# Structured APR: A Hybrid Approach for Efficient Custom Design

Atul Walimbe, Robert Sweeney, Raj Varada

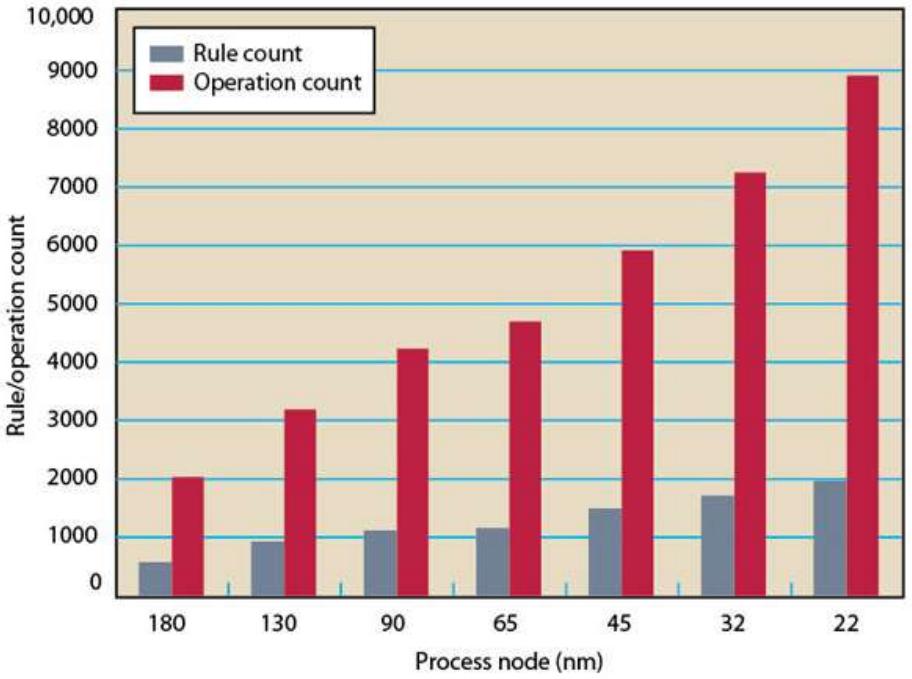
**Intel Corporation**

# Agenda

- Problem Statement
- Exploiting Design Segmentation
- Structured APR
- Case Studies and Results
- Conclusions

# Problem Statement

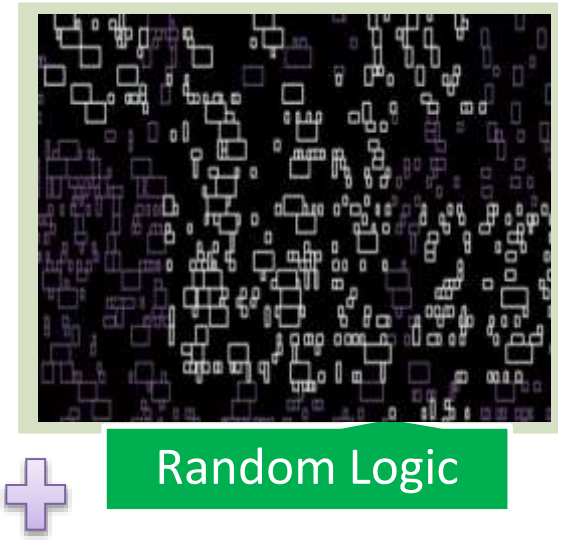
- Performance-critical designs still implemented using custom (non-APR) techniques to provide best QOR
- However, custom design effort no longer fits project schedule due to
  - Design Rule complexity on new process nodes
  - Increase in late, functional logic change
  - Need to develop multiple SKUs for different performance/cost segments

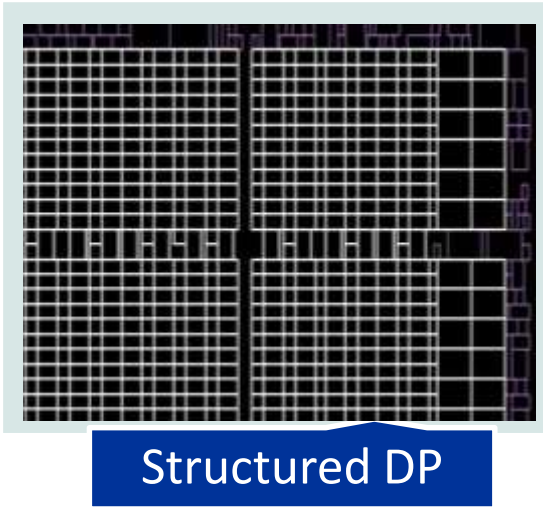



*Source: Electronic Design, March 2011*

# Exploiting Design Segmentation

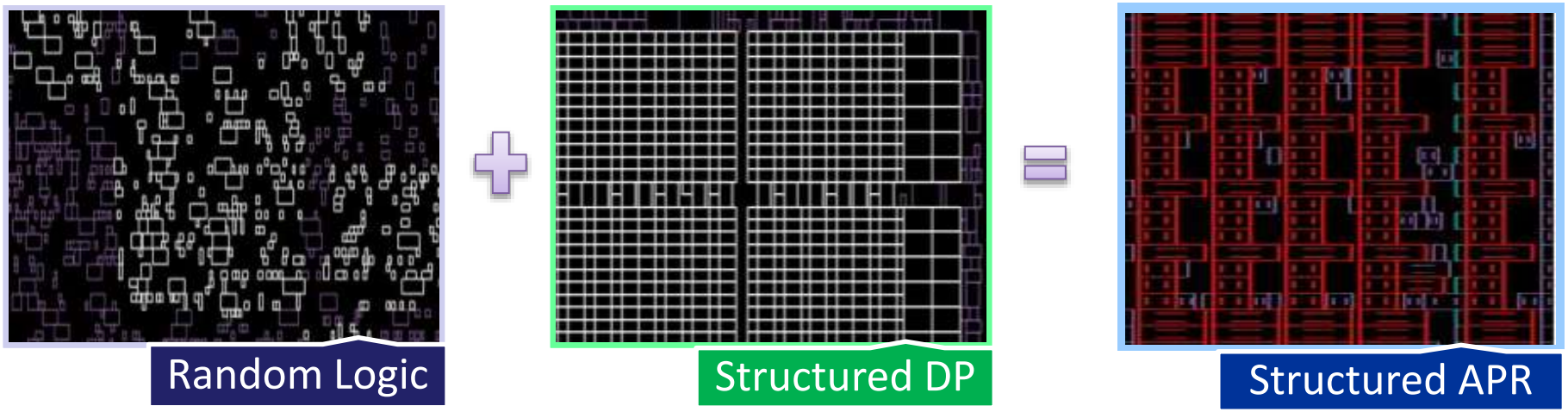
- Typical custom designs contain
  - Control logic
    - Random logic
    - Changes frequently
    - Needs efficient optimization
  - Datapath logic
    - Performance-critical & structured
    - Needs fine-grained control
- Need to co-locate these two for timing and performance



# Exploiting Design Segmentation

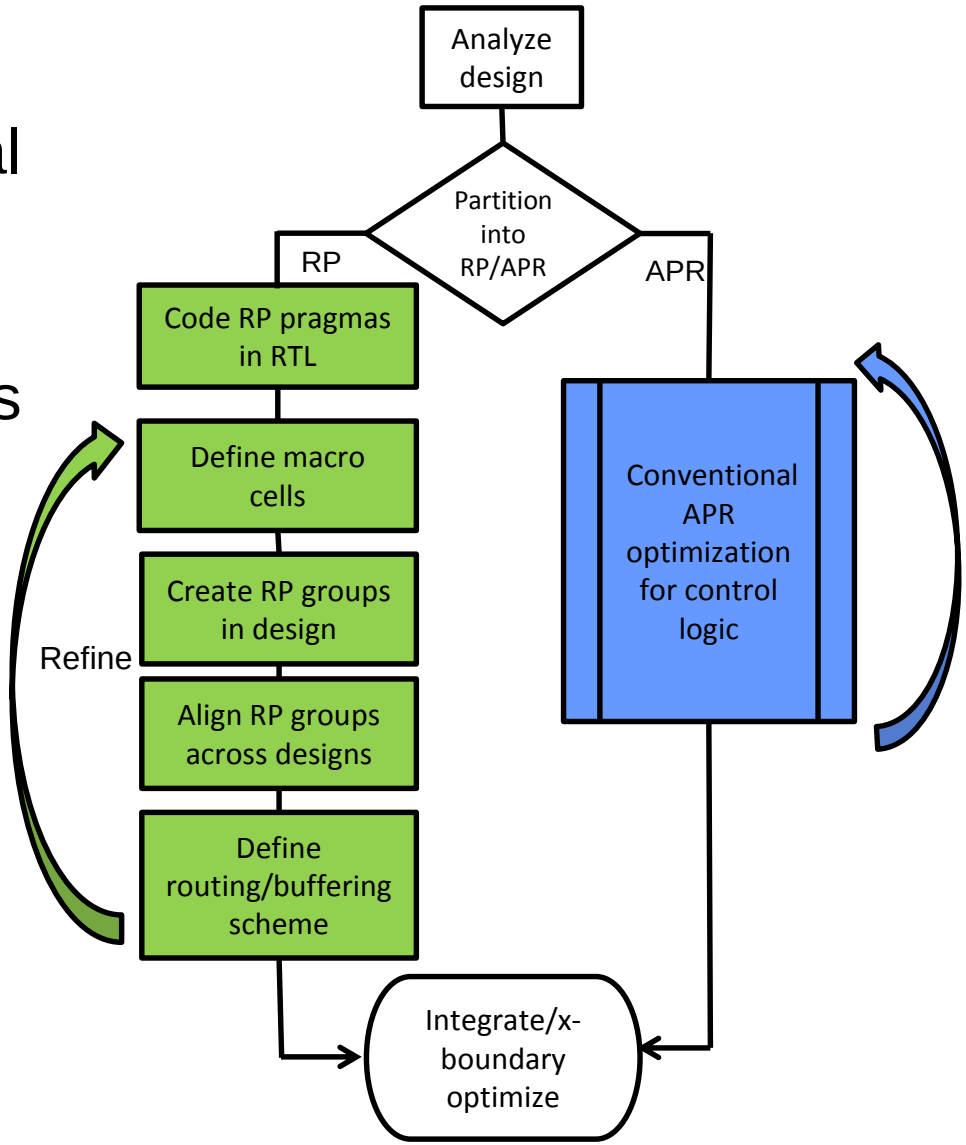
- We propose the methodology of “Structured APR” for such designs



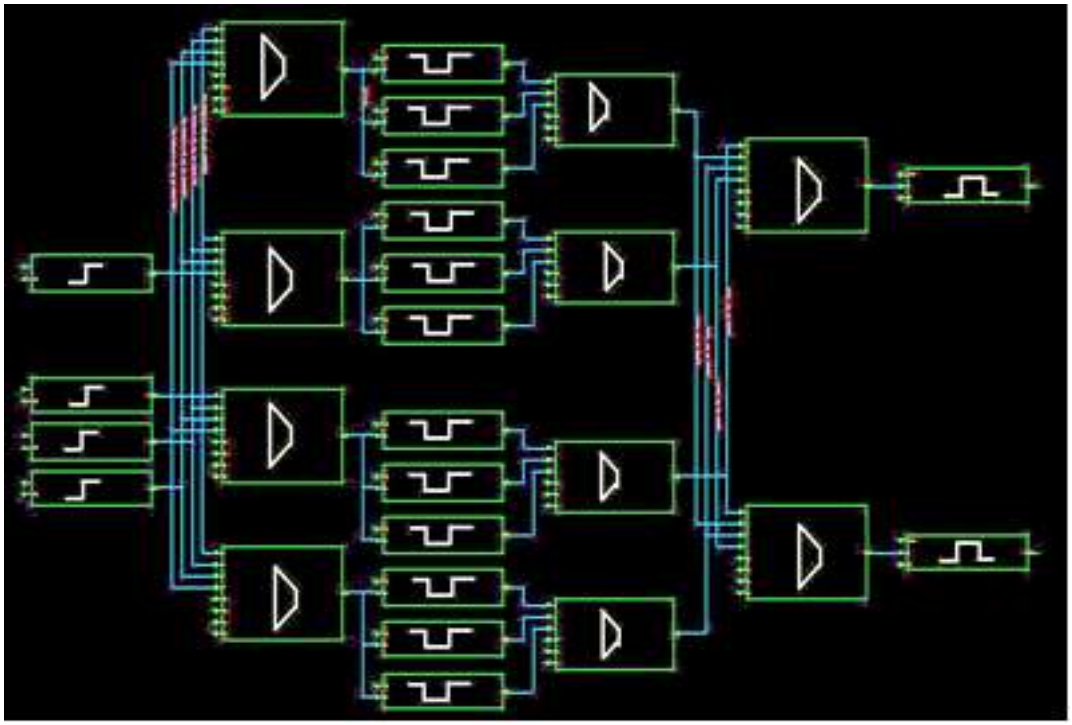

*Goal: Implement both in a single system and achieve “best of both worlds”*

# Structured APR

- Identify performance-critical logic suitable for RP
- Capture relative placement and special routing patterns from original design
- Refine RP construction within and across blocks iteratively
- Implement Random logic with standard APR techniques



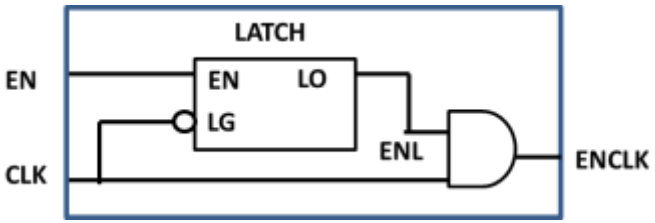
# Case Study 1: Instruction Decoder



- Structured Logic: 150 bit, 4 write, 12 entry, 2 read queue. Each bit constitutes a bit slice
- High speed, latch-based design on 22nm CPU
- Implemented as custom design in prior generations

# Implementation using RP Directives

1. Create “macros” by combining discrete leaf level cells
2. Construct each datapath bit slice by precise “stacking” of the macros
3. Replicate bit slice with RP to create complete datapath
4. Implement and distribute control logic through standard APR techniques

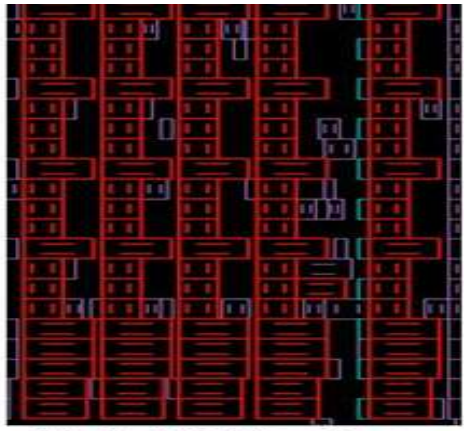



```
module CLOCK_GATE_HIGH ( CLK, EN, ENCLK );
  input CLK, EN;
  output ENCLK;
  wire  net1, net2, net3, net4, n1;
  assign net1 = CLK;
  assign ENCLK = net2;
  assign net3 = EN;

  //rp_group (cg_high_group)
  //rp_fill (0 0 RX)
  <AND function> cg_and ( .clk(net1), .e(net4), .clkout(net2) );
  <LATCH function> cg_latch ( .db(n1), .clkb(net1), .o(net4) );
  <INV function> cg_en_inv ( .a(net3), .o1(n1) );
  //rp_endgroup (cg_high_group)
endmodule
```

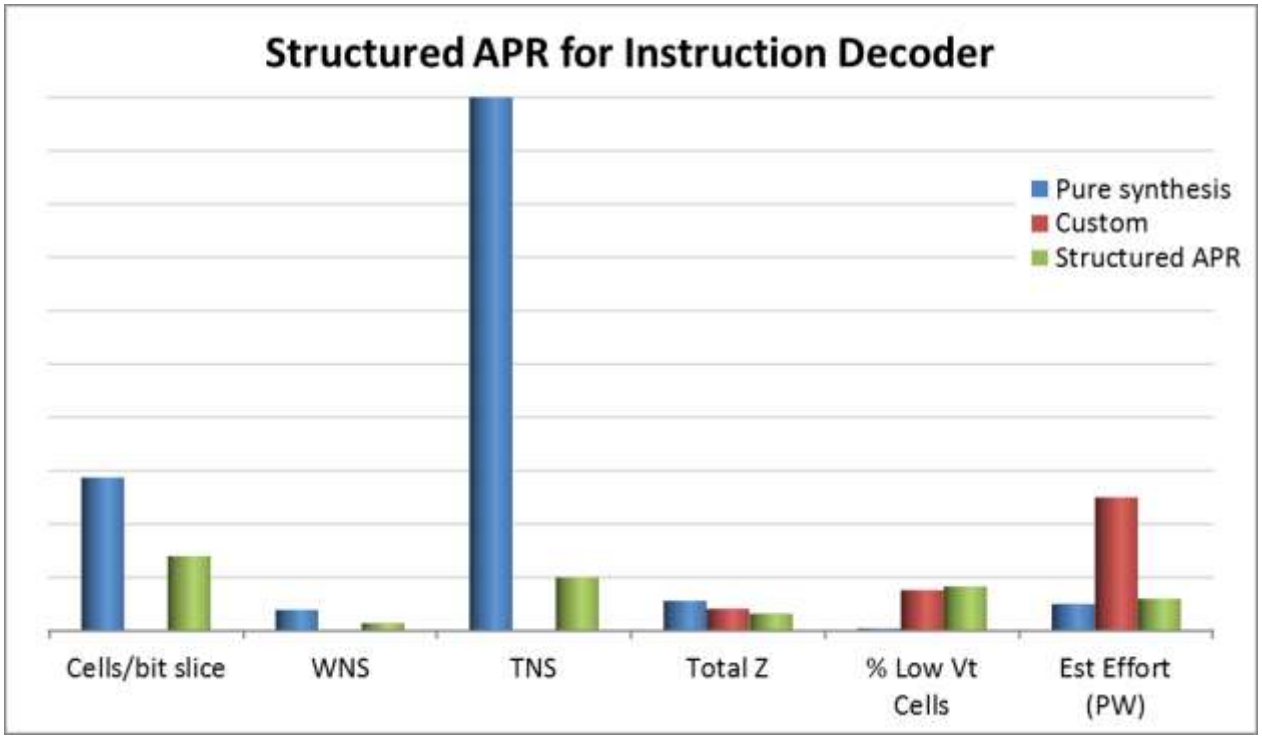
Integrated Clock Gate (ICG) “macro cell

Custom Design Approach



Hybrid Design Approach

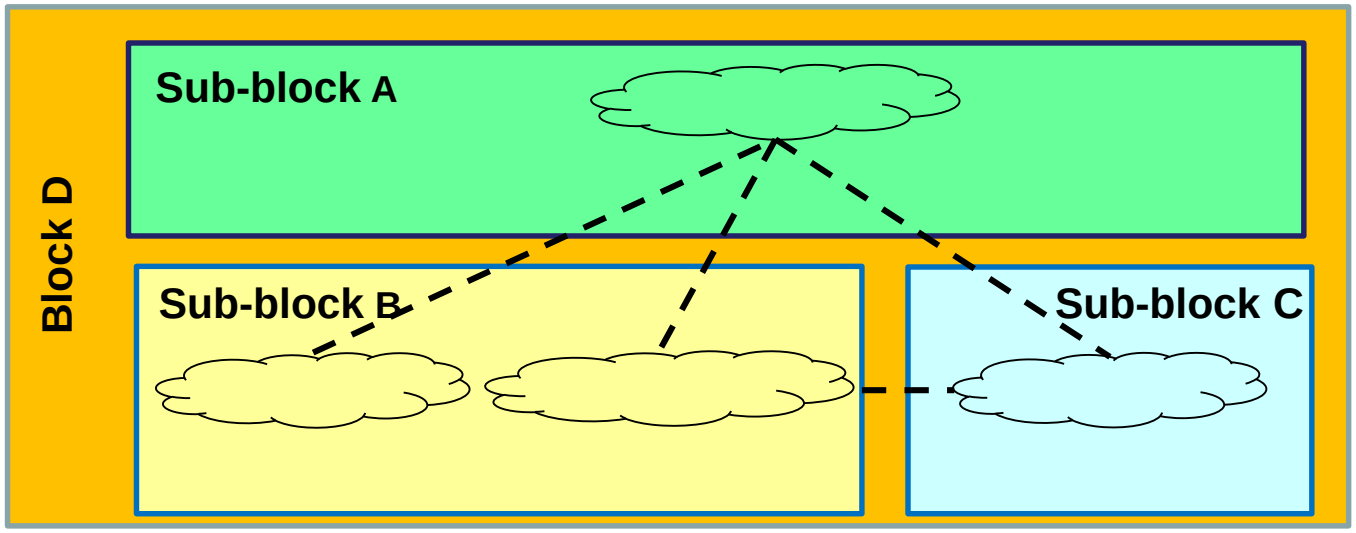
# Results




- Reduced cell count, wire length and power
- Better design convergence efficiency
- Slight degradation in TNS compared
  - Recovered with low effort design techniques

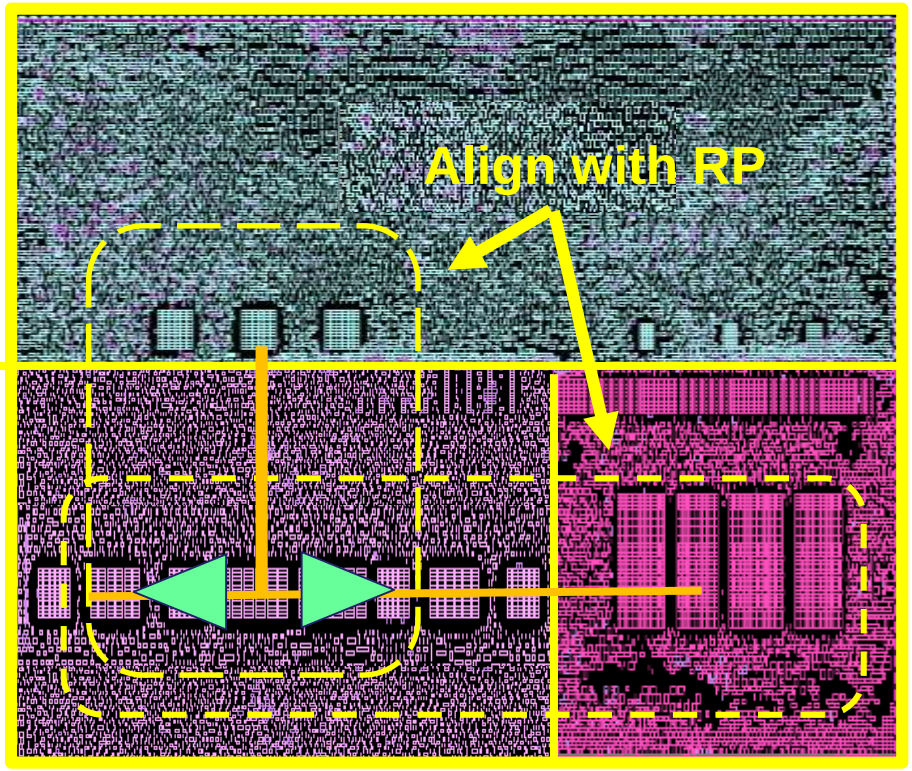
# Case Study 2: Hierarchical Design

- Test case Description
  - High speed CPU design. Implemented as multiple small blocks
  - High manual effort for convergence
  - High rate of logic change (ECOs). Requires fast TAT
  - High logical connectivity between sub-blocks



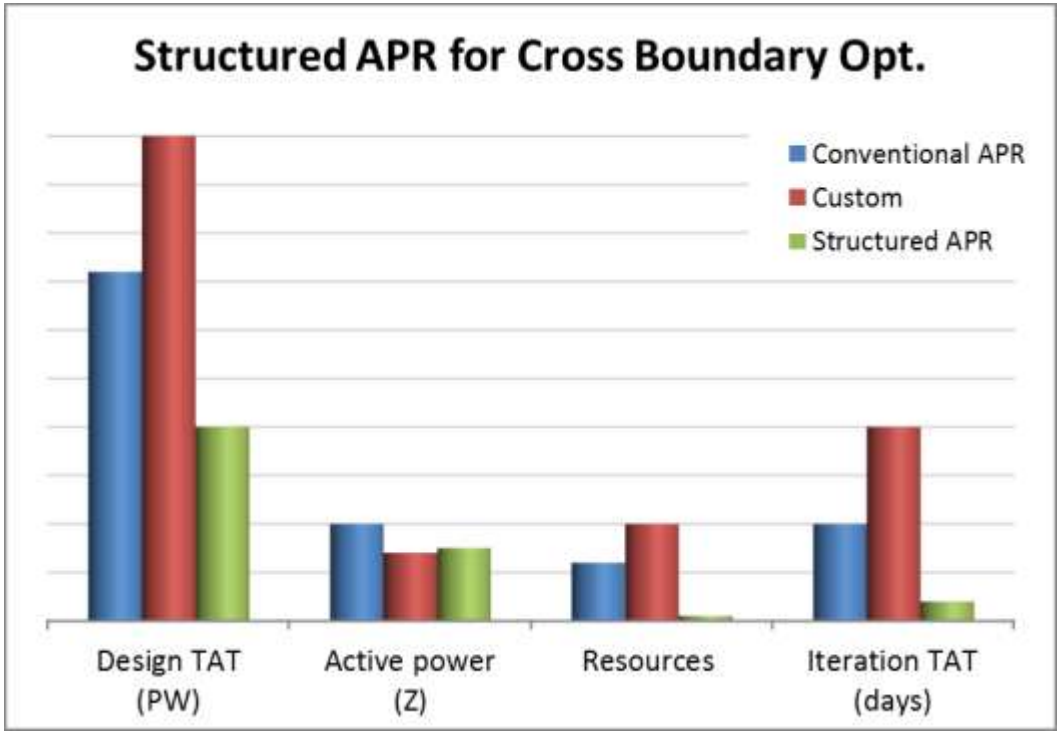
# Cross Boundary Optimization

- Align connected logic
- Freeze sequential cell placements
- Pre-route and buffer nets
- Reduces congestion, improves timing on critical nets




*Reduced manual convergence effort*

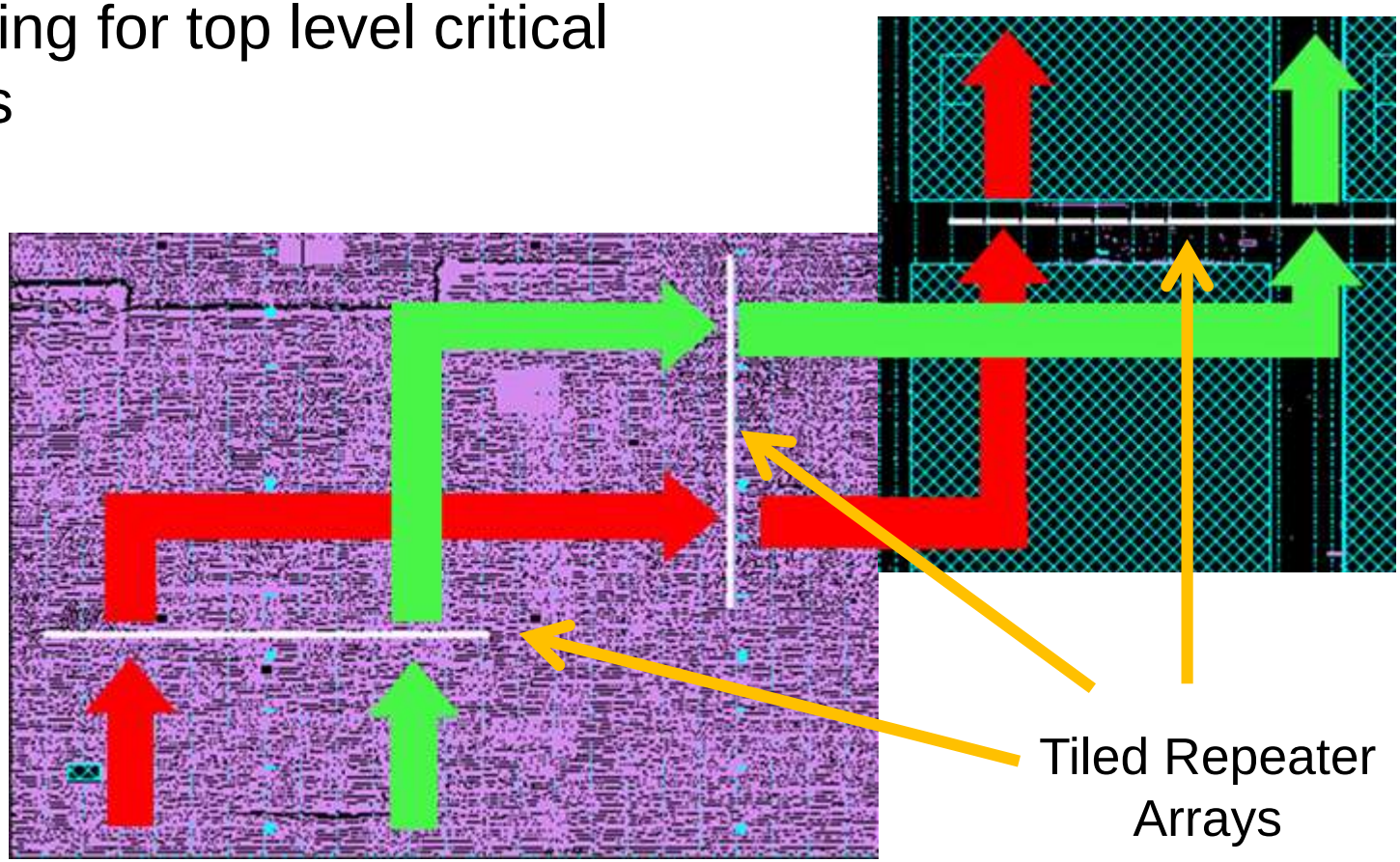
# Results




- Reduced timing convergence effort
- Fewer iterations = reduced resources & improved TAT

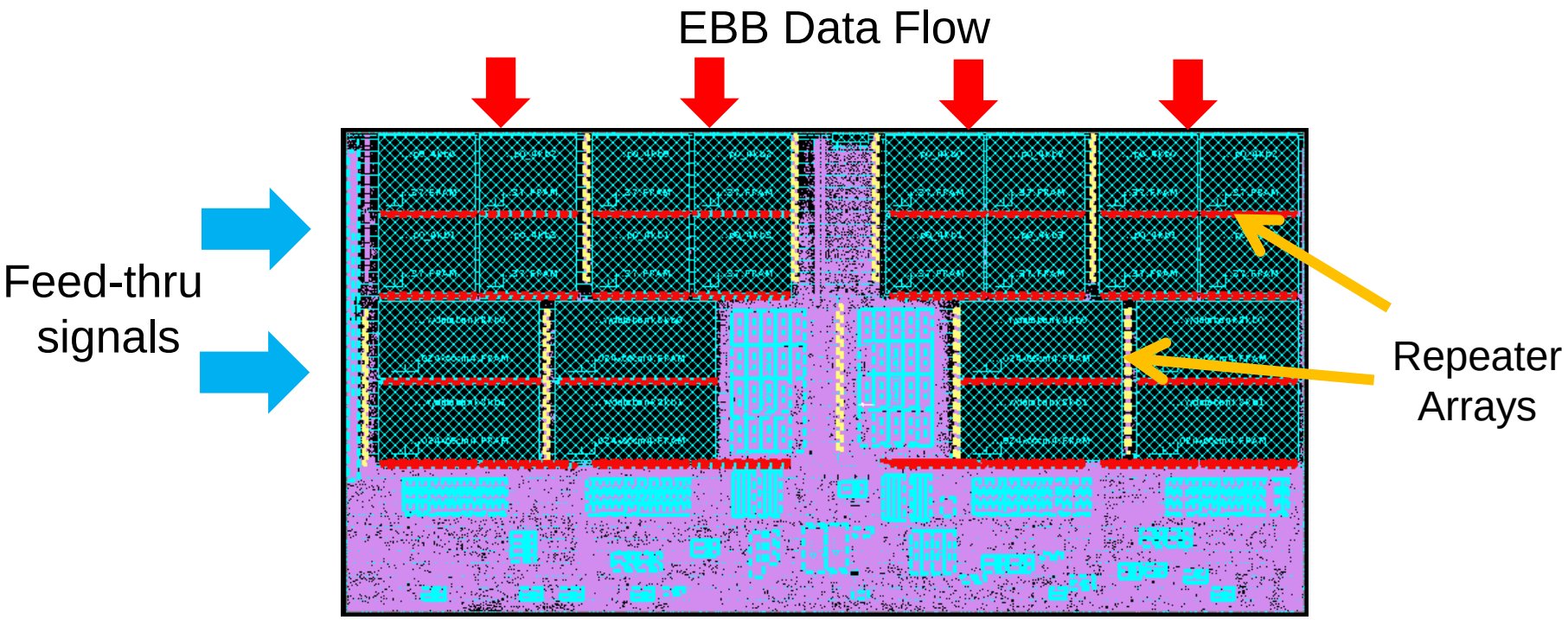
# Case Study 3: Dense Repeater Arrays

- Controlled and deterministic repeating for top level critical signals



# Dense Repeater Arrays

- Alignment of repeaters stages with available tracks enables a routable solution in EBB-saturated region
- Structured repeater placements + priority routing of signals achieves ~50% delay improvement over native tool results



# Conclusions

- Structured APR design approach combines efficiency of APR with fine-grained optimization of custom logic design
- Applicable to a variety of design styles and abstractions
- Improves quality and design closure efficiency over APR and custom design approaches
- Looking ahead: EDA tools already provide the hooks. Why not native “Smart-RP” in the tools?
  - Recognize regularity in logic.
  - Identify RP candidates
  - Auto-structure placement and pre-route

# Further Reading


- Henrik Eriksson, Per Larsson-Edefors, Tomas Henriksson, Christer Svensson ”Full-Custom vs. Standard-Cell Design Flow - An Adder Case Study” In *Proceedings of the 2003 Asia and South Pacific Design Automation Conference*, pp 507-510

- D.G. Chinnery and K. Keutzer, “Closing the Gap Between ASIC and Custom: An ASIC Perspective”, In *Proceedings of the Design Automation Conference*, 2000, pp. 637-642

- Raj Varada, Ragadeepika Kshatri, Andy Spix “Automated Pseudo-Flat design methodology for Register Arrays” in *Proceedings of the Design Automation Conference* 2009

- Steve McKeever, Wayne Luk, and Arran Derbyshire. “Compiling Hardware Descriptions with Relative Placement Information for Parameterized Libraries” In *FMCAD '02 Proceedings of the 4th International Conference on Formal Methods in Computer-Aided Desig*n