (capacitative coupling)

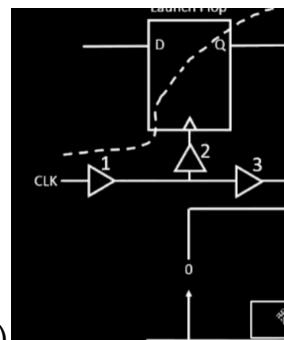


(dip in bottom bc top goes down)

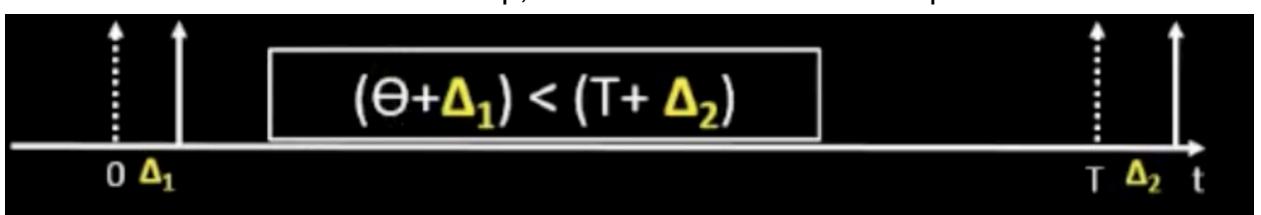
- So add wire of gnd or vdd - won't switch, so protection
- In hesitation bc of glitch, adds delay

Setup timing analysis w/ real clocks:

- Buffers to maintain signal, but add delay (img. At 0 signal supposed to be in flip flop, but will only reach flip flop at buffer1 + buffer2 delay)

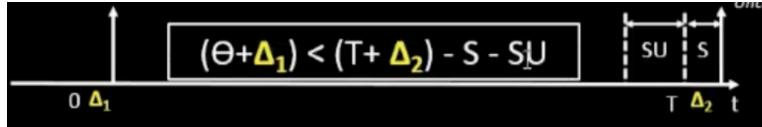


- Delta 1 = time from clk to launch flop, delta 2 = time from clk to capture



- delta1-delta2=skew

- Complete equation, taking into account uncertainty + setup time:



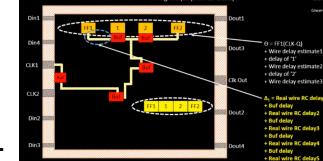
- First half = data arrival time, second = data required time
- Slack (data required time - data arrival time) should be + or 0

Hold timing analysis:

- Hold time = mux2 delay, time to leave flop
- Jitter not considered in equation, bc equal on both sides

$$\Theta + \Delta_1 > H + \Delta_2 + HU$$

- ($HU = \text{hold} * \text{uncertainty}$)
- Slack here is arrival - required time



- Delta 1 and theta values added up based on our model:

To do timing analysis w. Circuit now that clock tree built:

- Type 'openroad' in launcher to enter into openroad (part of openlane)
- Now we can use clock port vars etc, because already in openroad. No need to define like before
- Must create a db from lef and temp file - one time, read anytime you want
- 'Read_lef location' (in specific run, tmp, merged.lef)
- 'Read_def location' (in specific run, results, picorv32a.cts.def)
- 'write_db name' (for presenter, pico_cts.db)
- Must create new db if def changes (meaning components have changed)
- 'Read_db name'
- 'Read_verilog address' (specifc run, synthesis, picorv32a.synthesis_cts.v)
- 'Read_liberty -max \$: :env(LIB_MAX)'
- (bias pin missing from lef warning, but ok)
- 'Read_liberty -min \$: :env(LIB_MIN)'
- 'Read_sdc address' (picorv32a, src, my)base.sdc (or whatever you named it))
- 'Set_propagated_clock [all_clocks]'
- Report_checks -path_delay min_max -format full_clock_expanded -digits
- -2.17 slack, violation bc negative (even though we got it + in synthesis)