WINTECHCON 2018

September 28, 2018

# Divergence Engine: Early prediction of clock-tree divergence at logic-synthesis stage

Sanjana Sundaresh , Atul Garg , Murali Mohan Thota.

Texas Instruments

## Introduction

- Timing convergence, while reducing area and power, is one of the hardest challenges in the physical design of nanometer VLSI chips, especially those which push the limits of frequency entitlement.
- If there is considerable clock tree divergence, we need margins during optimization stage to account for the same.
- Pessimistic margins lead to incorrect frequency entitlement, impacting area and power.
- In reality, each timing path can have different divergence values and applying flat margins is not scientific.
- Accurate path specific margins based on divergence will help in early optimization stages,  i.e. at synthesis and placement, to achieve the best PPAS goals.
- Early identification of critical paths which have high path depth and are highly divergent especially at logic-synthesis stage  and in-time  RTL feedback goes a long way to reduce cycle time.

# Existing method of applying margins

*Current methodology (Flat margins)*

Certain amount of latency and divergence is assumed for the design based on previous experience.

Margins are calculated based on this assumption and applied across the optimization stage.

Example :

```
    Predicted Latency    = 7000ps           Assumed Divergence = 30%
    Launch_clock derate = 1.113             Capture_clock derate = 0.947
    Clock Skew           = 250ps
    Divergence margins = 7000 * 30 (1.113 – 0.947) + 250 = 602ps
```

So if the clock source jitter 150ps , we apply 150 + 602ps as clock to clock uncertainty

**602ps flat margin is applied to all paths in the design as uncertainty till preCTS stage (synthesis + placement ) and removed after CTS. This is in addition to jitter margins*

Limitations of existing methodology -

1. Inability to predict frequency entitlement at logic synthesis stage.
2. Requirement to complete CTS in order to find critical paths and hence delayed architectural feedback.
3. Unscientific iterative margin process impacting PPAS.

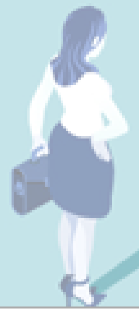
# Issues with existing method

Timing paths in any design can be divided into the below 4 categories

| | | Low divergence + Low path depth | Low divergence + High path depth | High divergence + Low path depth | High divergence + High path depth |
|---|---|---|---|---|---|
| **IMPACT of incorrect margins during optimization (synthesis + preCTS)** | **Low Margins** | Timing met. Design optimized for area and power. | Timing met. Design optimized for area and power. | Timing under optimized at preCTS stage. Timing is NOT met at postCTS. | • Under optimized at preCTS stage hence timing is not met at postCTS.<br>• Redo with extra margins from synthesis.<br>• Late feedback to RTL team on critical paths leading to wrong frequency entitlement. |
| | **High Margins** | Pessimistic margins leading to area and power impact. | • Pessimistic frequency entitlement at preCTS stage.<br>• Wrong paths getting highly optimized at preCTS stage.<br>• Huge impact on power and area. | Frequency target met with area and power impact on other parts of design. | Frequency target met with area and power impact on other parts of design. |
| **Advantages with path specific margins** | | Area and power gain | Accurate prediction of frequency entitlement | Optimize paths from the start | Top paths appear at preCTS stage. Early feedback to RTL team on frequency bottleneck. |

## Proposed Solution

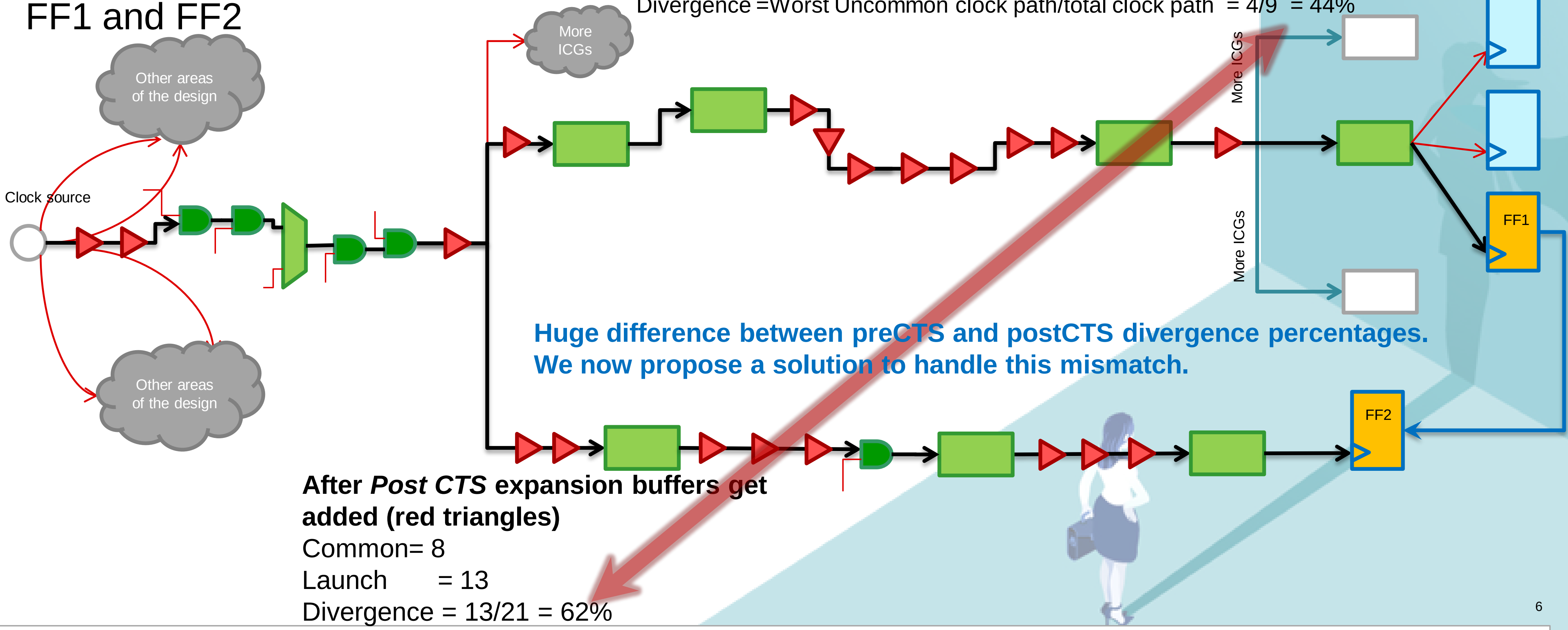
- We present an automatic “design clock-structure aware” method to predict accurate clock tree divergence as early as at “logic synthesis stage” called the Divergence Engine.
  - *Proposed method provides precise divergence margins to accurately constrain all the timing paths of design.*
  - *This enables accurate and required optimization of design and helps provide early feedback on critical performance, area and leakage power.*



# Problem Formulation

Take the case of a typical timing path between flop FF1 and FF2

Actual calculation of clock divergence preCTS stage
Common clock path = 5
Uncommon Launch clock path = 4
Uncommon Capture clock path = 4
Divergence =Worst Uncommon clock path/total clock path = 4/9 = 44%



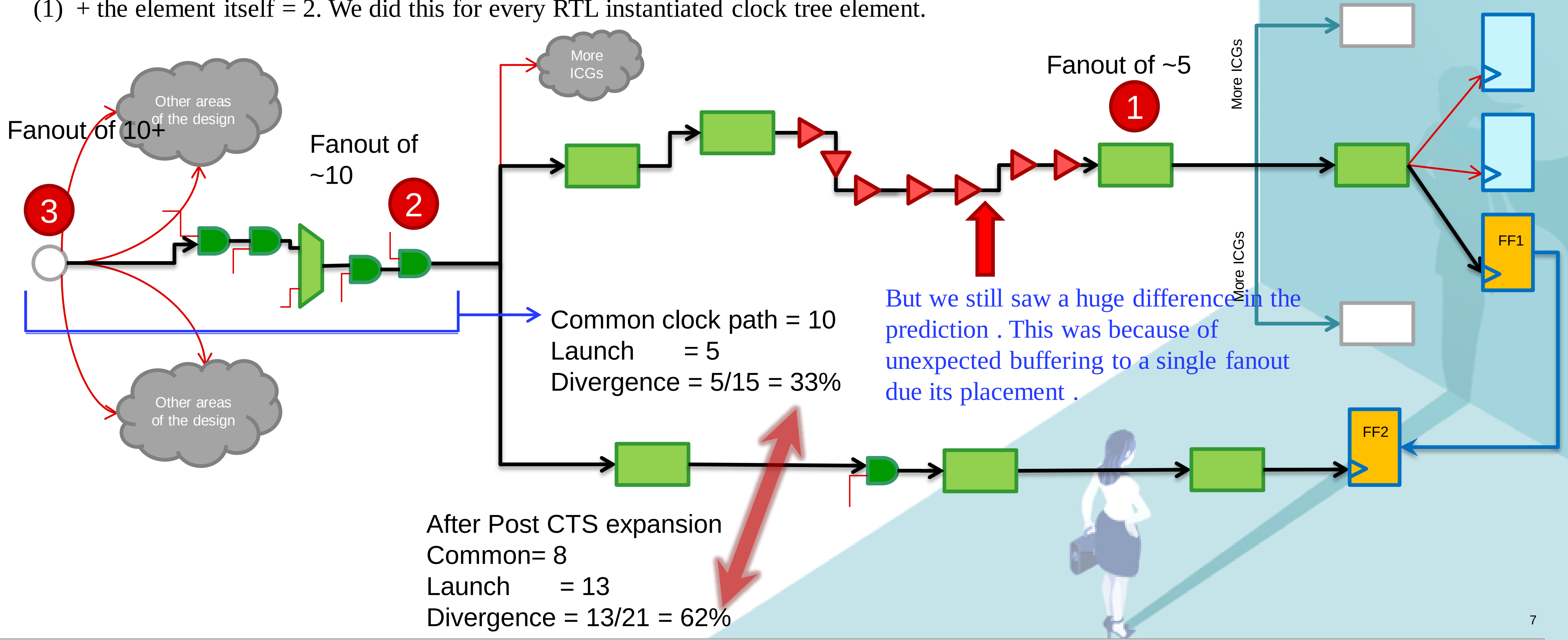
**Huge difference between preCTS and postCTS divergence percentages. We now propose a solution to handle this mismatch.**

**After *Post CTS* expansion buffers get added (red triangles)**
Common= 8
Launch = 13
Divergence = 13/21 = 62%

# Divergence Engine - Fanout based weight estimation

Adding weights by fanout : We assumed that if the fanout of RTL instantiated clock tree element (CTE) was high then CTS tool would put more buffers.

Example, if fanout of CTE was 5 then we assumed CTS would at least put 1 buffer hence weight is 1. So the total count of CTE would be weight (1) + the element itself = 2. We did this for every RTL instantiated clock tree element.



Common clock path = 10
Launch = 5
Divergence = 5/15 = 33%

But we still saw a huge difference in the prediction . This was because of unexpected buffering to a single fanout due its placement .

After Post CTS expansion
Common= 8
Launch = 13
Divergence = 13/21 = 62%

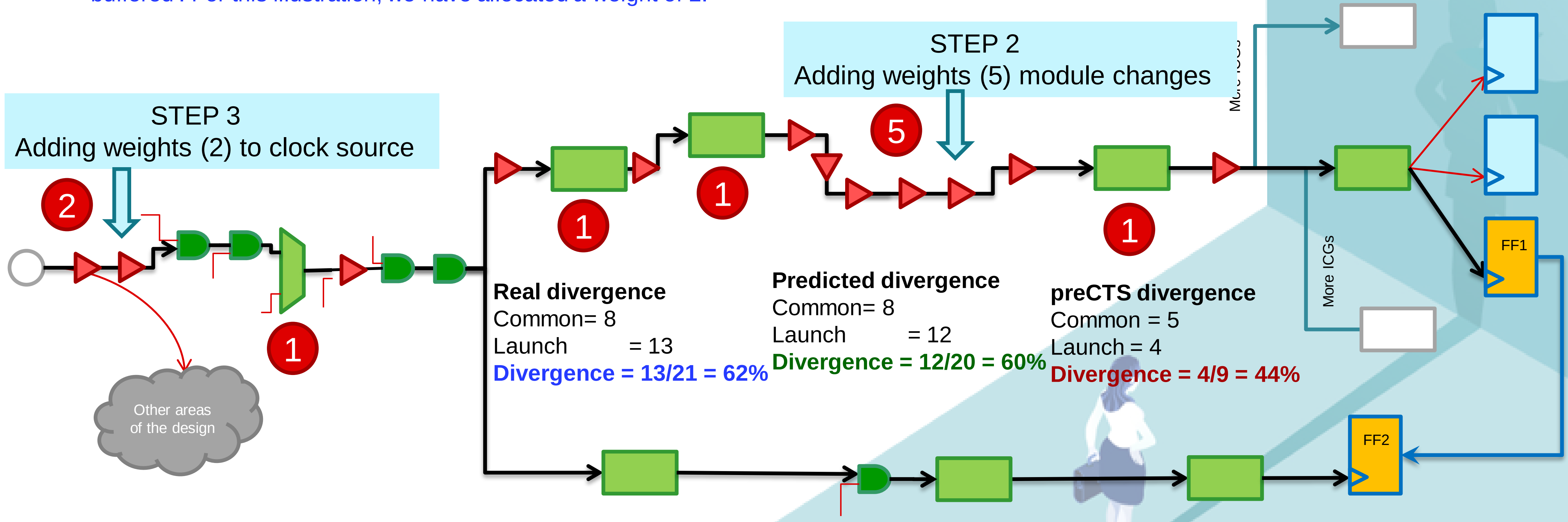
# Divergence engine - Mixed bag approach

*For illustration purposes calculation is shown for only launch clock*

STEP 1 -> Added weights for every RTL instantiated ICG and MUX. In this case, we have assigned a weight of 1 to both.

STEP 2 -> Added weights if the clock path changed modules. As a single module tends to be clustered together and placed far away from another module CTS tool will buffer to meet transition. For explanation, we have assumed 5.

STEP 3 -> Added weights to clock source .Since clock source will always be far away from its branch in floorplan , so it will always be buffered . For this illustration, we have allocated a weight of 2.



STEP 2
Adding weights (5) module changes
STEP 3
Adding weights (2) to clock source
2
1
1
1
5
1
Other areas of the design
More ICGs
More ICGs
FF1
FF2

**Real divergence**
Common= 8
Launch = 13
**Divergence = 13/21 = 62%**

**Predicted divergence**
Common= 8
Launch = 12
**Divergence = 12/20 = 60%**

**preCTS divergence**
Common = 5
Launch = 4
**Divergence = 4/9 = 44%**

**Excellent correlation of divergence percentage between predicted and actual**

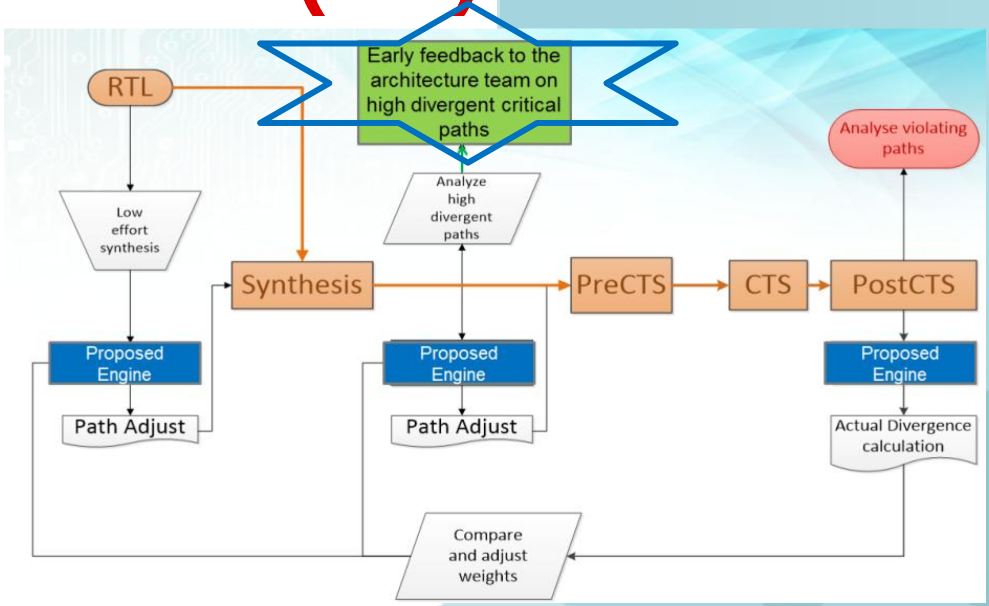
## Concept of assignment of weights.

- As the percentage of clock tree divergence is used to calculate the margins, the exact number of buffers is not required; rather just the correct ratio of worst uncommon clock path and total clock path is needed.
- CTS buffer insertion points are influenced by floorplan and placement hence assignment of weights needs to account for this impact.
- How much weight we actually assign for each of the above 3 key steps for divergence prediction algorithm is based purely on heuristically analysis done on existing postCTS DB for a platform.
- Engine once configured (weights configured) for a platform works with same consistency across different designs.
- Although a design could be first of its kind to a new platform we can run DE by assuming certain weights to begin with. Paths which are predicted to be divergent at preCTS will remain divergent even after postCTS. But predicted divergence percentage vs actual postCTS based divergence percentage could vary .We will have to fine tune the weights based on the postCTS feedback.

# Divergence Engine Implementation (DE)

Flow chart shows different stages where DE can be used

- ❖ DE needs signoff tool DB as input .This DB just has netlist and constraints inputs to calculate divergence .
- ❖ Since only instances and their connections are analyzed, accurate delay calculations are not required. Hence “Low Effort logic synthesis” is fine.
- ❖ Using flop pair to calculate divergence which can result analysis of 20 Million paths for a 100k flop design
- ❖ To solve this problem flop’s immediate fan-in level-1 element is used, that is a RTL instantiated CTE . If we calculate the divergence for one flop pair under two interacting level-1 CTE, then all the flops under them will have the same divergence
- ❖ In case fan-in level 1 is synthesis tool inserted ICG CTE of the flop needs to be mapped to level 2 fanin .
- ❖ Divergence is calculated for each and every CTE pair .



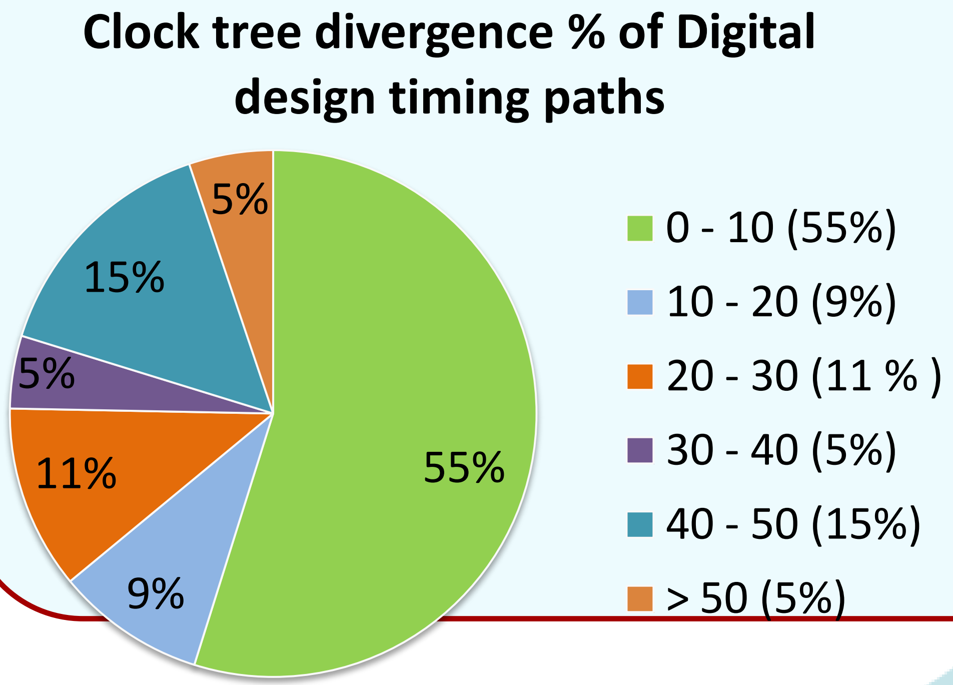
Early feedback to the architecture team on high divergent critical paths
RTL
Analyse violating paths
Analyze high divergent paths
Low effort synthesis
Synthesis
PreCTS
CTS
PostCTS
Proposed Engine
Proposed Engine
Proposed Engine
Path Adjust
Path Adjust
Actual Divergence calculation
Compare and adjust weights

Divergence Engine Implementation Flow chart

*Once the divergence values are estimated, the DE will require values of launch and capture derates, estimated skew and estimated clock latency to calculate the final values of uncertainty to be added to paths*

# Test case details

- Digital design which is area and power critical
- It has been closed using flat margins assuming 50% clock divergence.



***<u>Observations from pie-chart</u>***

- **Not many paths with high divergence exists .**
- If we assume 50% of the design is divergent for margin calculation , it would be valid only for ~5% of the design .
- Pessimistic margins for ~95% of paths
- With the use of margins based of divergence engine we expect to see area benefit .

Texas Instruments

# Margin calculation for Digital Design

## Divergence aware methodology (Flat margins + Path adjust)

- Native tool is unable to take all the path based uncertainty given.
- Hence pick a flat divergence % number which represents divergence for most of the paths and add that as flat uncertainty.
    - In case of this Digital-Design a flat margin of 12% was applied.
- To calculate final flat divergence uncertainty we use the below formula and add that as clock uncertainty in addition to clock jitter .

*Flat divergence margins = 8000 * 12 (1.063 – 0.945) + 250 = 363ps*

*If 150 ps was the clock jitter, set_clock_uncertanity –clock CLK [expr 150 + 363]*

- For paths which have more divergence ie > 12 % add extra uncertainty is in addition to flat margins
- Example
    - Assume a path between (Launch CT element ) ICG1 and (Capture CT element) ICG2 has a predicted divergence of 40%
    - Overall flat divergence is assumed to be 12% . Hence clock to clock uncertainty applied for CLK will be 363ps
- Rest 28% of divergence is specific to paths between ICG1 and ICG2 . Hence this is applied as path adjust at preCTS stage.

# Final results

|  | Traditional method with flat margins only (714ps) | | | |
|---|---|---|---|---|
|  | Area (mm²) | Timing WNS(ps)/TNS(ns)/FEP(no) | Leakage | Run-time |
| **preCTS** | 3.16 | -625ps / -647ns / 2919 | 51.73mW | 13H |
| **postCTS setup opt** | 3.09 | -188ps / -16ns / 384 | 32.561mW | 6H |

Maximum possible area that can be saved is 5.8%

-330ns of timing impact postCTS is undesirable

33% impact on leakage.

| No Margin flow | | | |
|---|---|---|---|
| Area (mm2) | Timing WNS(ps)/TNS(ns)/FEP(no) | Leakage power | Run-time |
| 2.82 | -180ps / -27ns / 567 | 36.7 mW | 13H |
| 2.92 | 289ps / -330ns / 3006 | 43 mW | 6H |

| Reduced flat margins only (363ps) | | | |
|---|---|---|---|
| Area (mm²) | Timing WNS(ps)/TNS(ns)/FEP(no) | Leakage power | Run-time |
| 2.99 | -271ps / -156ns / 1517 | 41.7 mW | 13H |
| 2.99 | -307ps / -173ns / 1535 | 36.29 mW | 6H |

3.5% logic area saving

**8X degradation in slack postCTS opt**

10% increase in LEAKAGE

| Divergence aware margins | | | |
|---|---|---|---|
| Area (mm²) | Timing WNS(ps)/TNS(ns)/FEP(no) | Leakage power | Run-time |
| 3.00 | -184ps / -112ns / 1416 | 49.1 mW | 24H |
| 2.99 | -188ps / -20ns / 547 | 32.8 mW | 6H |

3.5% logic area saving

**Predictable timing picture at preCTS stage**

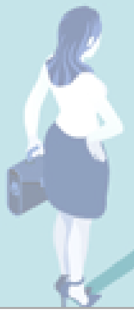
Good leakage

High run time

***Hold optimizations do not affect results as we are not influencing the CTS but just data Path optimization**

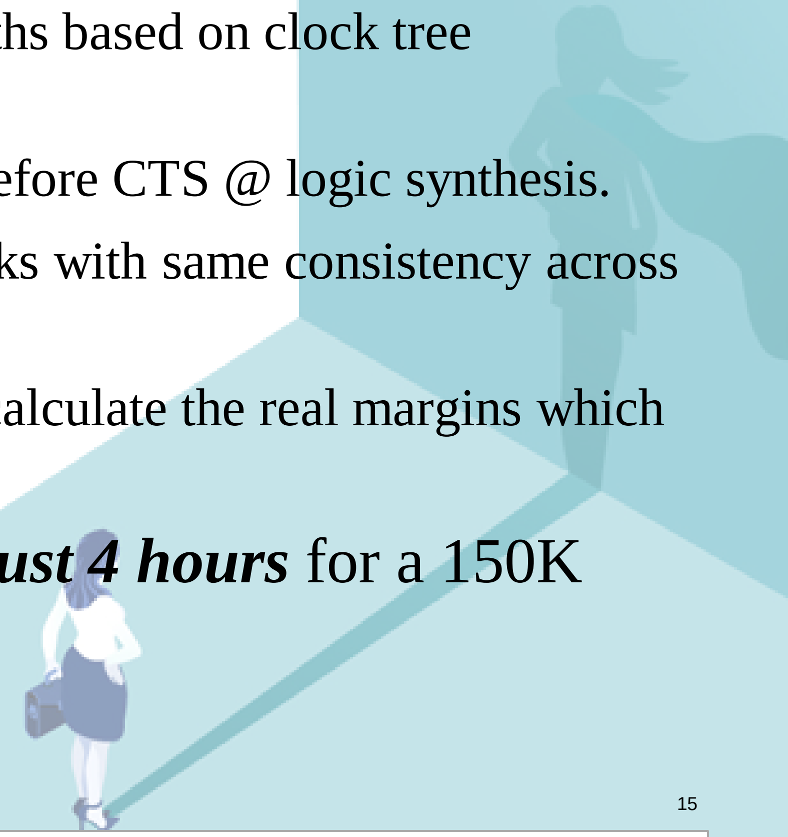
# Conclusions of the experiment

- As explained previously Digital Design does not have many highly divergent paths. Hence the aim was to see if same timing can be met with area recovery.
- Overall area that can be saved on this design without using margins at all is 5.8%.
- With divergence aware margins (DAM) we are able to gain 3.5% on standard cell logic area with no timing impact when compared to flat margins flow.
- At preCTS stage the WNS of DAM flow is more  realistic when compared to flat margins.
- Just reducing flat margins, we are unable to meet timing and pay a penalty on leakage for improved area. Only by using appropriate divergence based path adjusts, we can achieve timing closure and good leakage while gaining the same area benefits.
- Run times of DAM is twice that of flat margin flow.



## Novelty of the Divergence Engine

- Complete and accurate prediction of expanded clock tree divergence at logic synthesis level without any physical information like floorplan.
- Very early architectural feedback to the RTL team on critical paths based on clock tree divergence.
- With realistic margins, we can predict target frequency at well before CTS @ logic synthesis.
- Engine once configured (weights configured) for a platform works with same consistency across different designs.
- On a postCTS db, the engine can derive weights and hence can calculate the real margins which can be feed back to improve optimization.
- Divergence Engine can be run on any netlist with a run time of ***just 4 hours*** for a 150K flop



Texas Instruments

# THANK YOU