Yield-Aware Logic Design Through Cell Fail Rate Analysis

Ritochit Chakraborty, FPGA Silicon Design Engineer
Astrid T. Thomas, FPGA Silicon Design Engineer
Satish Sankaralingam, FPGA Silicon Design Engineer
Mahesh K. Kumashikar, Sr. Principal Engineer

Intel Corp

Agenda

Cell Fail Rate Driven ECO Framework

- Problem Statement
- Cell Fail Rate (CFR)
    - - Yield-Aware Logic Design
    - - Design-for-Yield (DFY) Modeling Prediction
- CFR-Driven Design Methodology
    - - Identify Design Methodology Intercept Strategy
    - - ECO Framework
- Results
- Recommendations

Motivation

Problem Statement

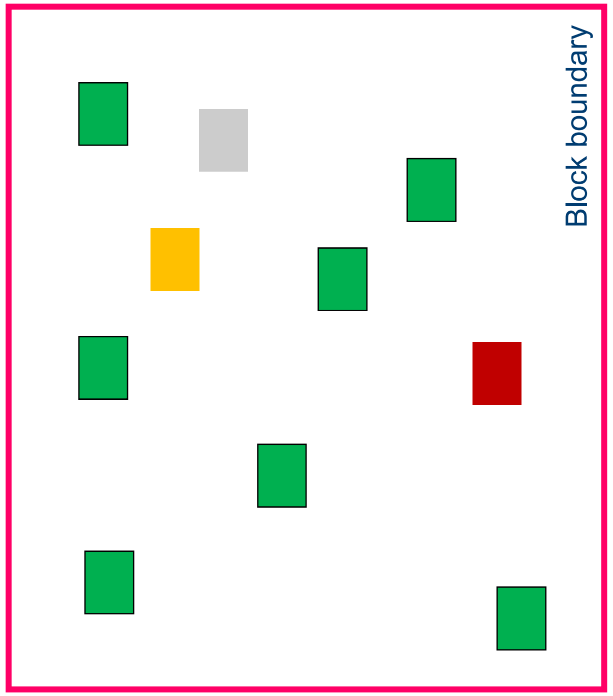
PPA optimization natively does not consider silicon health of standard cells

- Cell characterization does not embed yield information for APR engine to rank cells
  - Sufficient silicon data collection is necessary to identify cells with inferior yield
  - Layout data mining of cells can flag responsible critical areas within each cell

- Silicon yield sightings trigger unwanted ECOs and product steppings in the fab
  - Outlawing defect-prone cells via ECO is limited in scope
    - Heavily modulated by cell usage and timing criticality of paths with cell presence

Cell Fail Rate (CFR)

CFR-Driven Yield-Aware Logic Design



p_g*: prob that cell marked in green fails*
***N:** total #instances of this cell in a block*

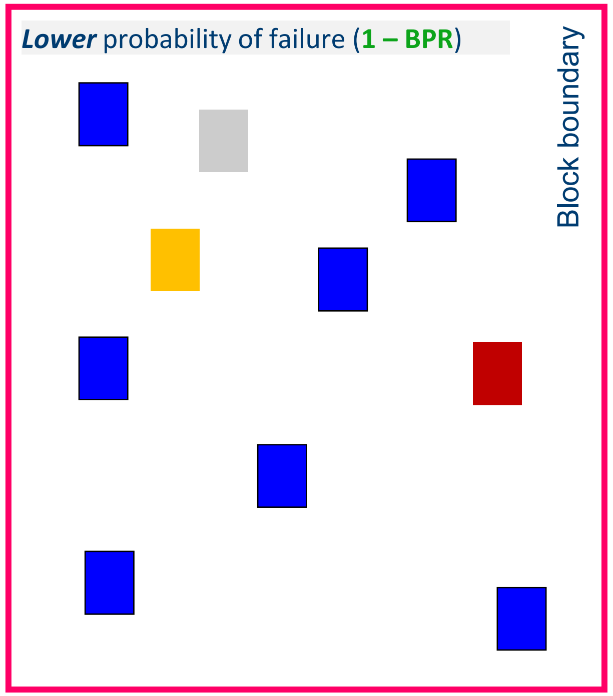
Block passes when none of the *N* cells fail (k=0)

$$\text{BPR} = \binom{N}{k} p_g^{\,k}(1-p_g)^{N-k} = (1-p_g)^N$$

Swap within same family of cells (with **same** logic functionality) but ***reduced*** *cell fail rate* **($p_b < p_g$)**

BPR improves for full-swap by factor: $\left(\frac{1-p_b}{1-p_g}\right)^N$

Lower probability of failure (**1 – BPR**)

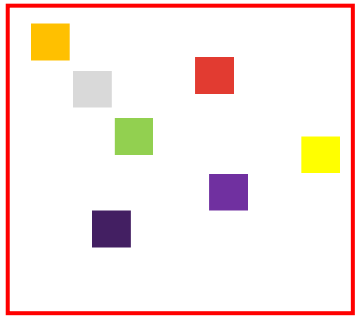


p_b*: prob that cell marked in blue fails*
***N:** total #instances of this cell in a block*

BPR: Block Pass Rate
BFR: Block Fail Rate

BFR = 1 - BPR

Block-Level & Full-chip Yield Improvement

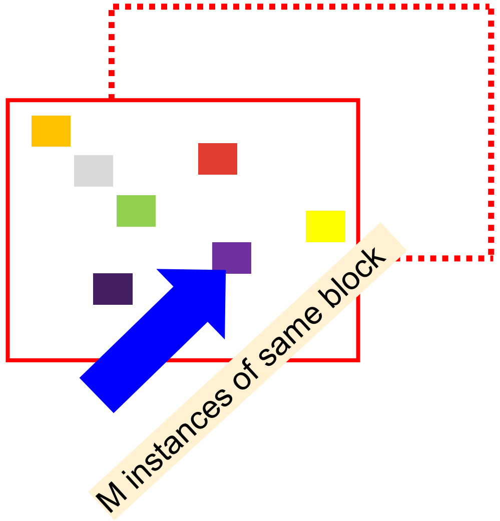


Intra-block cell swaps

1. *Improves block yield* per cell swap with another cell from same family but lower relative CFR
2. Improvement is *multiplicative* across different instantiations of swapped out cells within the block

*For **i** unique cells swapped*

$$p_{fail,block} = 1 - \left(1 - p_{fail,cell_1}\right)^{N_{cell_1}} * \cdots * \left(1 - p_{fail,cell_i}\right)^{N_{cell_i}}$$



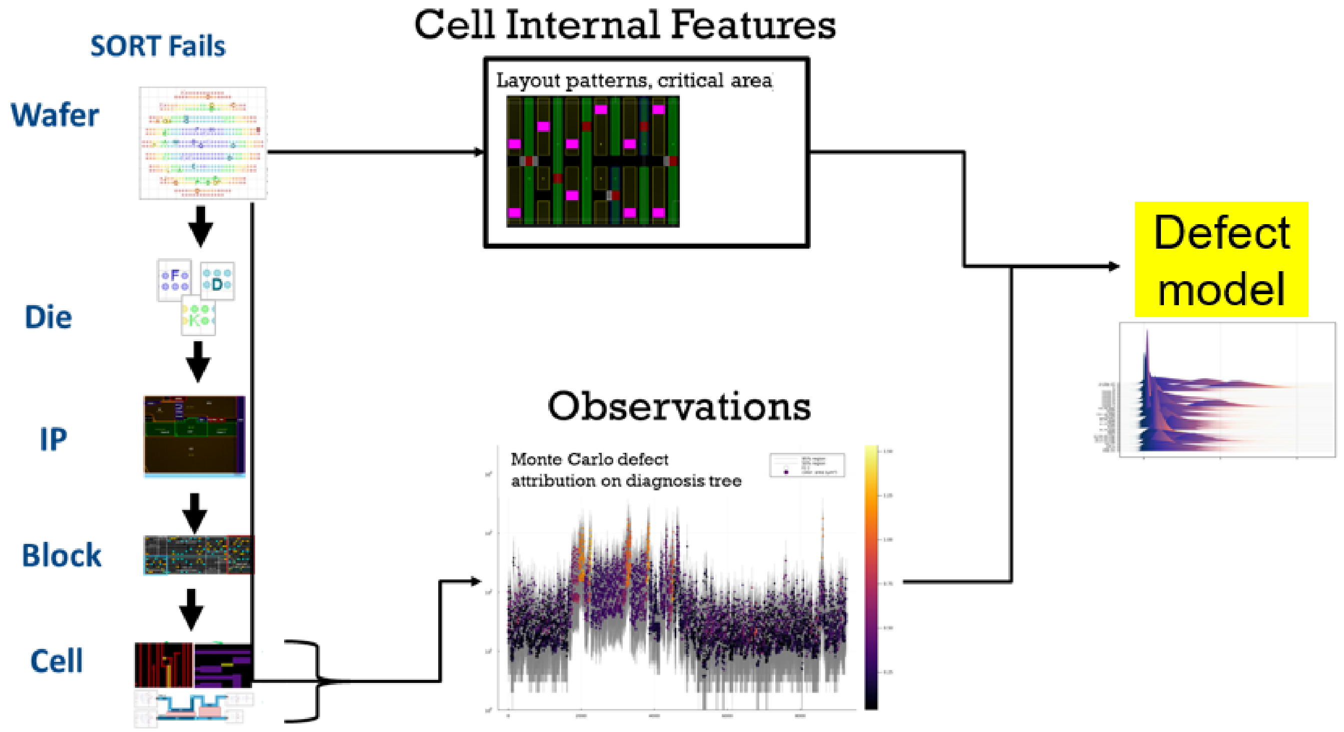
Improvement to full-chip yield accumulates across multiple instantiations of the same block

BPR for M blocks is $\left(1 - p_{fail,block}\right)^{M}$

Design-for-Yield (DFY) Modeling Prediction

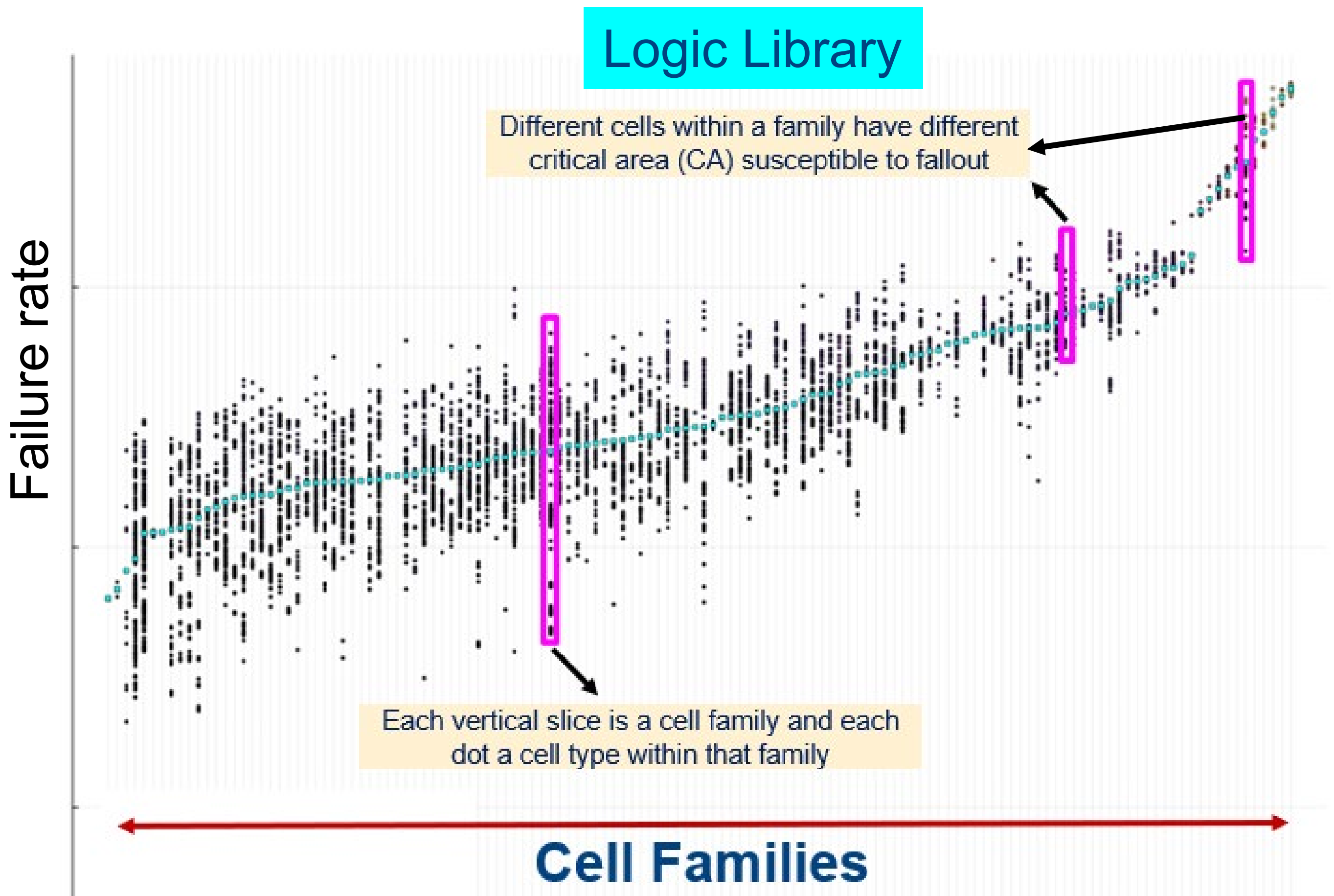
Automating Design For Yield: Silicon Learning to Predictive Models and Design Optimization
Srikanth Venkat Raman et al., International Test Conference, 2020

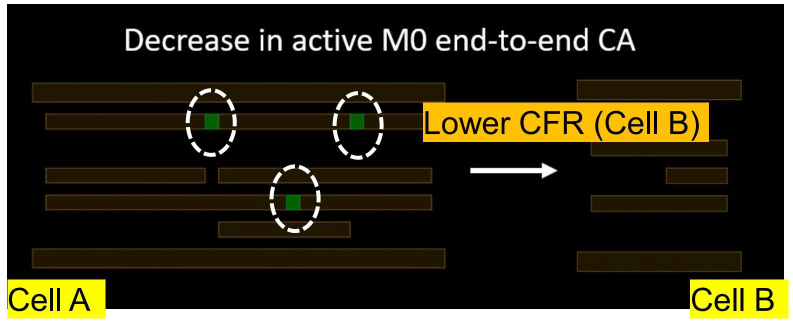
- Sequential cells are identified through scan-shift test fails while combinational cells are identified via scan capture ATPG tests


SORT Fails
Wafer
Die
IP
Block
Cell
Cell Internal Features
Layout patterns, critical area
Observations
Monte Carlo defect attribution on diagnosis tree
Defect model

If cell internal features are identified early in the technology, cells can be redrawn to avoid these features, or a golden cell list may outlaw such cells once PPA implications are fully evaluated

Relative Cell Fail Rates





Can design out higher fail rate cells out of blocks while being PPA-aware

CFR-Driven Design Methodology

Possible Design Methodologies

Different options of solution scope

- Do-not-use (DNU) cell list for logic synthesis through place-and-route

  - Outlaw cells which are most susceptible to yield fallout, but this may impact PPA

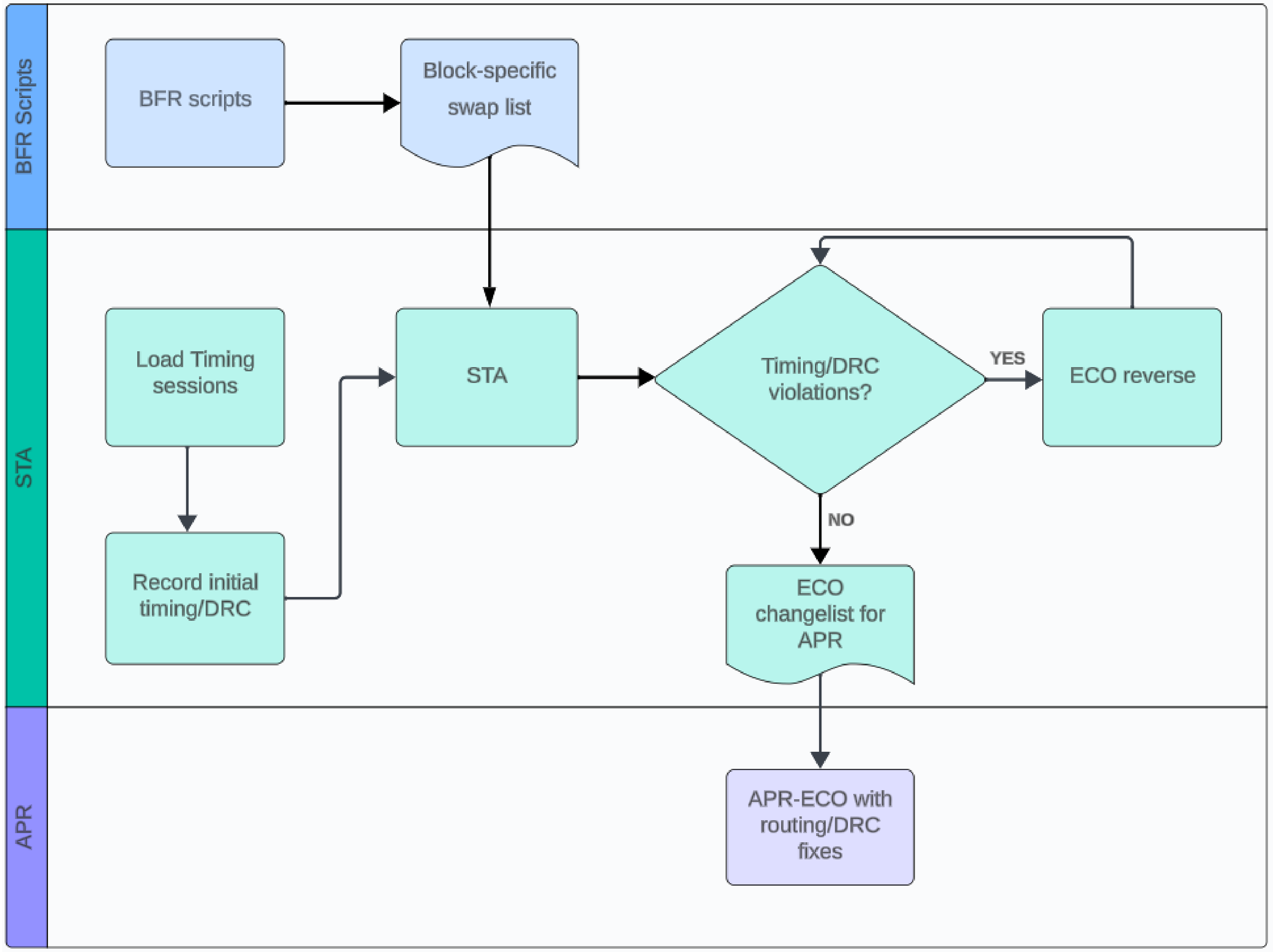
- Comprehend yield as an additional metric natively during logic synthesis through place-and-route

  - Penalize (but not preclude) usage of certain cells, but cell libraries today do not embed any yield information for EDA tools to consume

- <u>ECO framework</u>

  - Cell swaps within the same family with targeted Block Fail Rate (BFR) improvement
  - Managing ECO runtime complexity and additional design rule violations is critical
  - Cell swaps should still be able to meet design Quality of Results (QoR)

ECO Framework Flowchart



Criteria for cell swap
- same family
- same V_t
- same cell height
- +/-1 drive strength
- equal area or smaller footprint

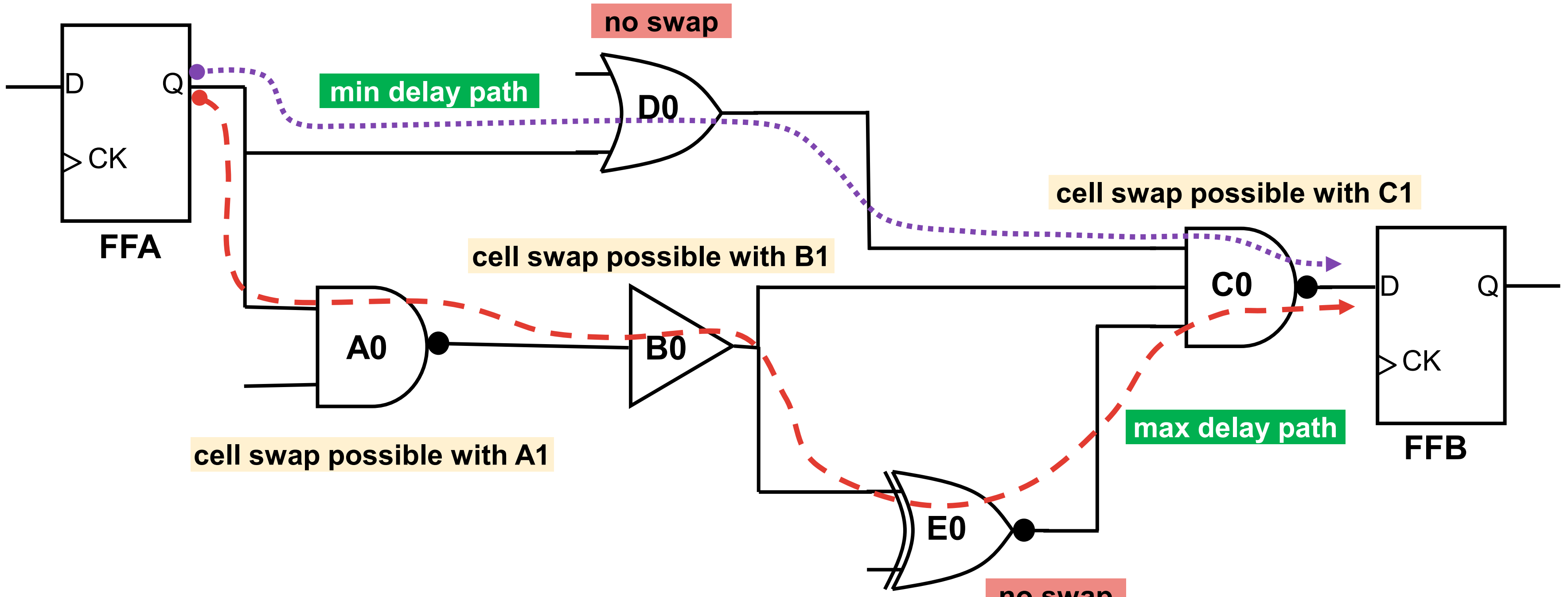
Original cell	Recommended swap cell
A0	A1
B0	B1
C0	C1
D0	D0
E0	E0

Timing path considerations
- exclusion of inout cells
- exclusion of clock network cells
- inclusion of cells only in reg-to-reg data paths

Step #1 – Determining Cell Swap List

Yield sensitivity to cells used in logic design

- Locate lower CFR candidate cells for each cell in the design based on a predetermined cell swap criteria
- Set BFR improvement target (typically 5-10%) to trim the number of unique cells for swapping
  - Reduces ECO complexity and targets highest ROI cell swaps in the design
  - swapping ~100-150 unique cells sufficient


no swap
min delay path
D
Q
CK
FFA
D0
cell swap possible with C1
cell swap possible with B1
C0
D
Q
CK
FFB
A0
B0
max delay path
cell swap possible with A1
E0
no swap

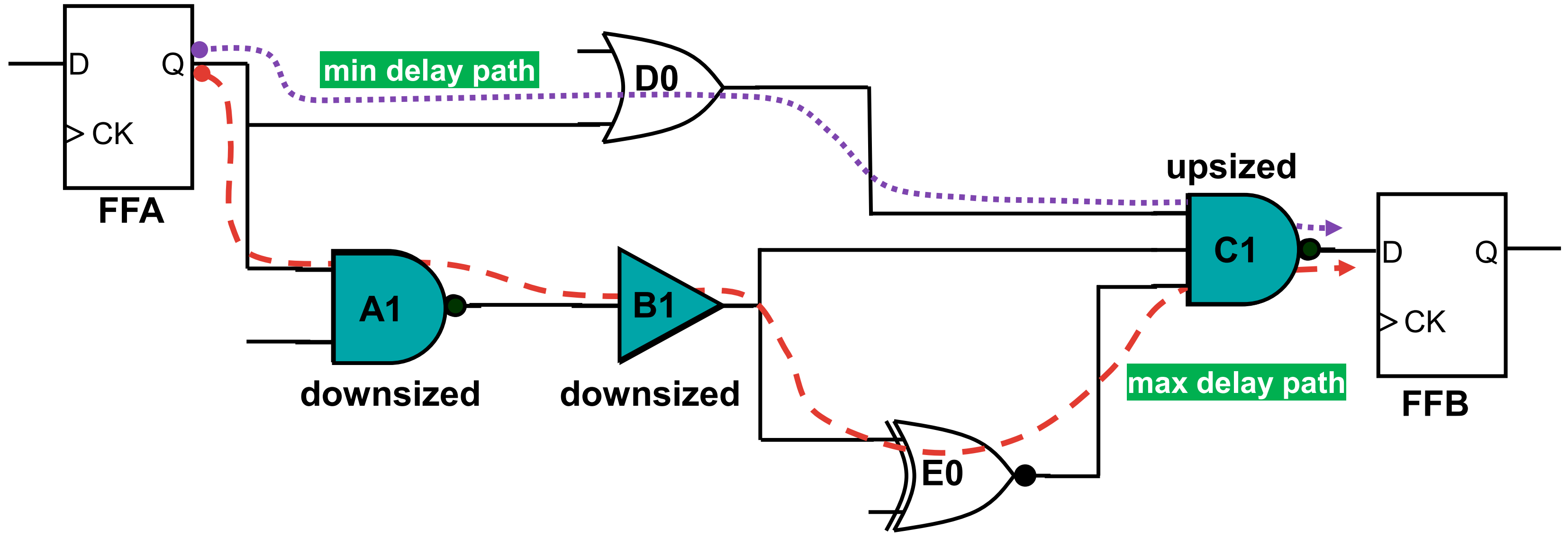
Representative reg-to-reg data path shown here

Original cell	Recommended swap cell
A0	A1
B0	B1
C0	C1
D0	D0
E0	E0

Step #2 – ECO Cell Swap in Multi-Scenario Analysis

Case 1. All cell swaps meet max and min delay in STA

- Timing is met for this reg-to-reg path and algorithm will accept the CFR swaps
  - Additionally, max transition and max capacitance are also monitored

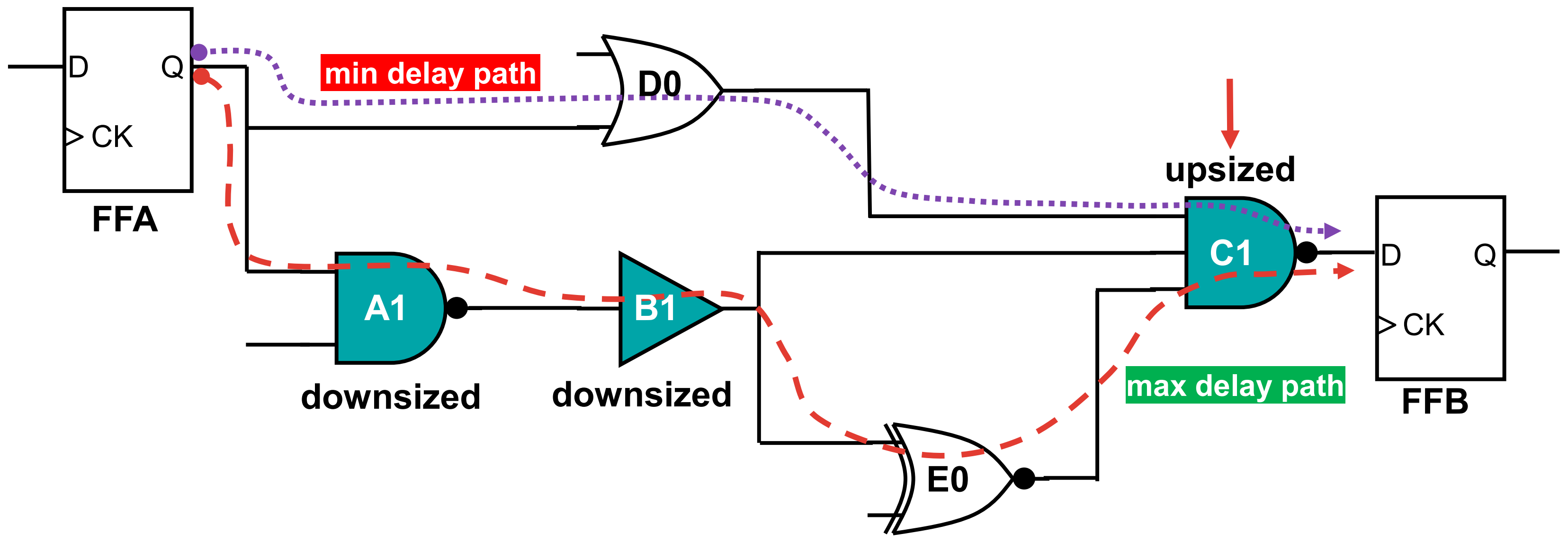


🟩 *Cells which have lower CFR but realize same functionality*

Step #2 – ECO Cell Swap in Multi-Scenario Analysis

Case 2. Max delay is met but min delay violation is introduced in STA

- The next phase of the algorithm will restore the violating timing paths to their original composition via **ECO reversal**

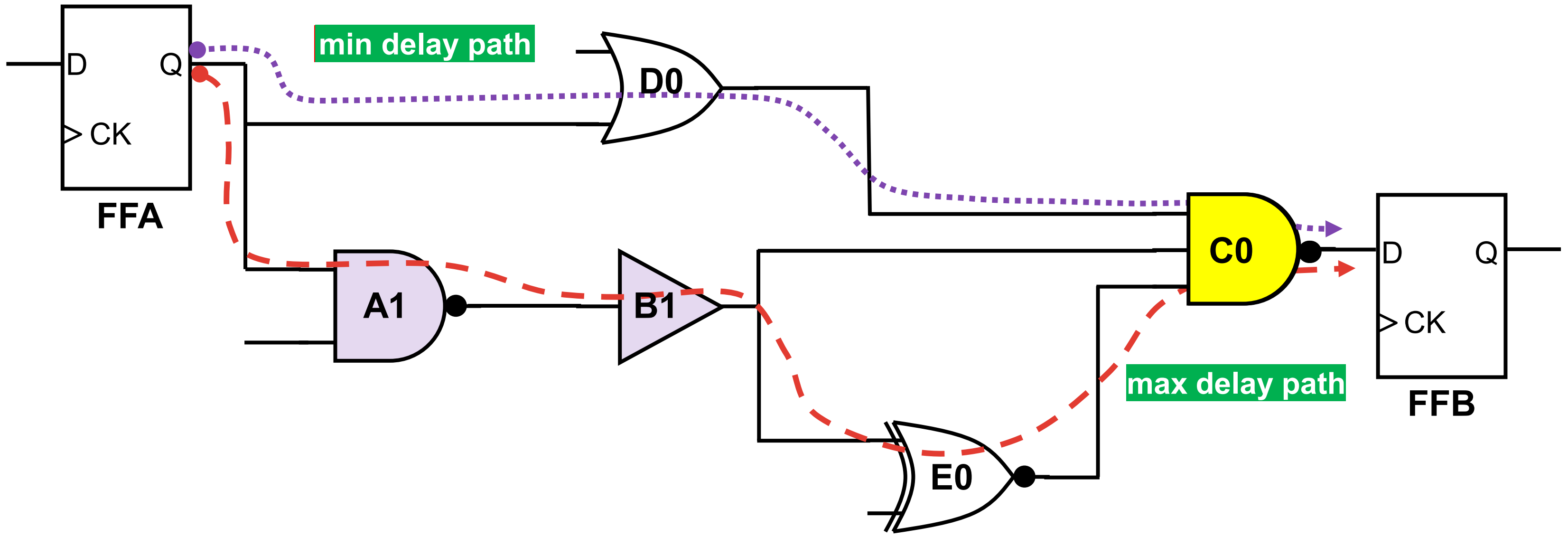


🟩 *Cells which have been swapped*

Step #3 – ECO Reversal in Multi-Scenario Analysis

Case 2a. Fixing only the min delay path is necessary in STA

- Restore entire min delay path to original cell composition
  - ECO flow does not insert additional min delay buffer to fix hold path
- Timing met along max delay path
  - A1 and B1 swaps are accepted



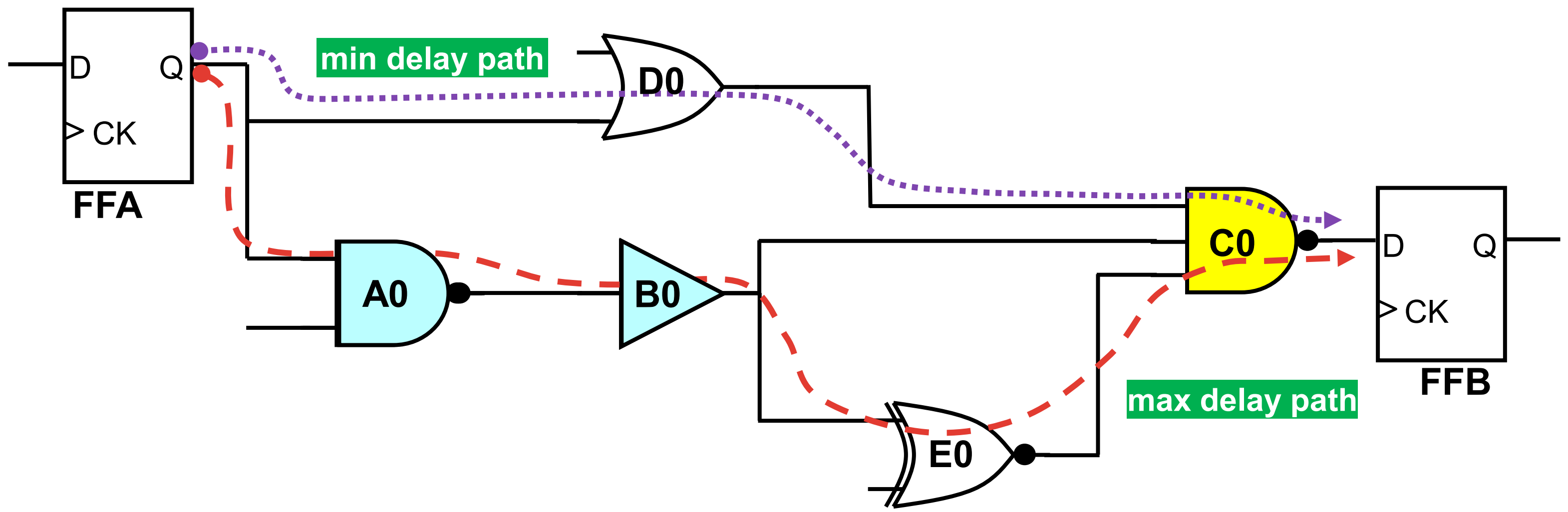
🟪 *Cells which have undergone CFR-based cell swap and need no reversal*

🟨 *Cell which underwent ECO reversal to fix min delay violation*

Step #3 – ECO Reversal in Multi-Scenario Analysis

Case 2b. Fixing min delay path introduces max delay violation in STA which needs a fix as well

- With min delay fix, if the max delay violation crops up again
  - Restore entire max delay path
  - CFR-based swap is rejected for this reg-to-reg path



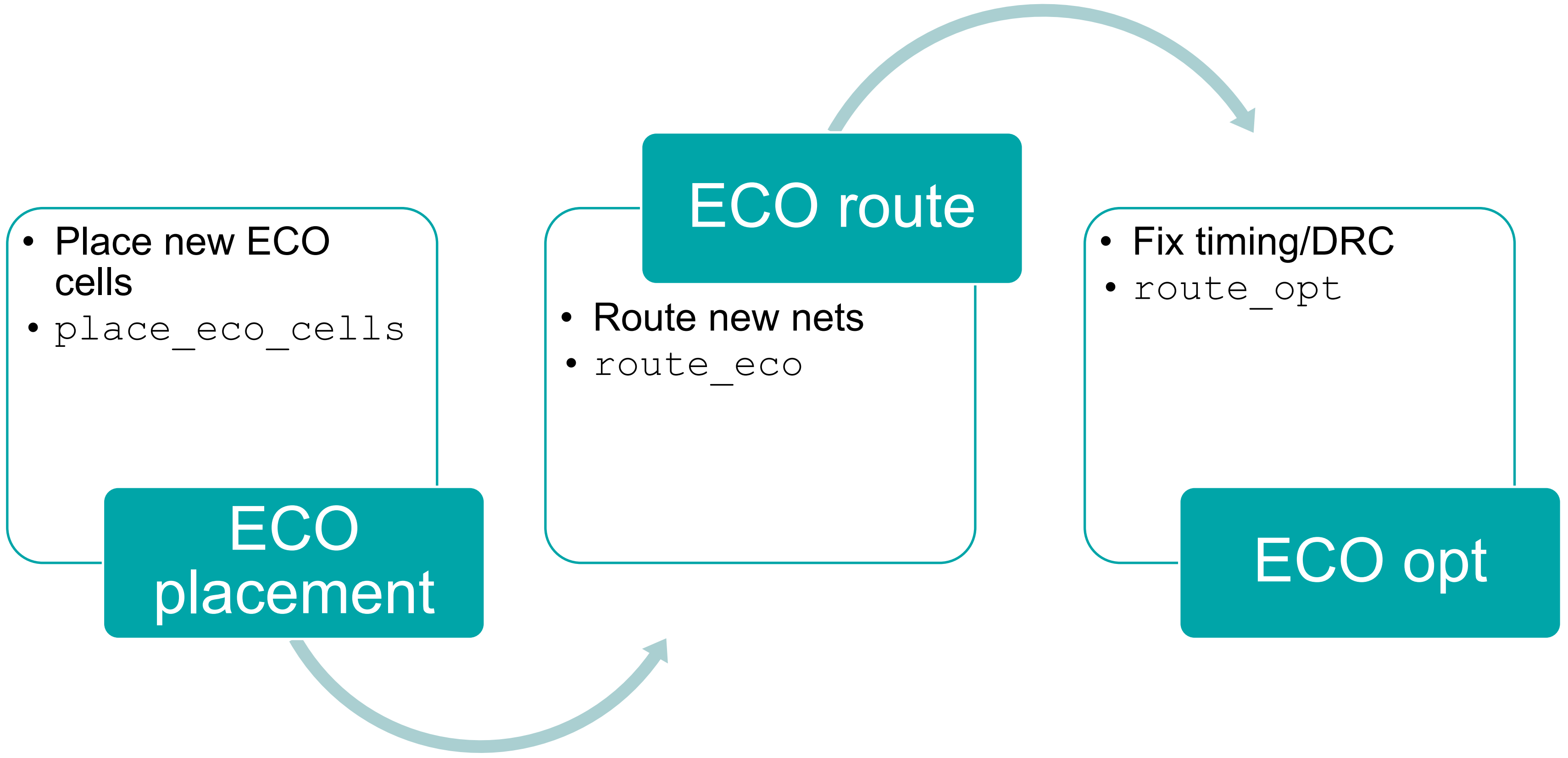
🟦 *Cells which have undergone swap*

🟨 *Cell which underwent ECO reversal to fix min delay violation*

🟦 *Cells which have undergone ECO reversal to fix max delay violation*

Step #4 – APR-ECO

Taking it all the way through with DR/LV fixes



Results and Recommendations

Cell Swap Across Test Cases

- POC on Intel FinFET technology currently in High Volume Manufacturing (HVM)
- Run on 128G/4-core machines employing two virtual workers per machine
  - >150 PT sessions were attempted per block via PrimeTime DMSA *remote_execute* option
- CFR models are modulated by library architecture choices as well
  - Tall height library vs short height library

Design	Total cell count	Initial % cells swapped	# Unique cells swapped	% Cells reverse swapped	Final % cells swapped	Algorithm Runtime
Block #1	**2.99M**	**34%**	143	**2%**	**32%**	**6 hours**
Block #2	**4.69M**	**44%**	69	**4%**	**40%**	**14 hours**
Block #3	**243k**	**17%**	157	**2%**	**15%**	**2 hours**
Block #4	**449k**	**34%**	71	**2%**	**32%**	**3.5 hours**
Block #5	**2.33M**	**33%**	113	**1%**	**32%**	**5 hours**

Yield Improvement from CFR-Driven ECO

Conceptual floorplan demonstrates net yield gain

- In the table below, all logic blocks are modeled as being at 95% yield (5% fail rate)
- Yield improvement could be significant based on number of instances

Area	Neutral
Cdyn	Neutral
Leakage power	±1%

Design	Instance Count	Per instance BFR improvement	Net Yield improvement
Block #1	6	**1.3%**	**0.4%**
Block #2	1	**3.1%**	**0.16%**
Block #3	3	**1.9%**	**0.29%**
Block #4	3	**4%**	**0.6%**
Block #5	1	**4.4%**	**0.2%**

Net yield improvement for Block #1 which has (100-95)%=5% fail rate: $\left[1-\left(1-\frac{1.3}{100}\right)^6\right] \times 5\% = \sim 0.4\%$

Block#1 yield improved from 95% to **95.4%**

Implementation Recommendations

Faster runtime and minimize APR-ECO perturbations

Employ a block specific swap list

Multi-scenario analysis is essential in STA

Focus on register-to-register data paths

Design Methodology Recommendations

Drive optimality via full-blown place-and-route

Translate CFR to cell weightage during gate-level synthesis to facilitate correct-by-construction design to mitigate yield

Clone the cell reversal algorithm into the synthesis flow to swap out cells upfront prior to initiating place-and-route

Summary

- Current place-and-route tools are blind to silicon health of standard cells used in logic design
  - PPA and yield trade-offs happen late in the game leading to unwanted ECOs and silicon steppings
  - Cell Fail Rate (CFR) can alleviate this shortcoming by presenting an insight into logic block yield during PPA

- CFR-based design methodology is a novel approach
  - Leverages silicon monitored fail rates across an entire library of standard cells to preclude cells which pose higher yield risk
  - The approach is feasible for mature processes and new process nodes

THANK YOU

YOUR INNOVATION ***YOUR*** COMMUNITY