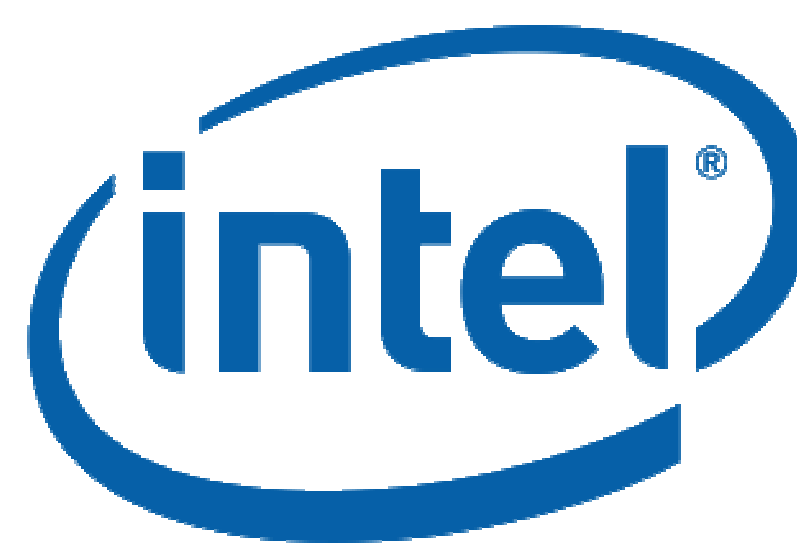


# Re-timing Trade-off in High Performance Graphics Designs

Murali Seshadri, Sharon Martin, Ragadeepika Kshatri, Raj Varada  
Intel Corporation, Santa Clara, CA

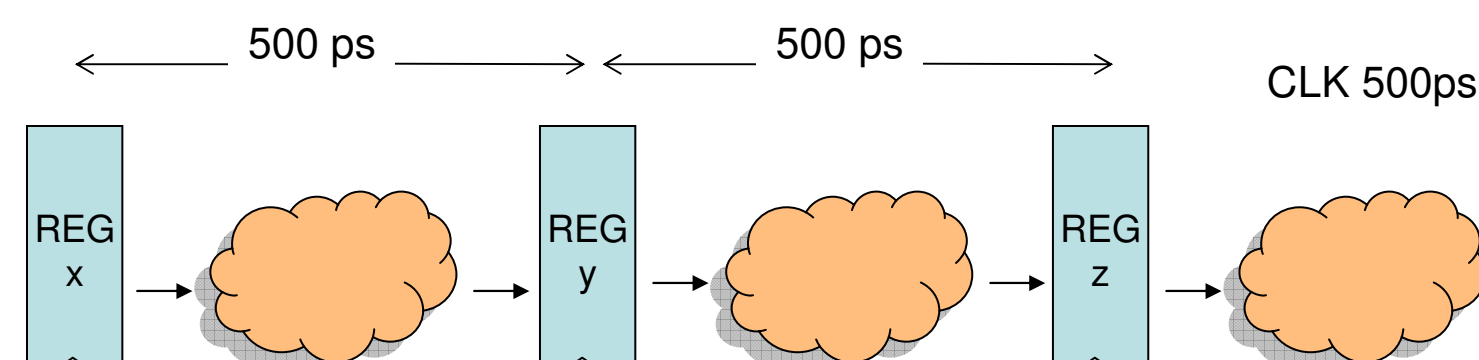


## What is Register Retiming

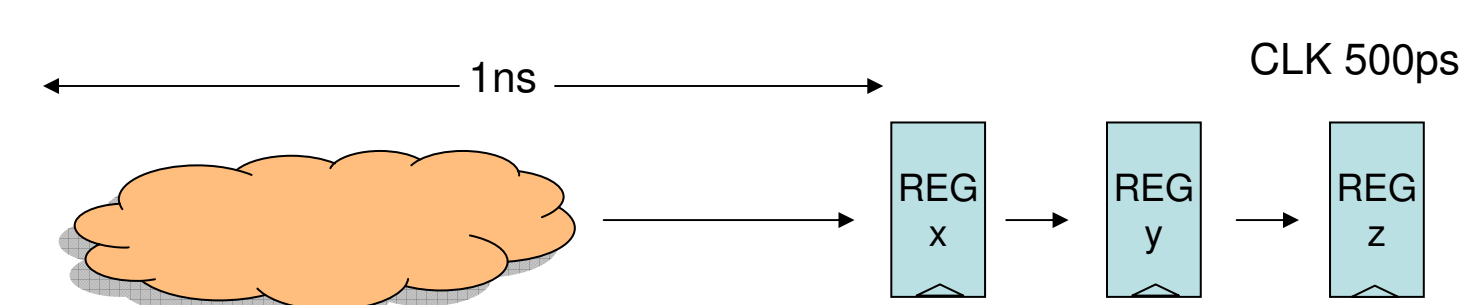
Sequential State definition is a complex problem

Two common methods

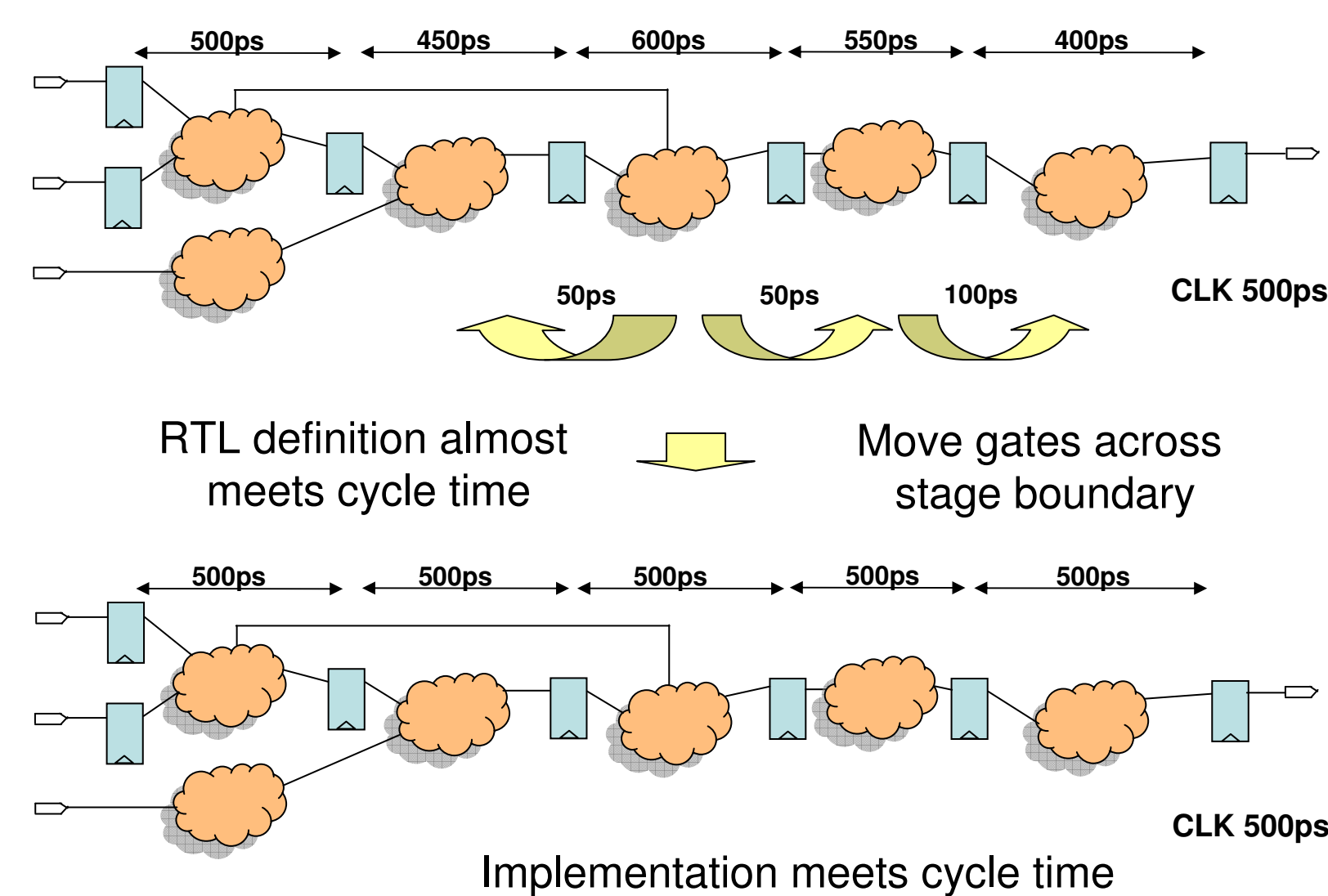
– State Matching



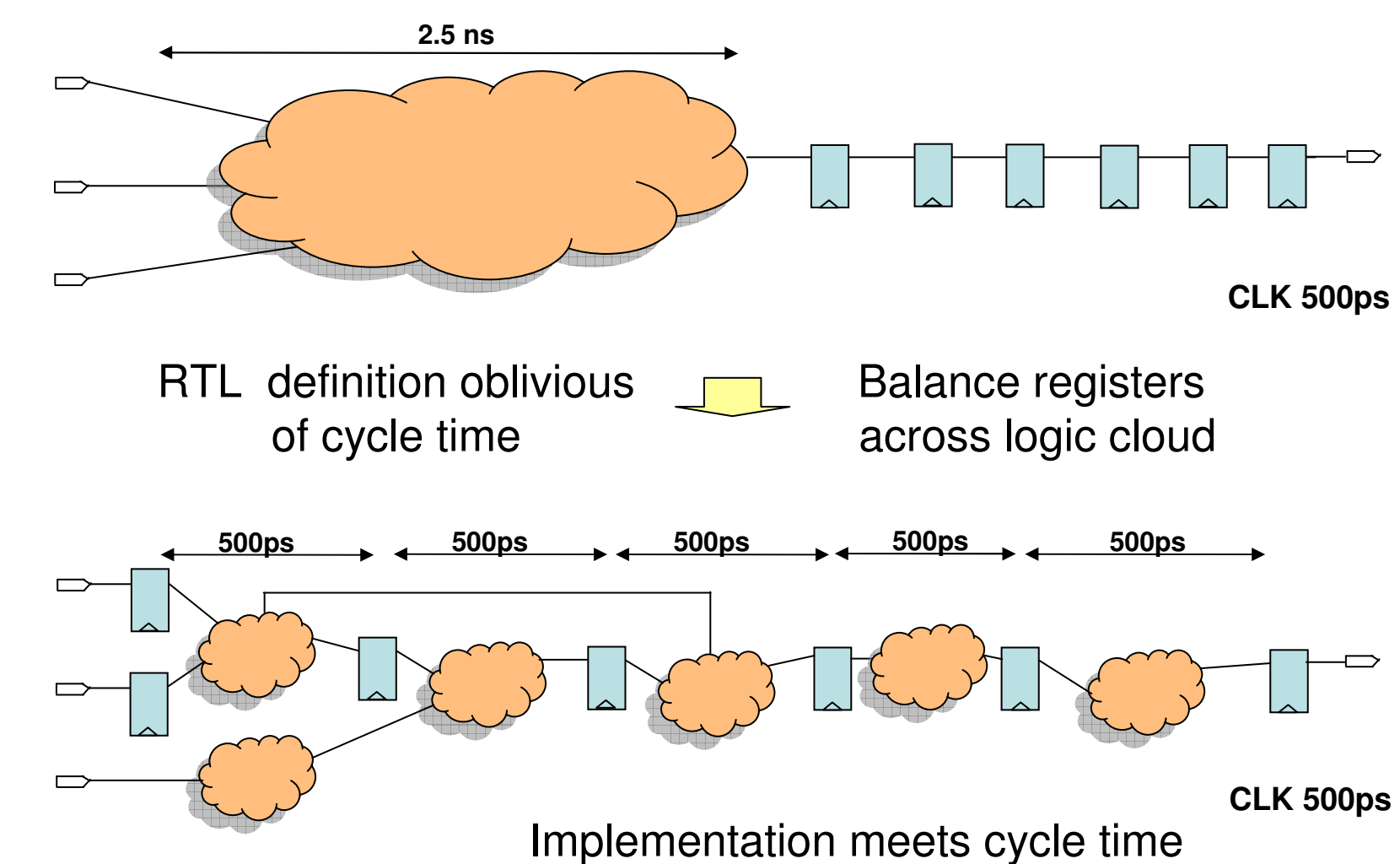
– Register Retiming



## Register Retiming – Case I



## Register Retiming – Case II

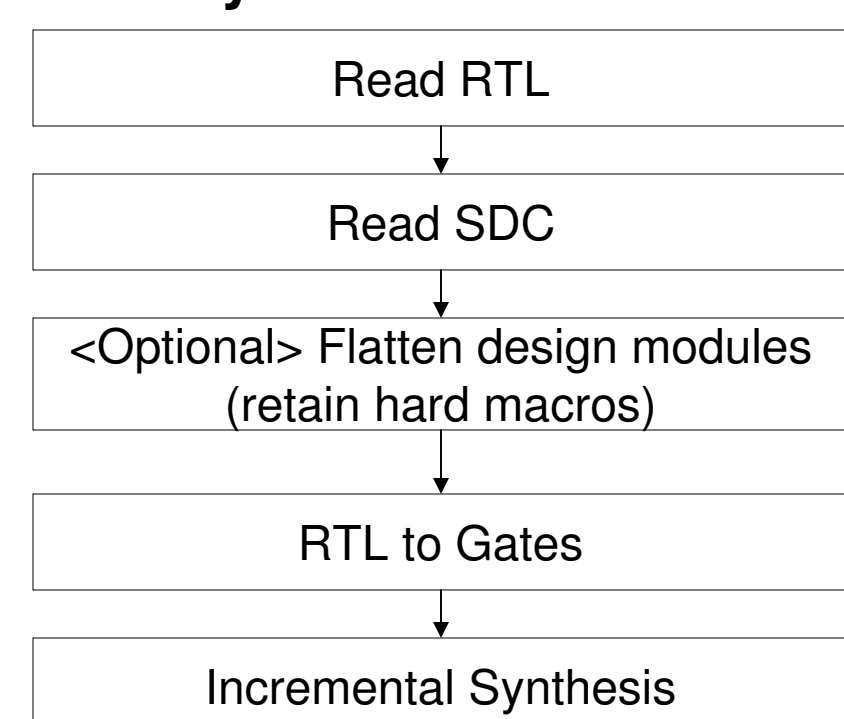


## Logic Synthesis Flow

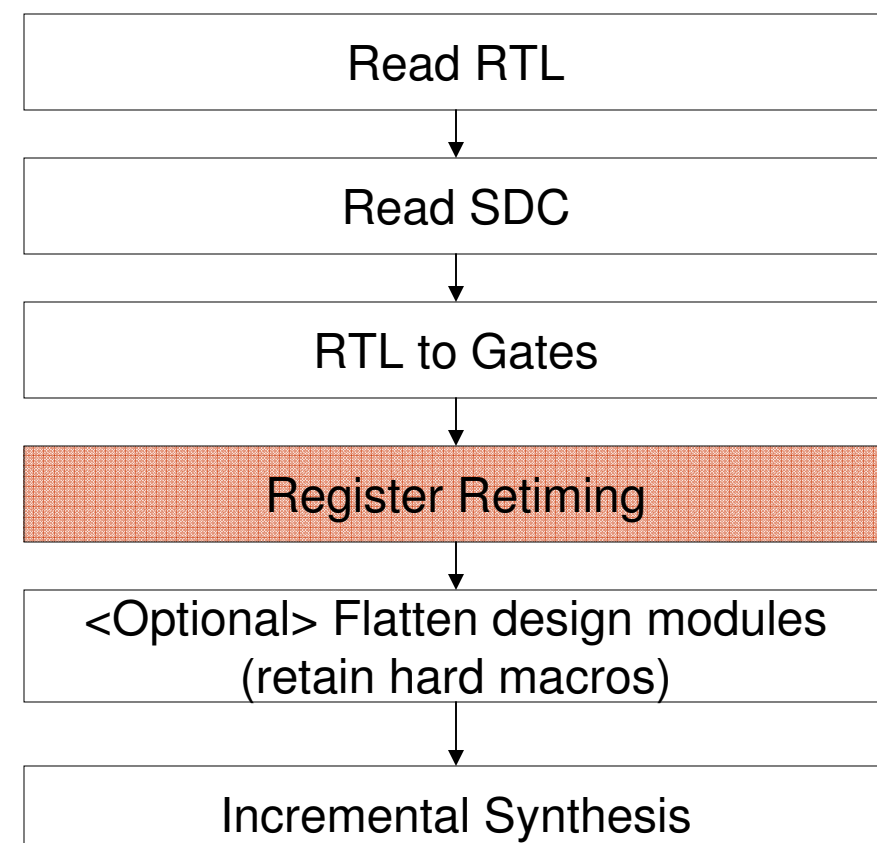
Retiming is integrated into logic synthesis flow

– Flattening of design happens after retiming

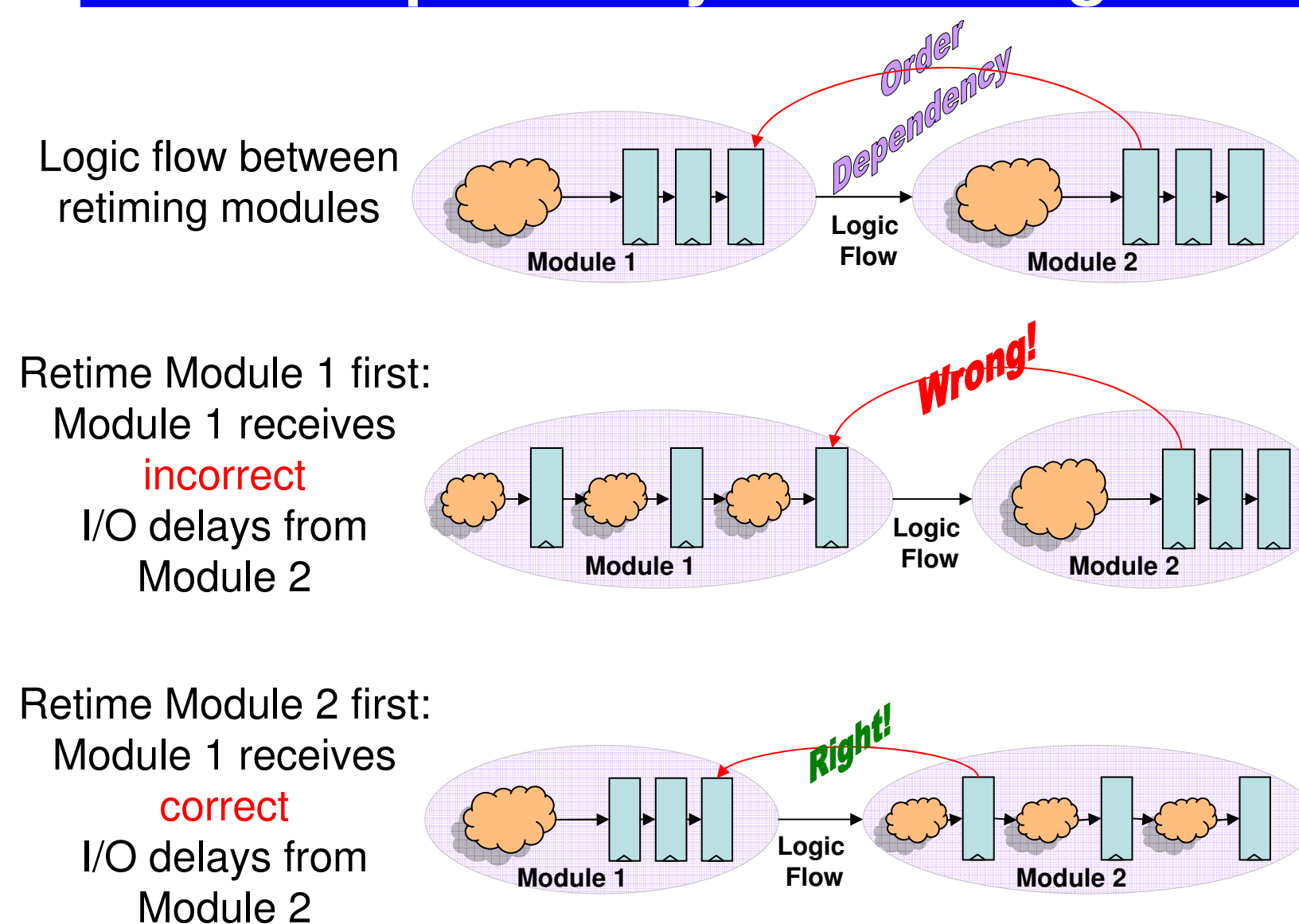
### Synthesis Flow



### Synthesis Retiming Flow



## Order Dependency in Retiming Flow

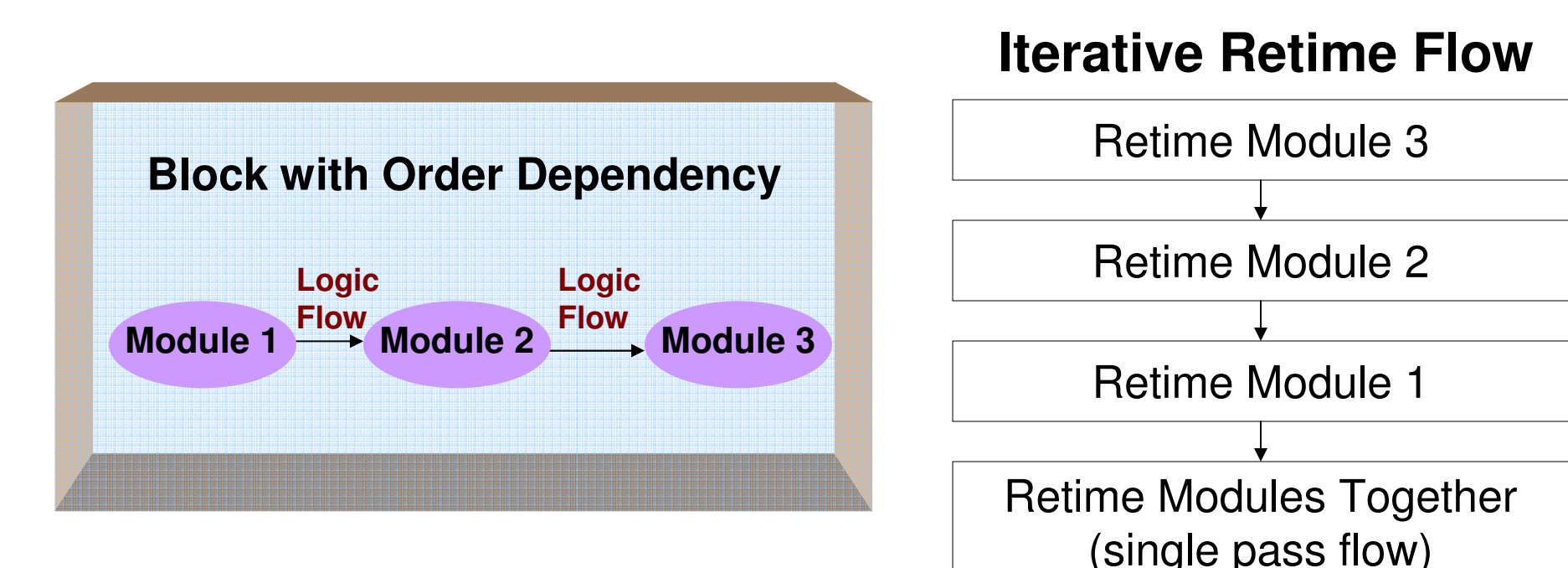


## Single Pass vs. Iterative Retiming

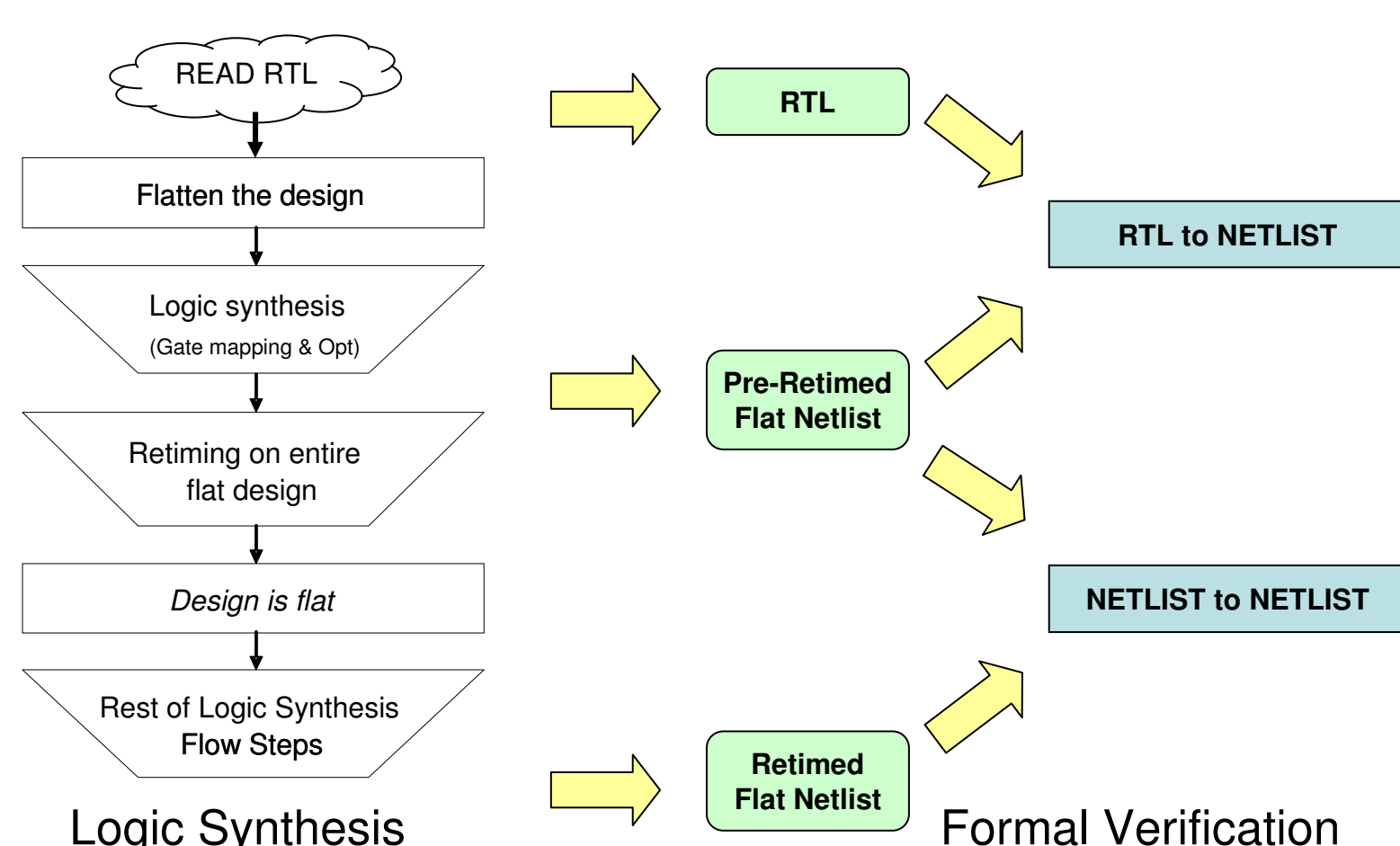
Logic with no direct connectivity in retimed sub-hierarchies can use single pass retiming flow

– No order dependency

Order dependency in logic requires iterative retiming flow

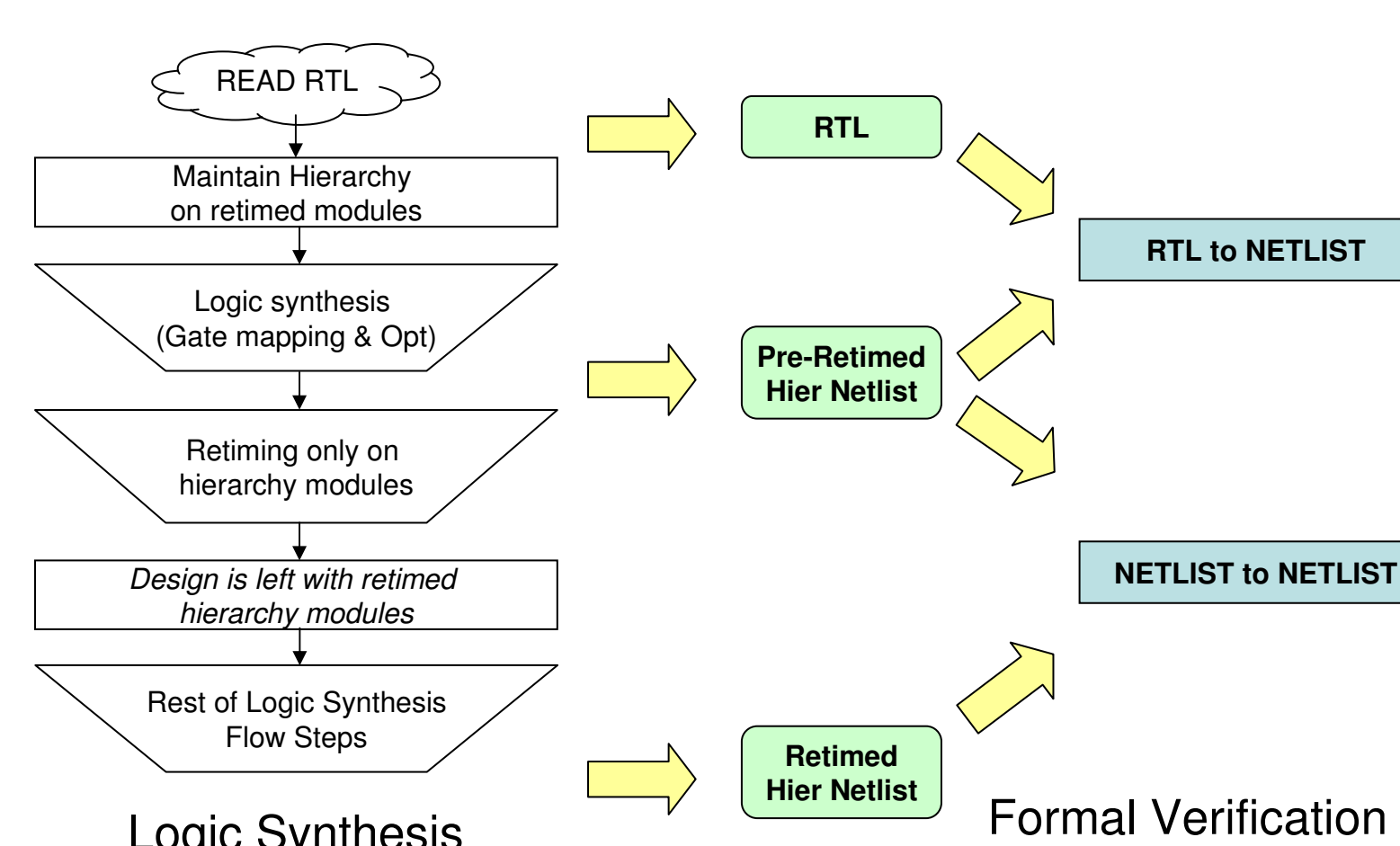


## Flat Design Flow



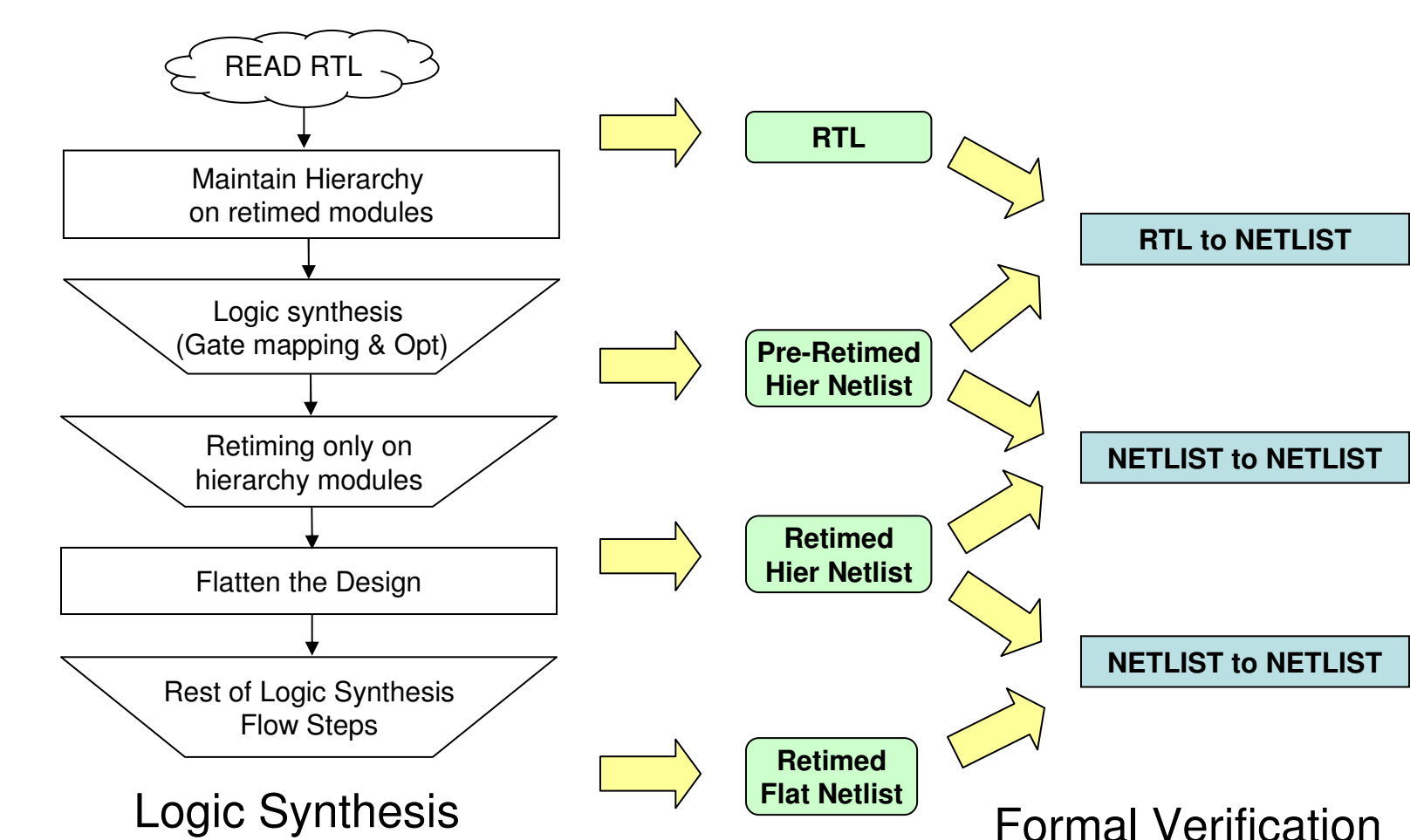
Retiming is done on entire flat design  
Register names and boundaries change in entire design  
+ Standard simple flow  
- Not optimal for area, power, timing  
- Formal verification is difficult with lot of mismatches  
~ Needs 2-stage FV flow

## Hierarchical Design Flow



Modules which need retiming are left as hierarchies  
Register names change only within retimed modules  
+ Good for area, timing and power  
+ Formal verification is clean  
- Optimization & Verification flows are more complex  
~ Needs 2-stage FV flow

## Hybrid Design Flow



Modules which need retiming are left as hierarchies  
Design is flattened after retiming  
Register names change only within retimed modules  
Better area, power and timing optimization  
Standard physical verification flows  
Needs 3-stage FV stage

## Results – Relative Performance

Block#	Flow	Ratio of Logic Area	Ratio of Int WNS	Ratio of Int #paths	Ratio of Total Z	Ratio of Max Lkg	Comments
Block1	Flat	1.18			1.14	1.17	Hierarchical & Hybrid Flows show equal results Hybrid Flow chosen due to simplicity of Verification flows
	Hierarchy	1.00	1.00	1.00	1.00	1.00	
	Hybrid	1.01			1.04	1.04	
Block2	Flat	1.45	0.27	2.46	1.66	1.76	Hybrid Flows show significantly better results
	Hierarchy	1.00	1.00	1.00	1.00	1.00	
	Hybrid	0.92	0.20	0.19	0.88	0.89	
Block3	Flat	1.05	2.63	1.39	1.07	0.84	Hybrid Flows show significantly better results
	Hierarchy	1.00	1.00	1.00	1.00	1.00	
	Hybrid	1.02	0.79	0.42	0.97	0.75	
Block4	Flat	0.98			1.00	1.00	Hierarchical & Hybrid Flows show equal results Hybrid Flow chosen due to simplicity of Verification flows
	Hierarchy	1.00	1.00	1.00	1.00	1.00	
	Hybrid	1.00			1.00	1.01	
Block6	Flat	1.02			1.04	1.05	Hierarchical & Hybrid Flows show equal results Hybrid Flow chosen due to simplicity of Verification flows
	Hierarchy	1.00	1.00	1.00	1.00	1.00	
	Hybrid	1.00			1.00	1.00	

## Retiming – Design Flow Comparison

	Flow Complexity	Area Optimization	Power Optimization	Physical Design Verification	Timing Optimization	FV Complexity
Flat	Best	Worst	Worst	Best	Worst	Worst
Hierarchical	Worst	Best	Best	Worst	Best	Best
Hybrid	Worst	Best	Best	Best	Best	Best

Best Average Worst

## Conclusions

- Retiming is used as a technique to decouple RTL and PD for sequential state location
- Presented an efficient hybrid retiming flow
  - Preserves logic hierarchy for retiming
  - Flattening after retiming for best QOR
  - Supports easy to use FV methodology

