Exploiting Level Sensitive Latches in Wire Pipelining

Jiang Hu
Department of Electrical Engineering
Texas A&M University

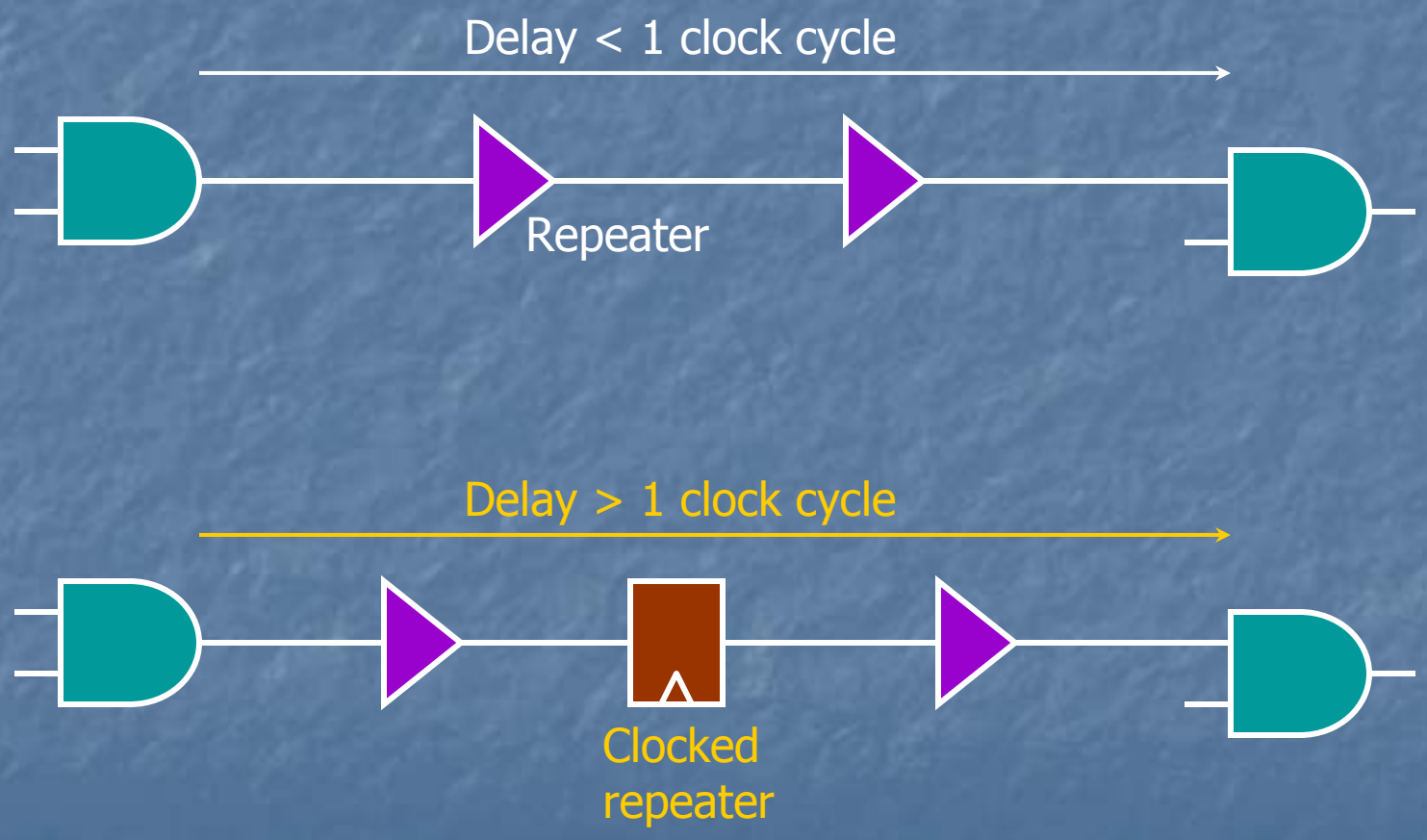
Outline

- Technology trend and wire pipelining
- Previous works
- Advantages and challenges of using latches
- Concurrent repeater, flip-flop and latch insertion
- Experimental results
- Conclusion

Technology Scaling

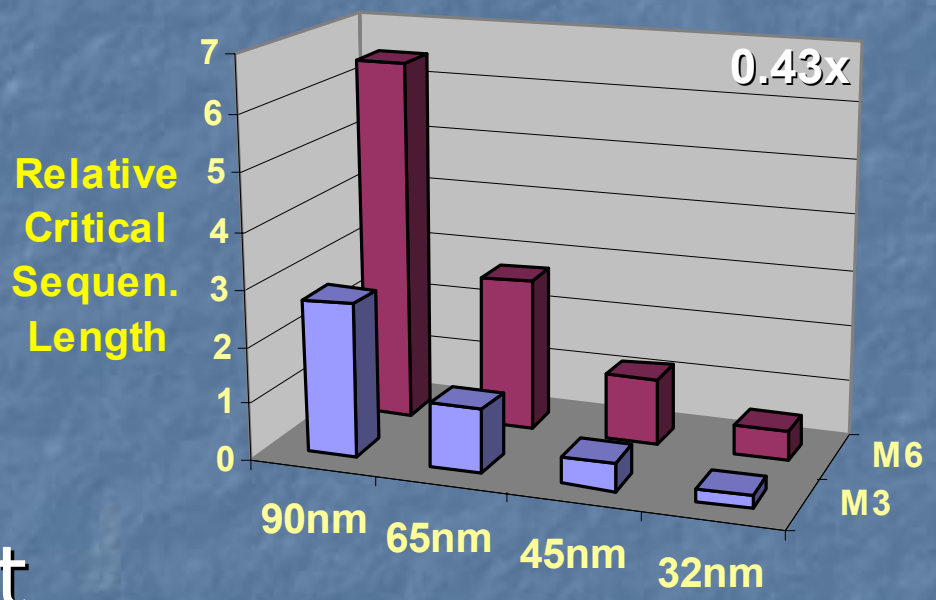
- Wire delay increase
- Clock period decrease
- Chip size increase
- Multiple clock cycles for long distance signal propagation

Wire Pipelining



Future Trend

- Critical sequential length: distance between adjacent clocked repeaters
- Intel: 20% of total chip cells will be clocked repeaters at *32nm* technology



P. Saxena, et al, ISPD 03

Design Implications

- Minimizing delay is inadequate
  - Latency: number of clocked repeaters along a source-sink path
  - Latency needs to be minimized or latency constraint needs to be satisfied
- More power dissipations
  - Clocked repeaters are generally larger than unclocked repeaters
  - Extra load to clock network
- Greater vulnerability to variations
  - Dependence on clock skew

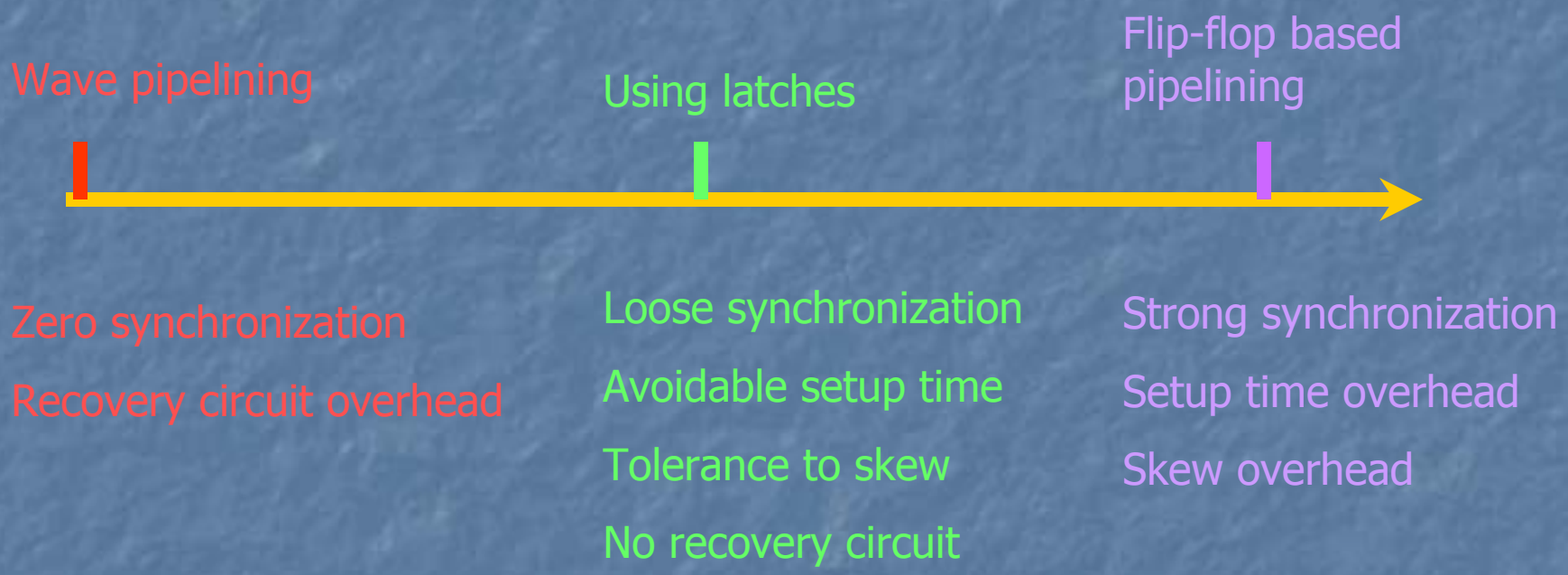
Previous Work I: Concurrent Repeater and Flip-flop Insertion

- *Cocchini*, *Hassoun-Alpert-Thiagarajan*, ICCAD 02
- Given
  - Steiner tree, candidate repeater locations on the tree
  - Repeater library: repeaters + edge triggered flip-flops
  - Clock skew
- MiLa
  - Find repeater solution to Minimize the max Latency
- GiLa
  - Find min cost repeater solution to satisfy Given Latency constraint at each sink

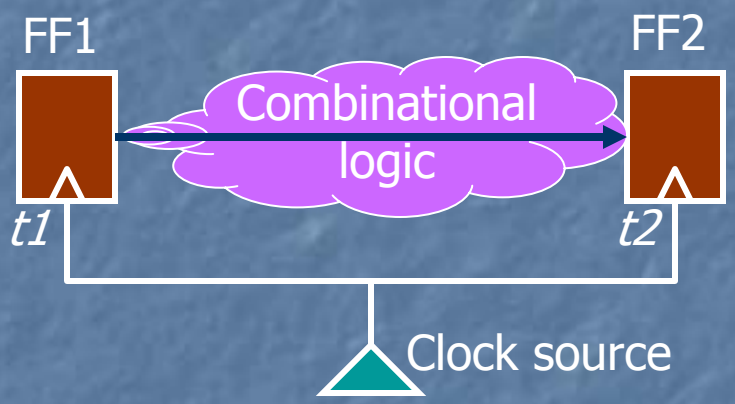
Previous Work II: Wave Pipelining

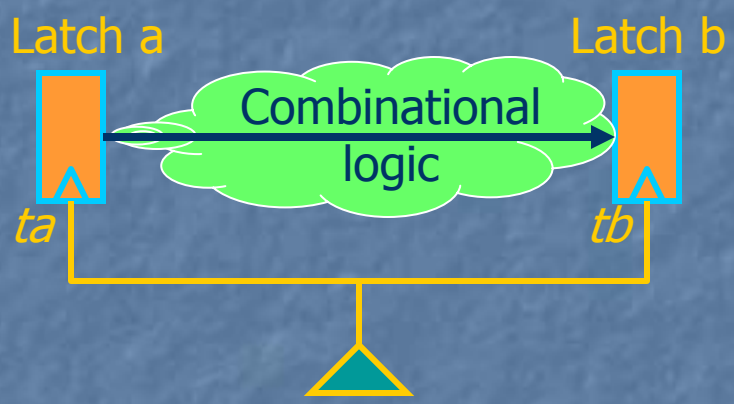
- *L. Zhang*, *Y. Hu* and *C. C.-P. Chen*, TAU 2004
- Wave pipelining
  - Signals are allowed to propagated over multi-cycles without synchronous elements
- Advantages
  - No setup time and skew overhead
- Weakness
  - Complicated recovery circuits at receiver

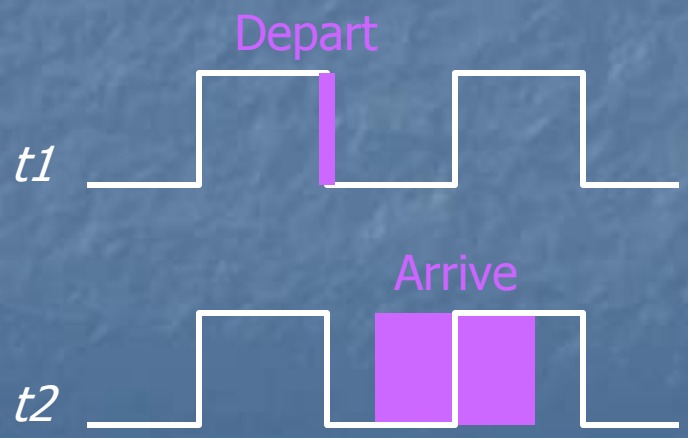
Spectrum of Synchronization Effort

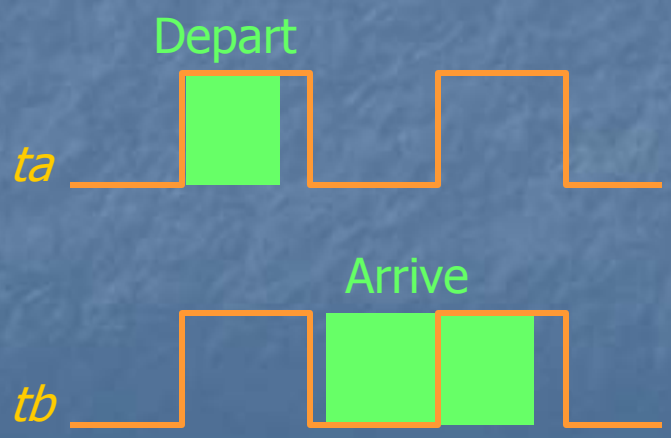


Basics of Flip-flop and Latch

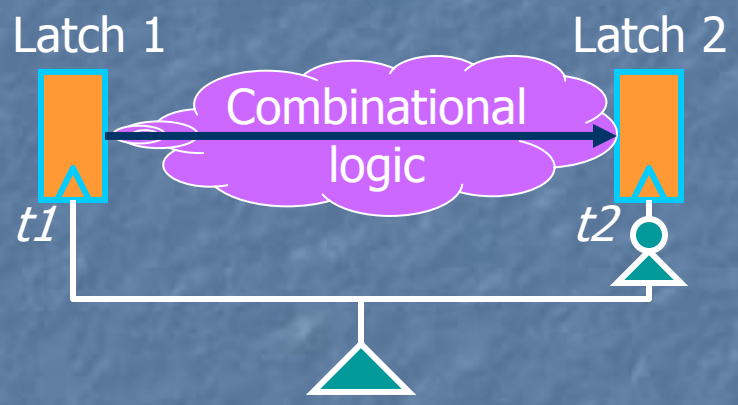


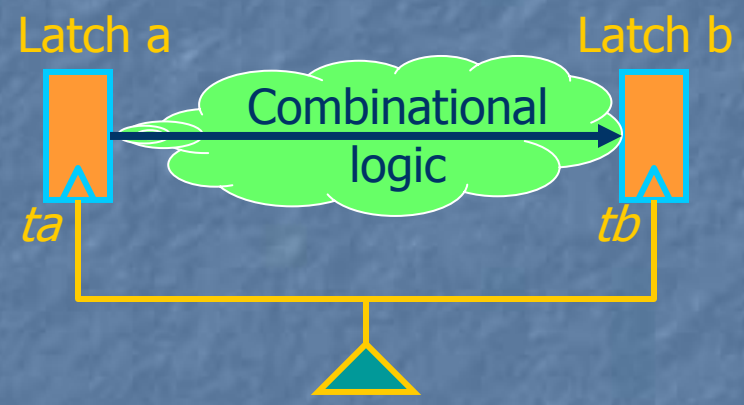






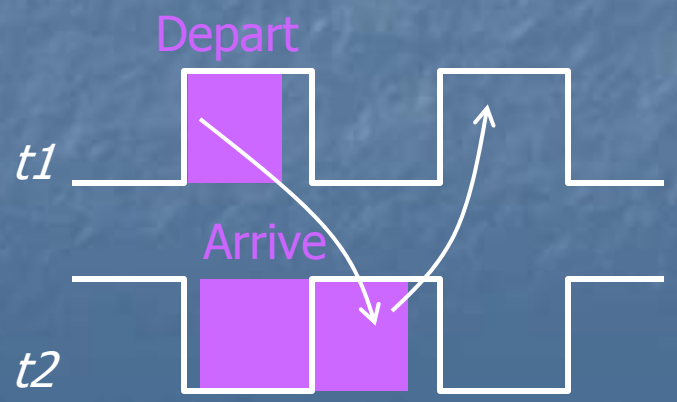
Clocking for Latch Based Designs

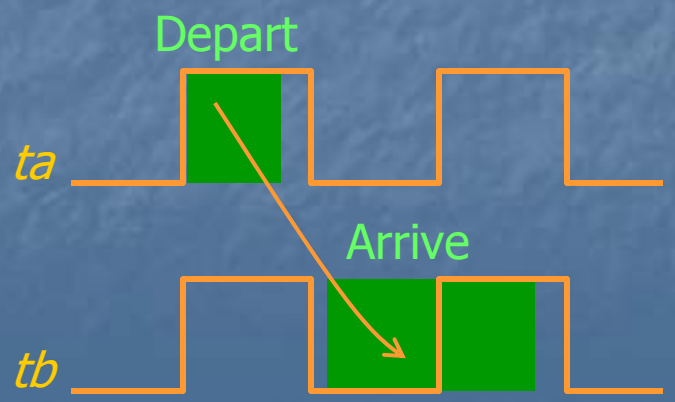




- Traditional 2-opposite-phase clocking
- Each latch drives half clock cycle

- Propose 1-phase clocking
- Each latch drives 1 clock cycle





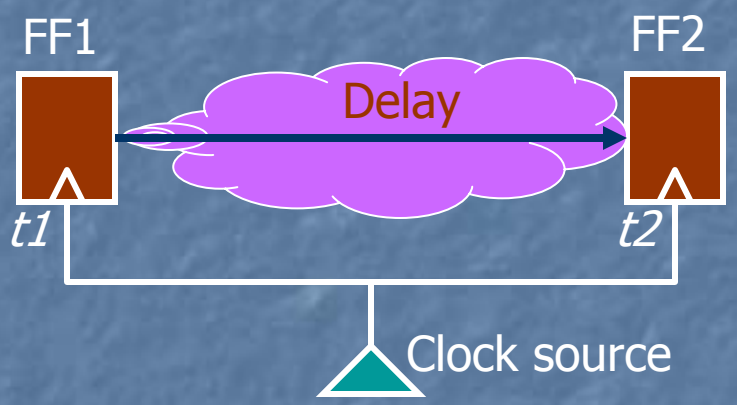
Area and Power Advantages of Using Latches

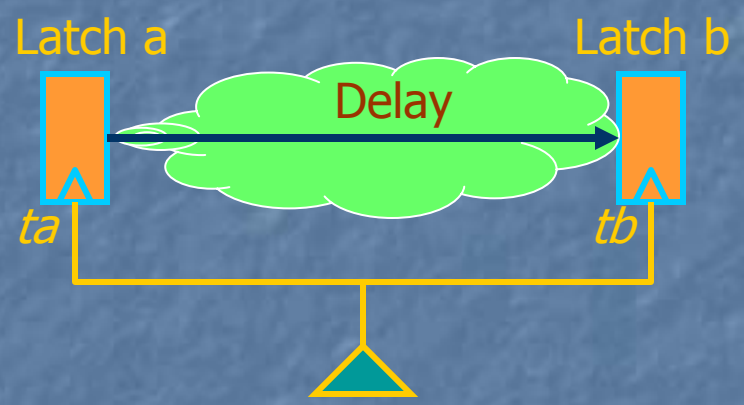
- A flip-flop is usually composed by two latches
- Replacing each flip-flop with one latch may reduce both
  - Area
  - Dynamic power
  - Leakage power
  - Load to clock network

Timing Advantages of Latches

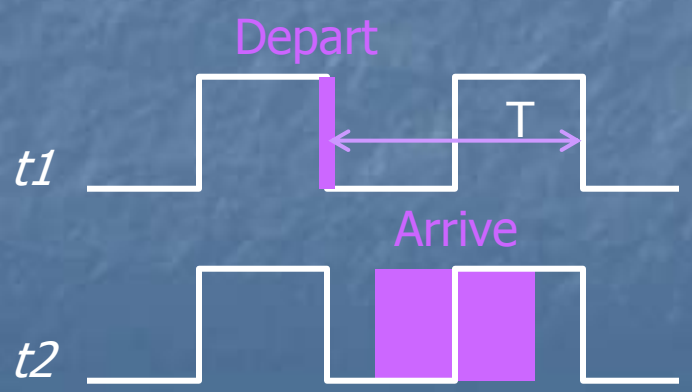
- Each flip-flop is associated with t_{setup}, t_{skew}, and propagation delay t_{prop}
- An optimally buffered long wire, total delay D
- Latency of flip-flop based pipelining
  $Y = D/(T_{cycle} - t_{setup} - t_{skew} - t_{prop})$
- Latency of latch based pipelining
  $Y = D/(T_{cycle} - t_{prop})$

Timing Flexibility of Latches

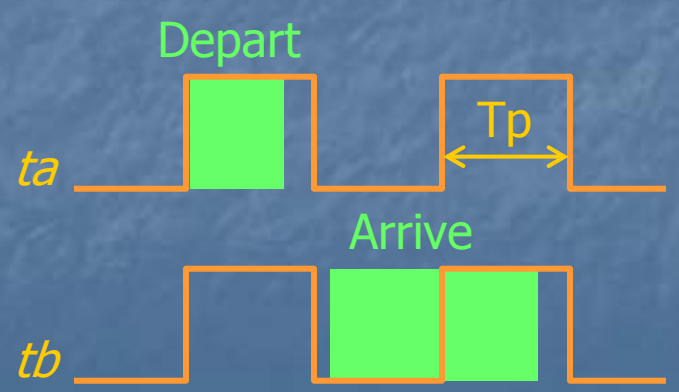




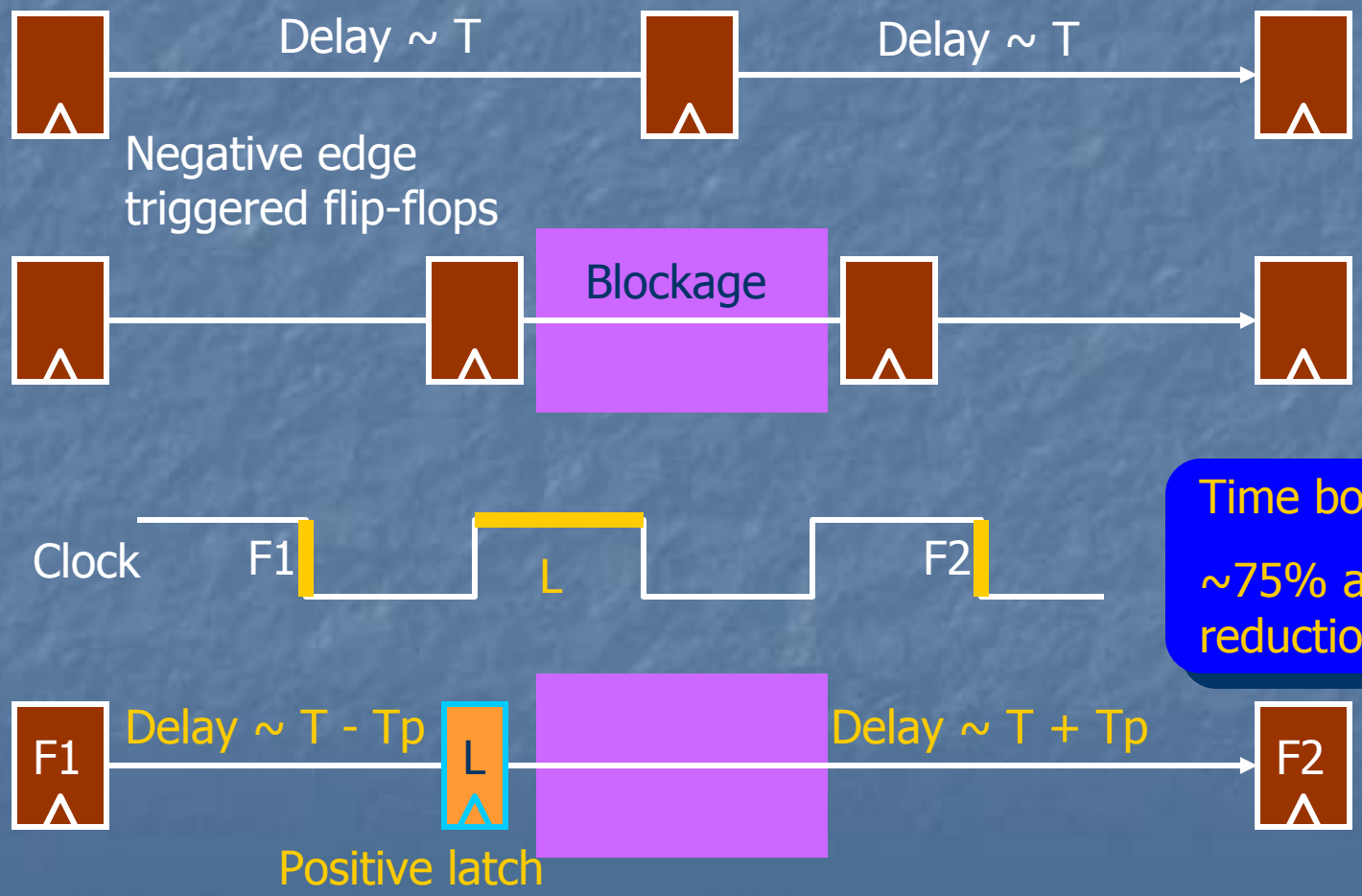
Delay < T



Delay < T + Tp



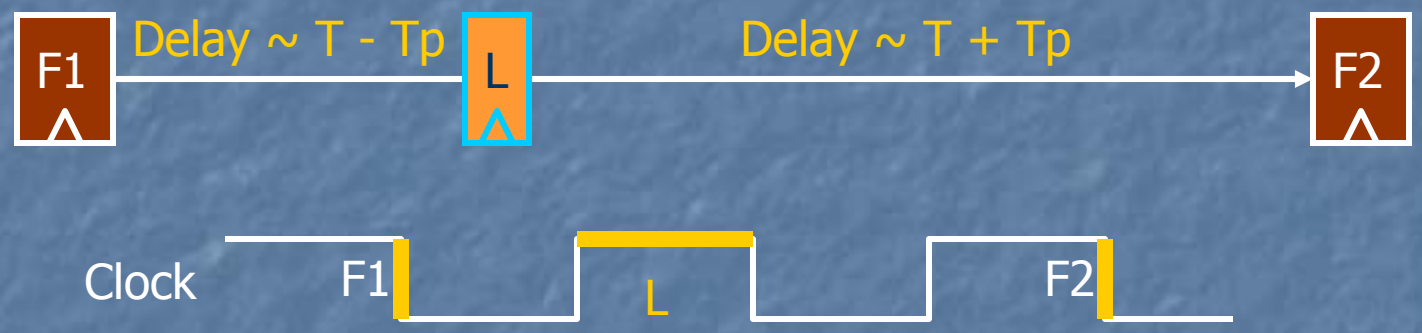
Advantage of Latches on Handling Blockages



Time borrowing

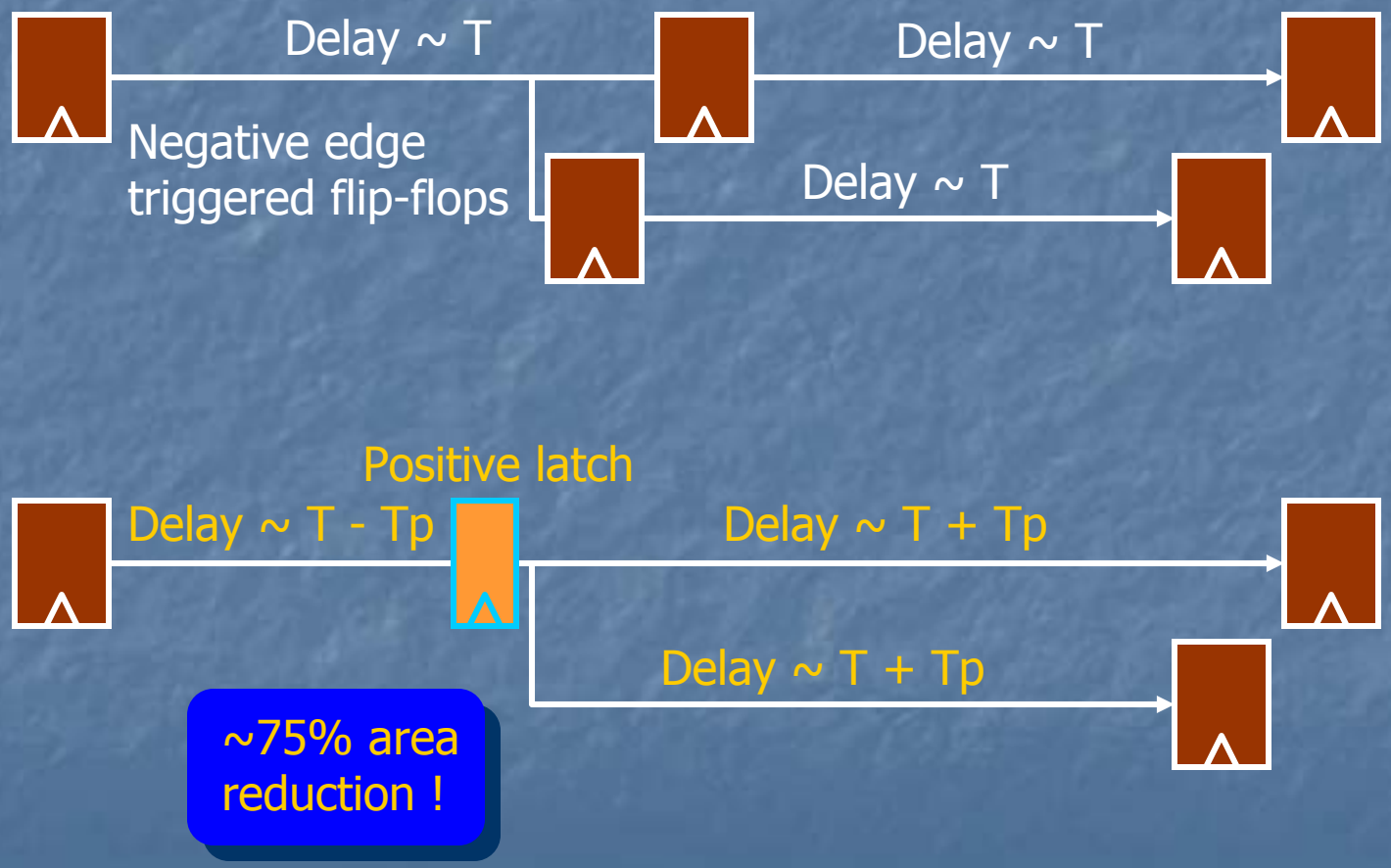
~75% area reduction !

Marginal Wave Pipelining

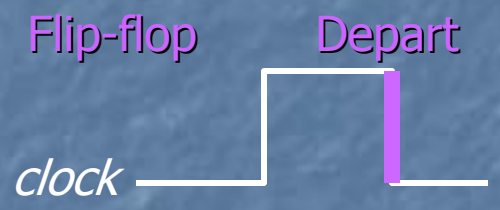


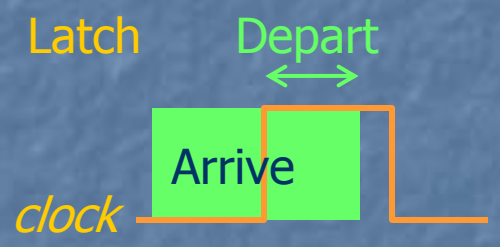
- Sometimes there are two signals between L and F2
- Insert repeaters between L and F2
  => one signal may not overwrite the other
- Turn off during inactive clock level => avoid signal loss

Advantages of Latches in Tree



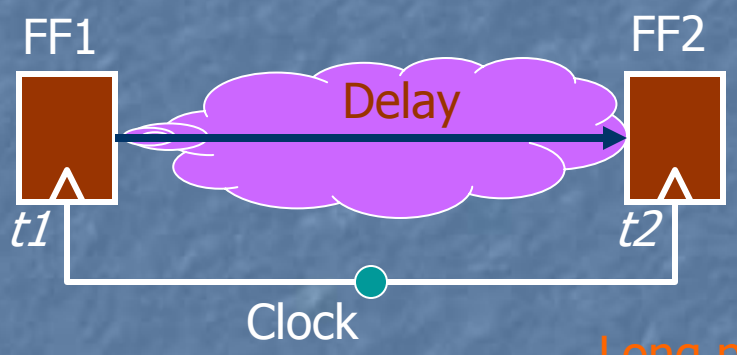
Challenge: Input-Output Timing Coupling

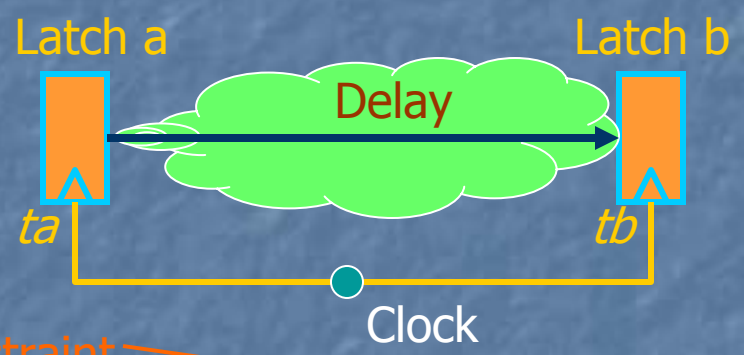




- Output departure time of flip-flop
  - Aligned to clock edge
  - Independent of input arrival time
- Output departure time of latch
  - May depend on input arrival time

Challenge: Tight Short Path Constraint





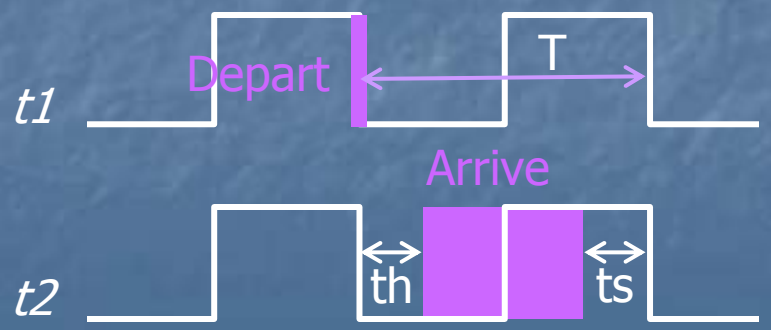
Long path constraint

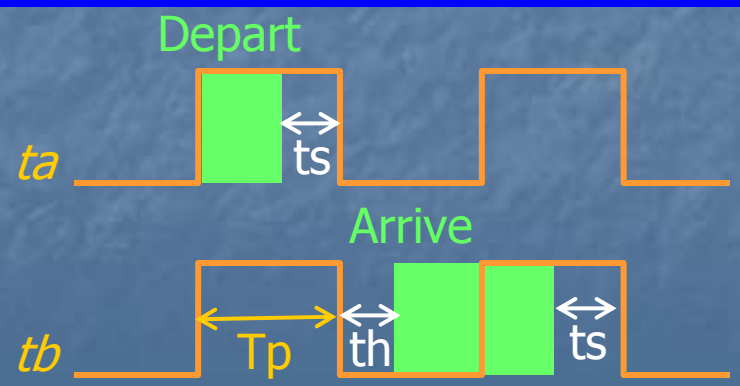
$th < Delay < T - ts$

$Tp + th < t_{depart} + Delay < T + Tp - ts$

$ts + th < Delay_{min} < Tp + th$

Short path constraint





There's Something About Latch

- Designers don't like latch despite its advantages
- Complicated for circuit designs
  - Circuit topology == graph, sometimes with loops
- Wire pipelining topology == tree
- Use latches together with flip-flops
  - Timing flexibility of latches + short path immunity of flip-flops

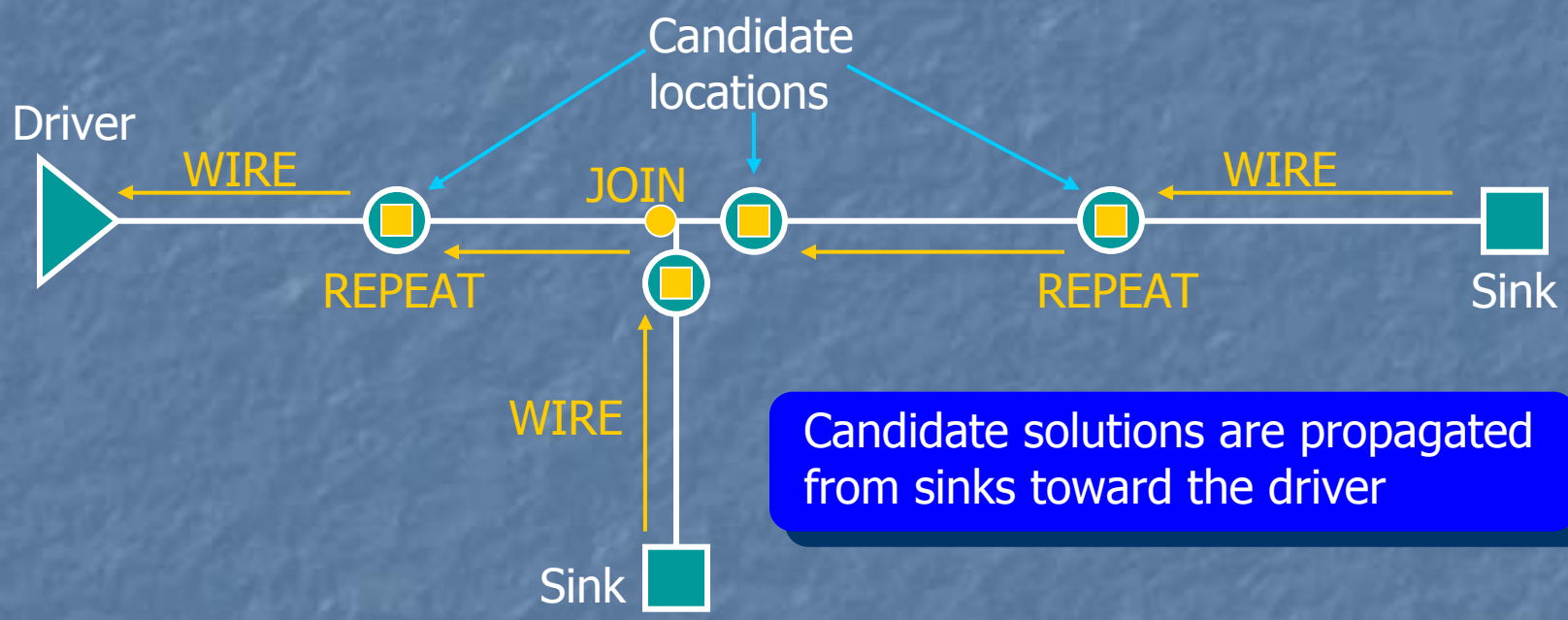
Assumptions

- Flip-flop + latch based wire pipelining is applied in a flip-flop based circuit design
- Single phase clock for flip-flops and latches
- Flip-flops are negative edge triggered
- Latches are positive level sensitive
- Timing reference point is aligned with fall egde of the clock signal
- Clock skew and repeater intrinsic delay are neglected for simplicity of expression

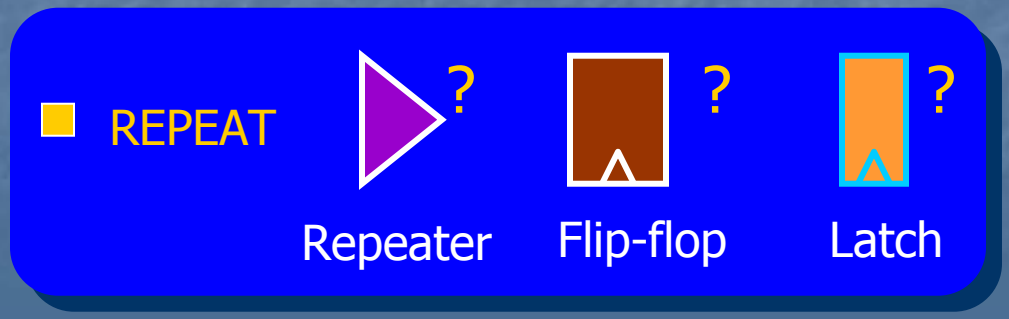
Problem Formulation

- Given
  - Steiner tree, candidate repeater locations on the tree
  - Repeater library: repeaters + flip-flops + latches
- MiLa: find repeater solution to Minimize the max Latency
- GiLa: find min cost repeater solution to satisfy Given Latency constraint at each sink
- Both long path and short path constraints are satisfied for flip-flops and latches
- Gate RC switch model, wire Elmore delay model

Algorithm Overview



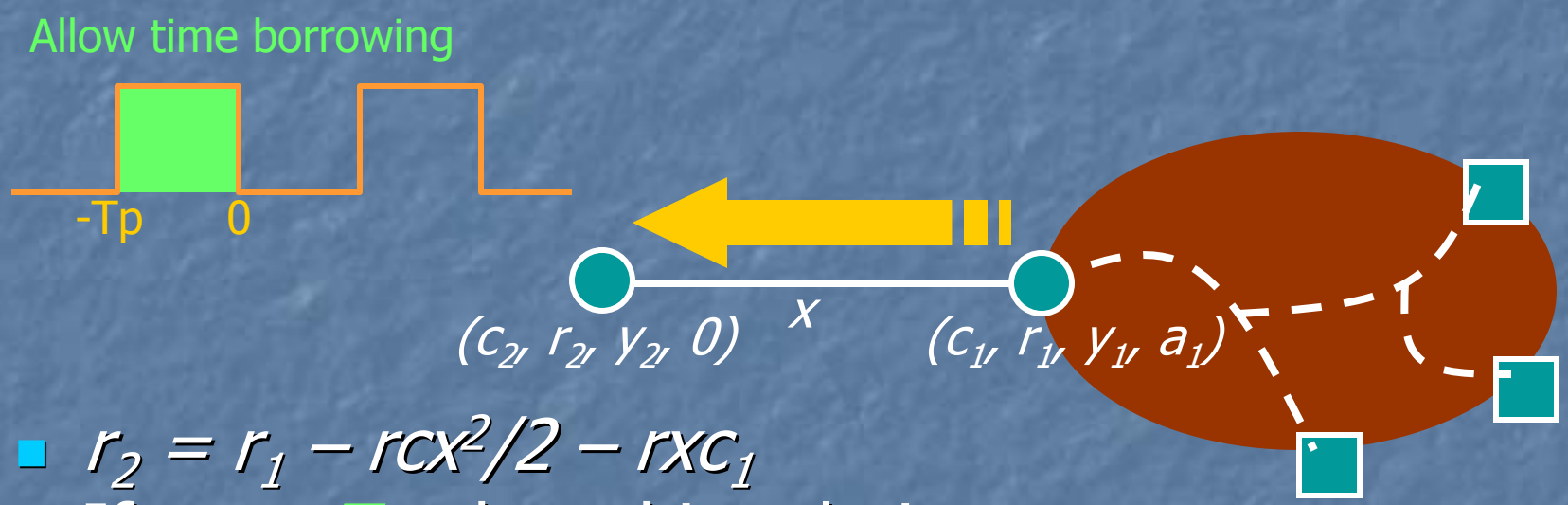
Candidate solutions are propagated from sinks toward the driver



Candidate Solutions

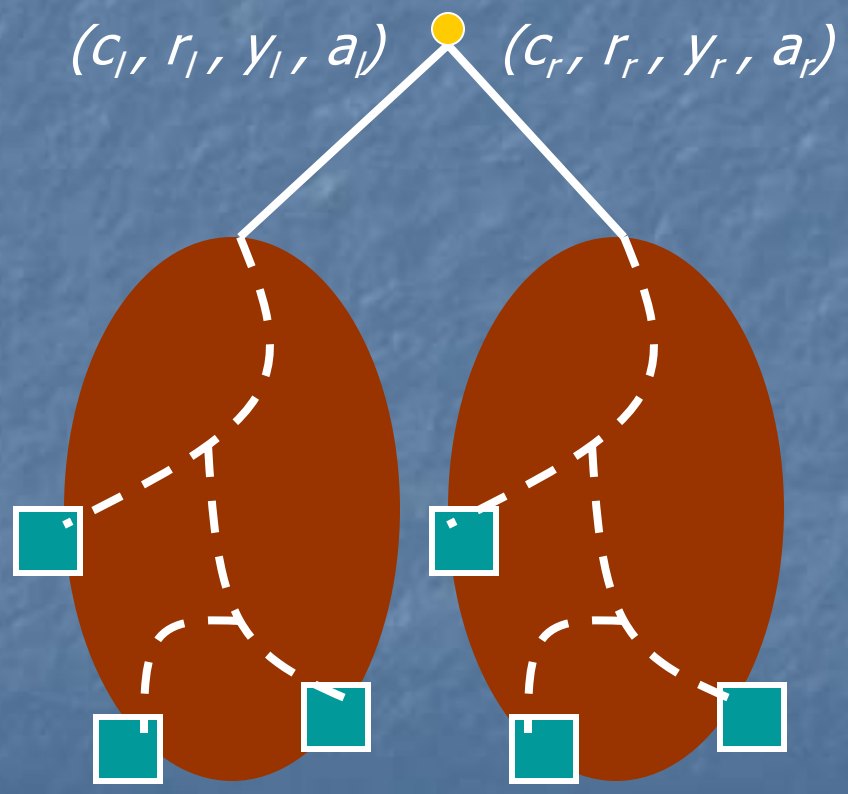
- Each candidate solution is associated with a node in tree, and is characterized by
  - c: downstream cap seen from the node
  - r: required arrival time
  - y: latency
  - a: repeater assignment
- At each sink node j, initial candidate solution $(c_j, r_j, 0, 0)$

Solution Propagation: WIRE



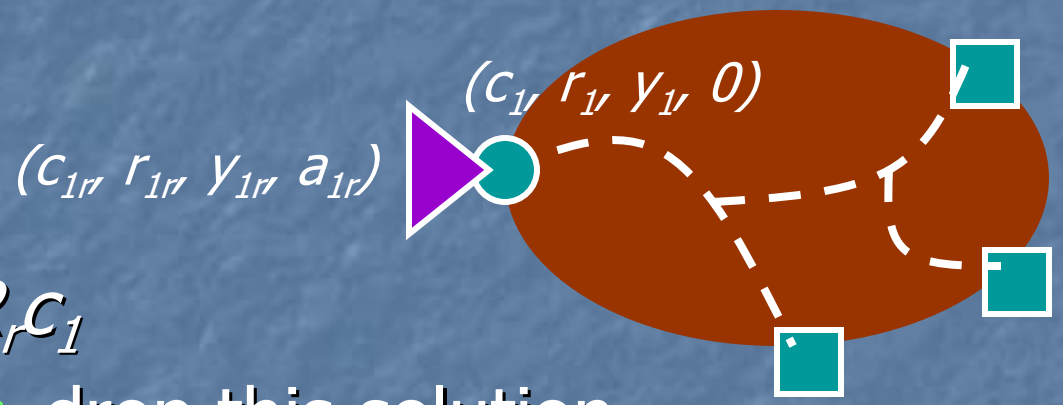
- $r_2 = r_1 - rcx^2/2 - rxc_1$
- If $r_2 < -Tp$, drop this solution
- $c_2 = c_1 + cx$
- $y_2 = y_1$
- *r:* wire resistance per unit length
- *c:* wire capacitance per unit length

Solution Propagation: JOIN



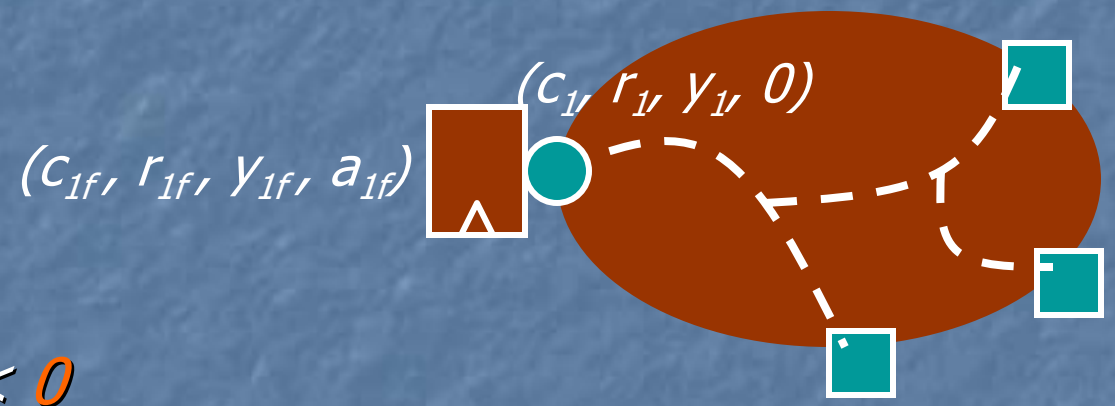
- $c_{join} = c_l + c_r$
- $r_{join} = min(r_l, r_r)$
- $y_{join} = max(y_l, y_r)$
- $a_{join} = a_l \cup a_r$

REPEAT: Insert Repeater



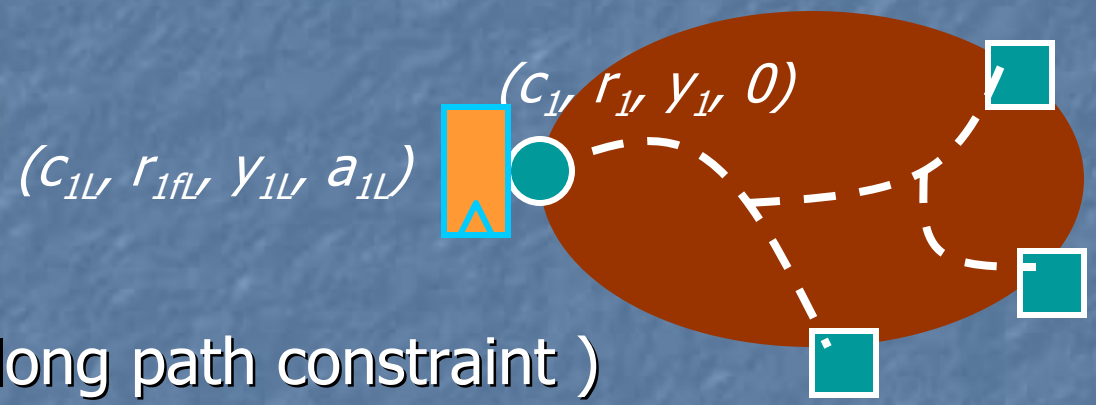
- $r_{1r} = r_1 - R_r c_1$
- If $r_{1r} < -Tp$, drop this solution
- $c_{1r} = C_r$
- $y_{1r} = y_1$
- C_r: repeater input capacitance
- R_r: repeater output resistance

REPEAT: Insert Flip-flop



- If $r_1 - R_f c_1 < 0$
  - Skip the REPEAT to enforce long path constraint
- $c_{1f} = C_f$
- $r_{1f} = T - t_{setup}$
- $y_{1f} = y_1 + 1$
- C_f: flip-flop input capacitance
- R_f: flip-flop output resistance

REPEAT: Insert Latch

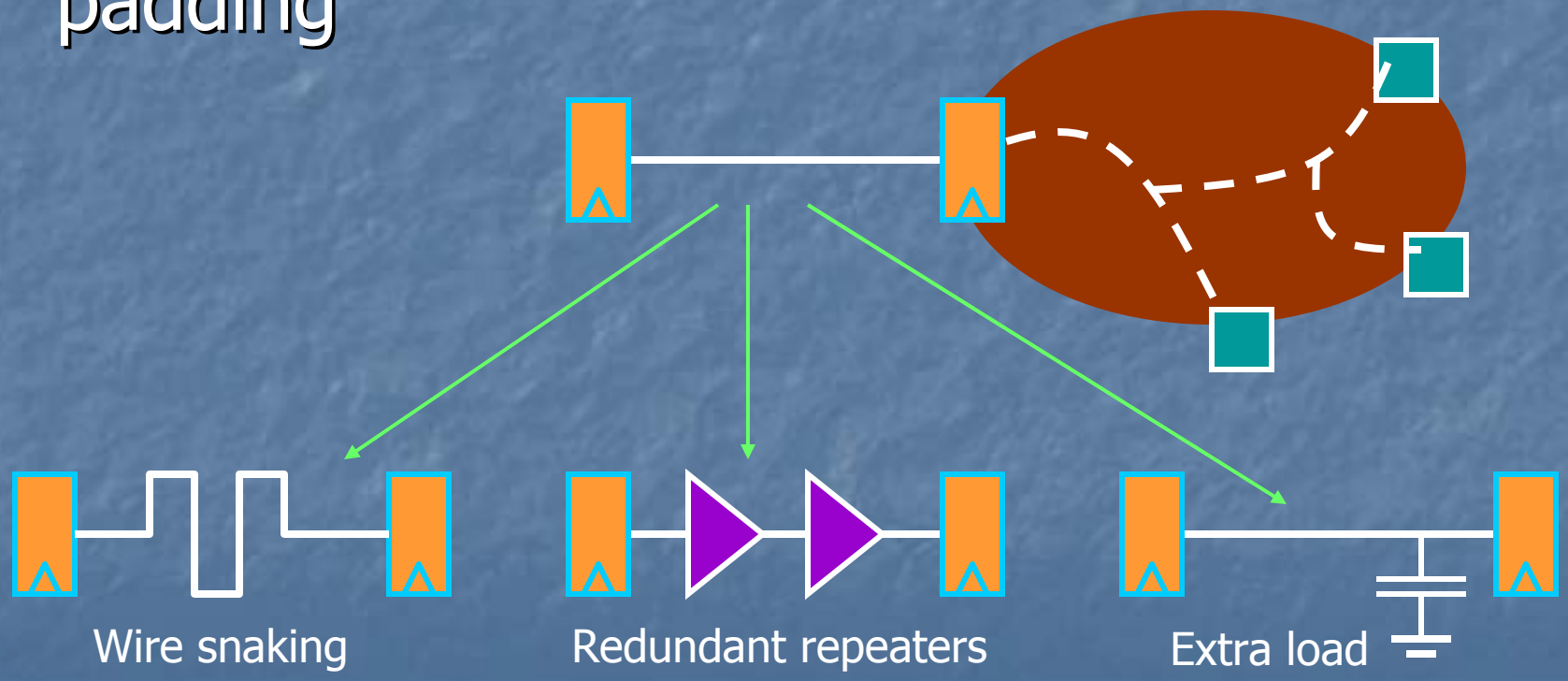


- $r' = r_1 - R_L c_1$
- If $r' < -Tp$, quit (long path constraint)
- $r' <=$ delay padding (short path constraint)
- $c_{1L} = C_L$
- $r_{1L} = min(T - t_{setup}, T + r')$
- $y_{1L} = y_1 + 1$
- C_L: latch input capacitance
- R_L: latch output resistance

$r' < -t_{setup}$ implies time borrowing

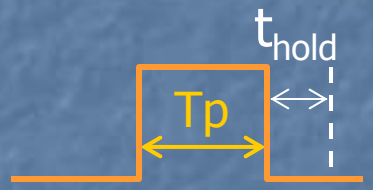
Delay Padding

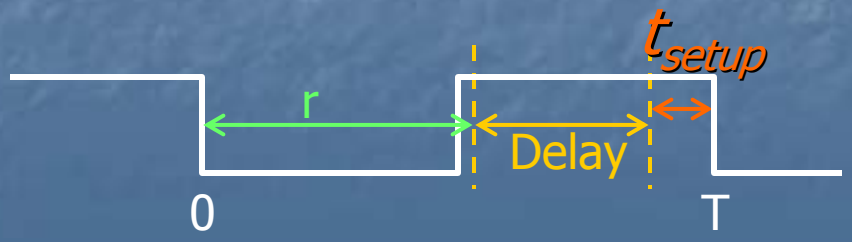
- Short path violation can be fixed by delay padding



Uniform Delay Padding

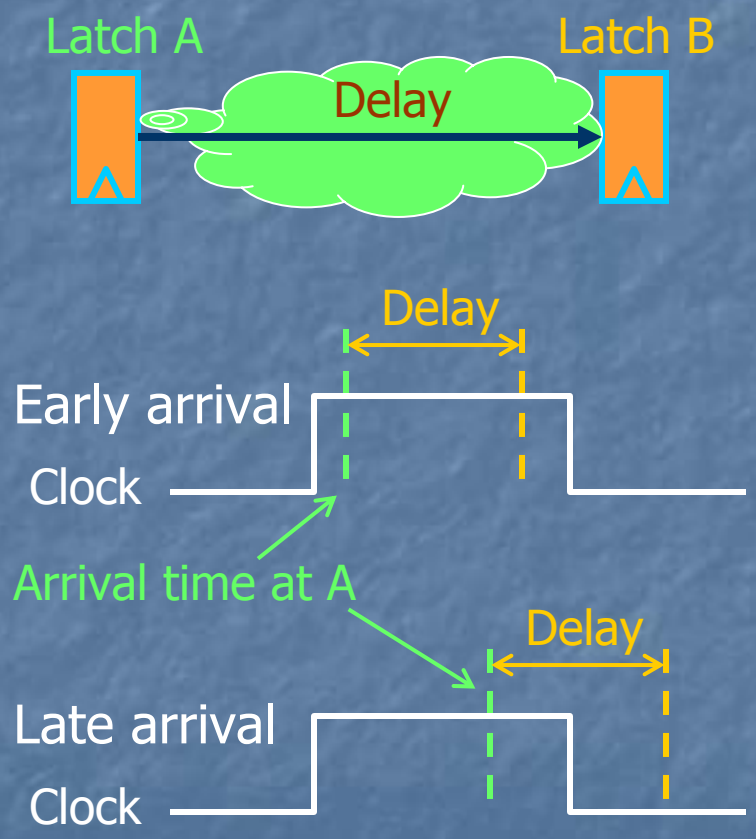
- If $delay > Tp + t_{hold}$, there is no short path violation
- Pad delay if $delay < Tp + t_{hold}$ when insert latch
- If $r > T - t_{setup} - Tp - t_{hold}$ pad delay of $r - T + t_{setup} + Tp + t_{hold}$





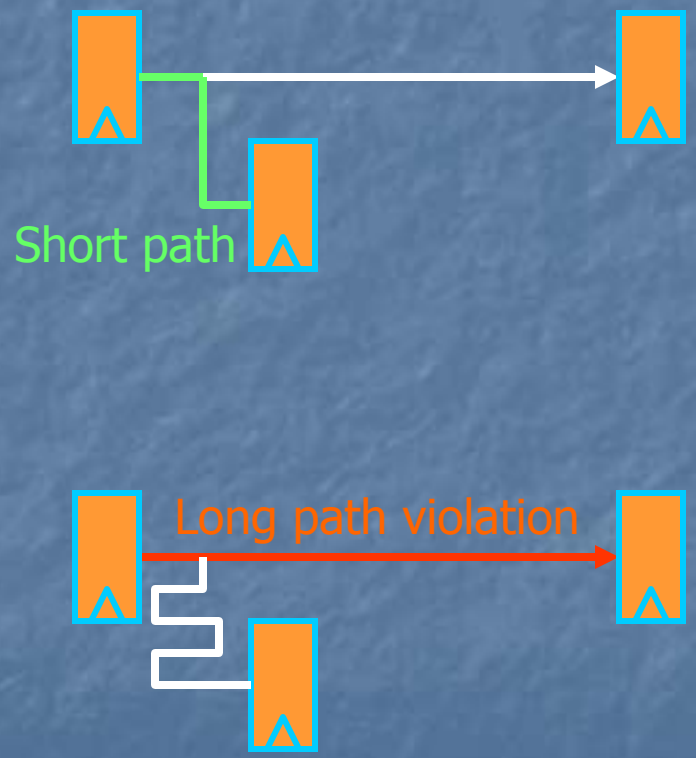
Pessimism of Uniform Delay Padding

- Even if *delay* < *Tp* + t_{hold} , short path constraint may be satisfied when signal arrival time is late
- Uniform delay padding may cause unnecessary padding
- Arrival time is not known in bottom-up solution propagation!



Delay Padding in Post Processing?

- To do padding with known arrival time
  - Propagate solutions ignoring short path constraint
  - Do delay padding in post processing
- Problem: in multi-fanout tree, delay padding in one branch may affect delay of other branches



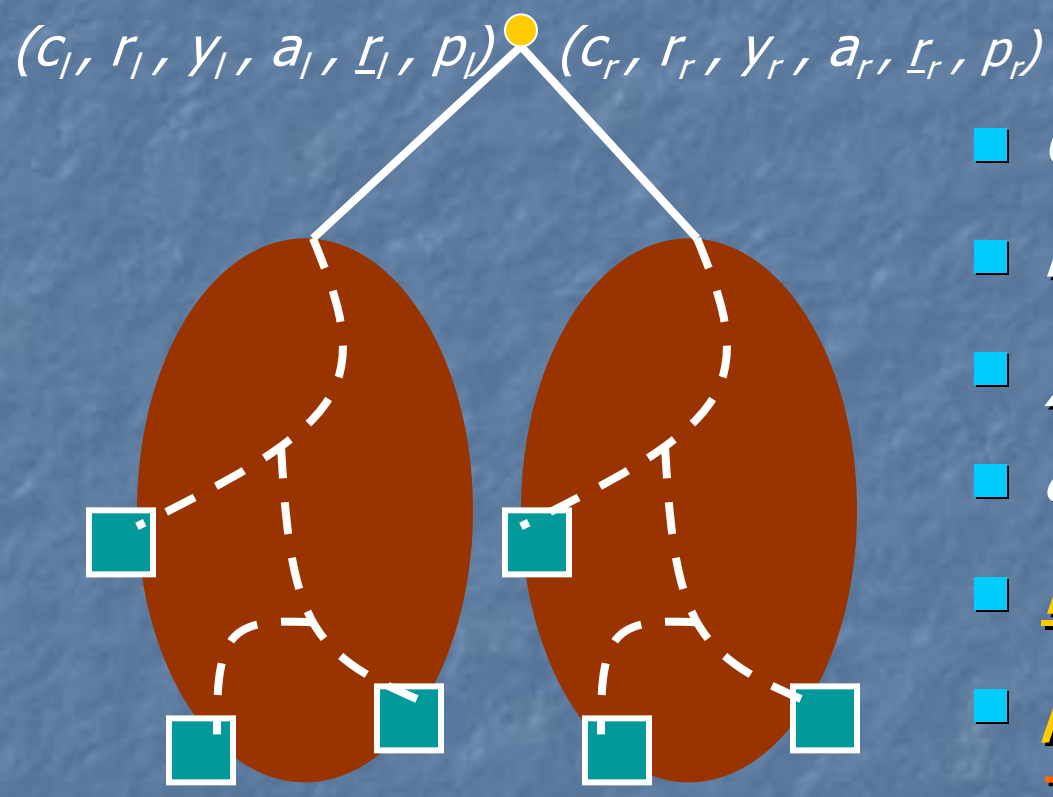
Deferred Delay Padding

- Along with solution propagation
- Note potential delay padding
- Defer actual padding till arrival information is available

Additional Characteristics for Solutions

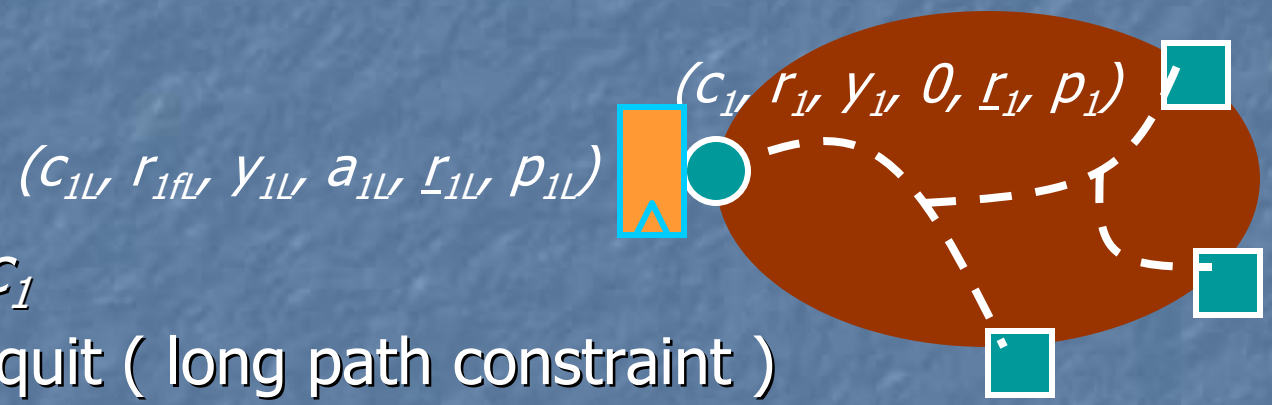
- Earliest required arrival time $\underline{r}$
  - $\underline{r} < RAT < r$
  - Large $\underline{r}$ implies more chance of actual padding
- Potential delay padding p associated with a specific short path
- Each solution is characterized by $(c, r, y, a, \underline{r}, p)$
- At sink j, solution is $(c_j, r_j, 0, 0, r_j - T, 0)$

JOIN Operation for Deferred Delay Padding



- $c_{join} = c_l + c_r$
- $r_{join} = min(r_l, r_r)$
- $y_{join} = max(y_l, y_r)$
- $a_{join} = a_l \cup a_r$
- $\underline{r}_{join} = max(\underline{r}_l, \underline{r}_r)$
- $p_{join} = p_l \cup p_r$
- If $\underline{r}_{join} > r_{join}$, pad delay

Latch Insertion Revisited

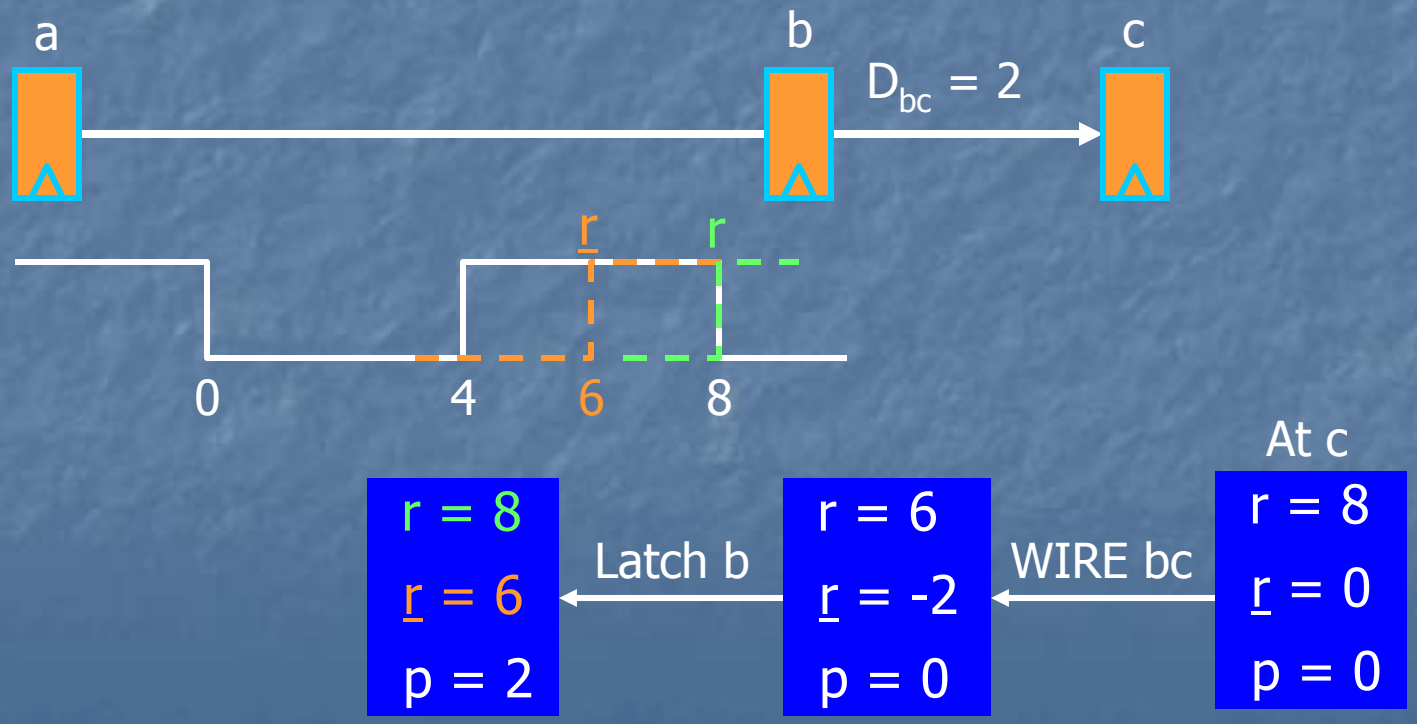


- $r' = r_1 - R_L c_1$
- If $r' < -Tp$, quit (long path constraint)
- $r' <=$ deferred delay padding, decide
  - Potential delay padding p
  - Any part of p needs to be instantiated
  - Update $\underline{r}$
- $c_{1L} = C_L$
- $r_{1L} = min(T - t_{setup}, T + r')$
- $y_{1L} = y_1 + 1$

$r' < -t_{setup}$ implies time borrowing

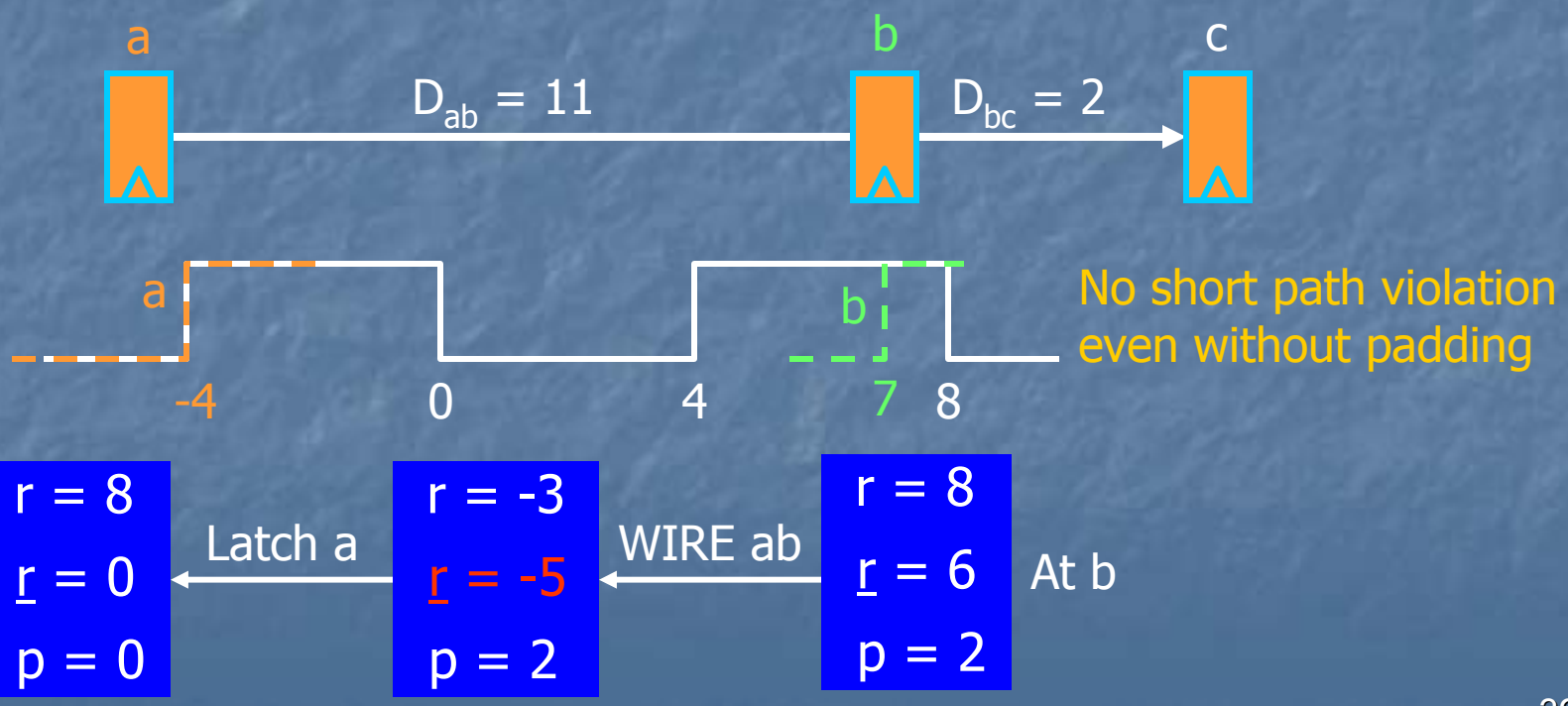
DDP: Generating Potential Delay Padding

- If $-Tp < \underline{r} < 0$, generate $p = \underline{r} + Tp$, propagate $\underline{r}$ through latch
- Neglect setup/hold time for simplicity



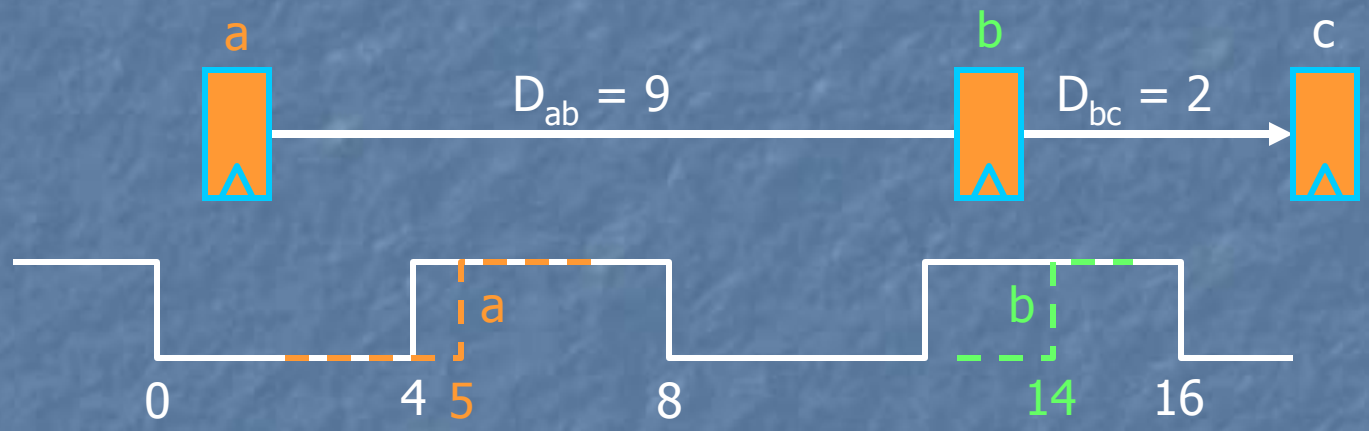
DDP: Small Earliest RAT

- If $\underline{r} < -Tp$, arrival time can be arbitrarily early without causing short path violation
- No padding, $p = 0$, $\underline{r} = 0$ after latch insertion



DDP: Medium Earliest RAT 1

- If $-Tp < \underline{r} < 0$, $\underline{r} + Tp < p$, update p, propagate $\underline{r}$, no actual padding,

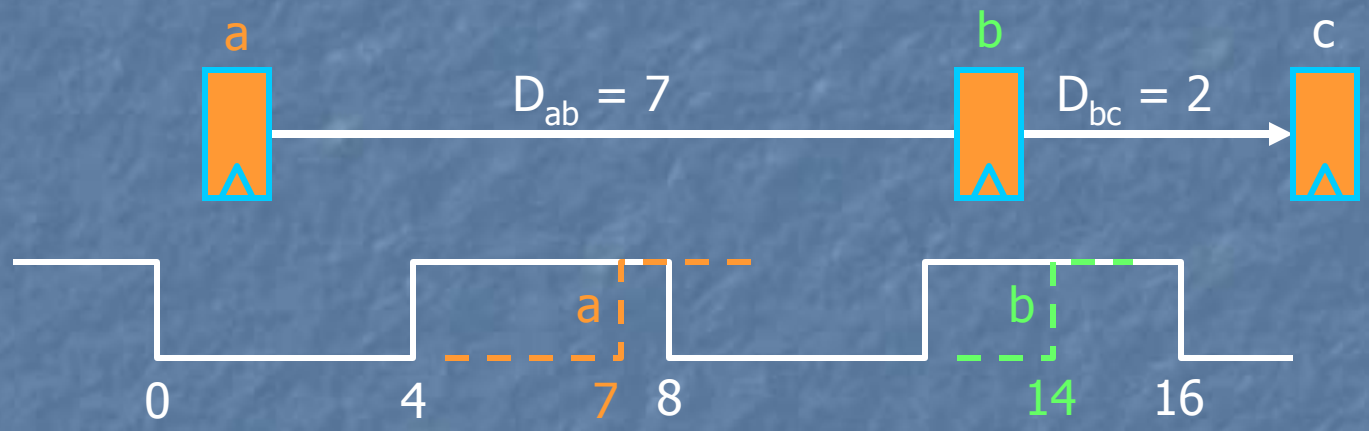


r = 8 r = 5 p = 1	Latch a ←	r = -1 r = -3 p = 2	WIRE ab ←	r = 8 r = 6 p = 2	At b

p = min (previous $\underline{r}$ + Tp, previous p), potential padding on (a, b)

DDP: Medium Earliest RAT 2

- If $-Tp < \underline{r} < 0$, $\underline{r} + Tp > p$, update p, propagate $\underline{r}$, no actual padding

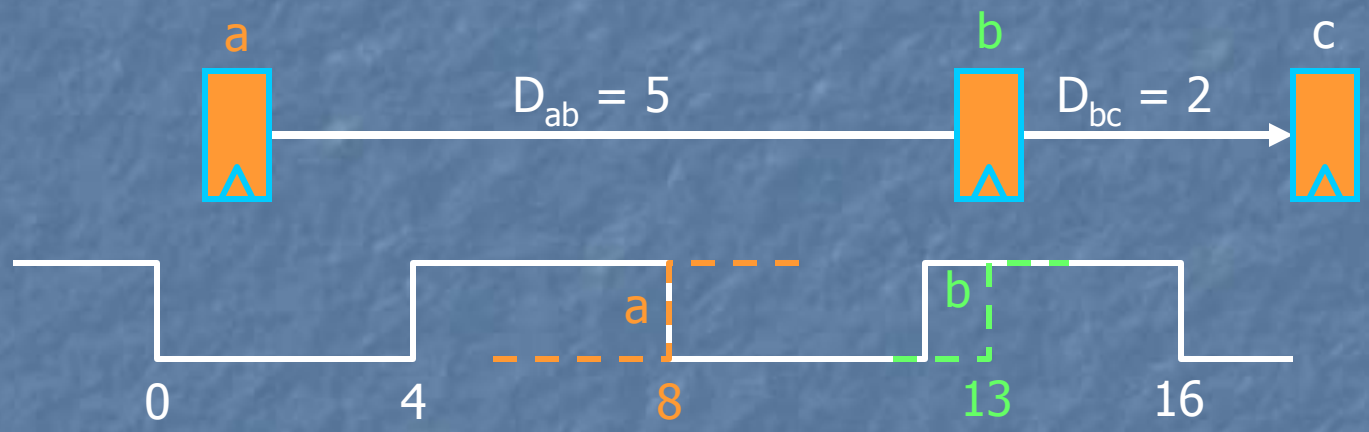


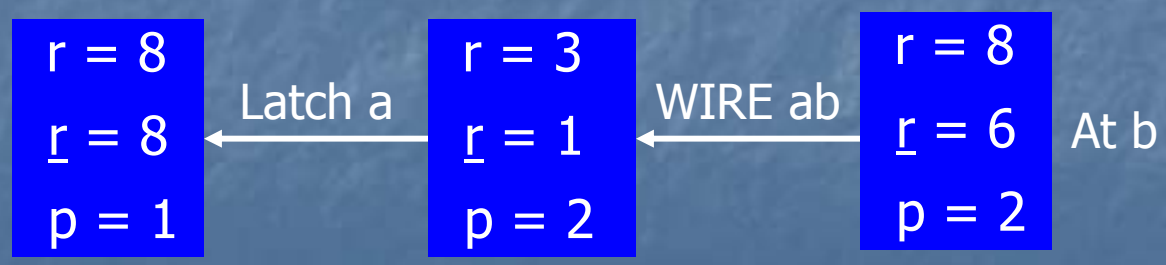
r = 8	Latch a	r = 1	WIRE ab	r = 8	At b
r = 7	←	r = -1	←	r = 6	
p = 2		p = 2		p = 2	

p = min (previous r + Tp, previous p), potential padding on (b, c)

DDP: Large Earliest RAT 1

- If $0 < \underline{r} < p$, instantiate padding of amount $\underline{r}$! p decreases by $\underline{r}$, $\underline{r}$ = T,

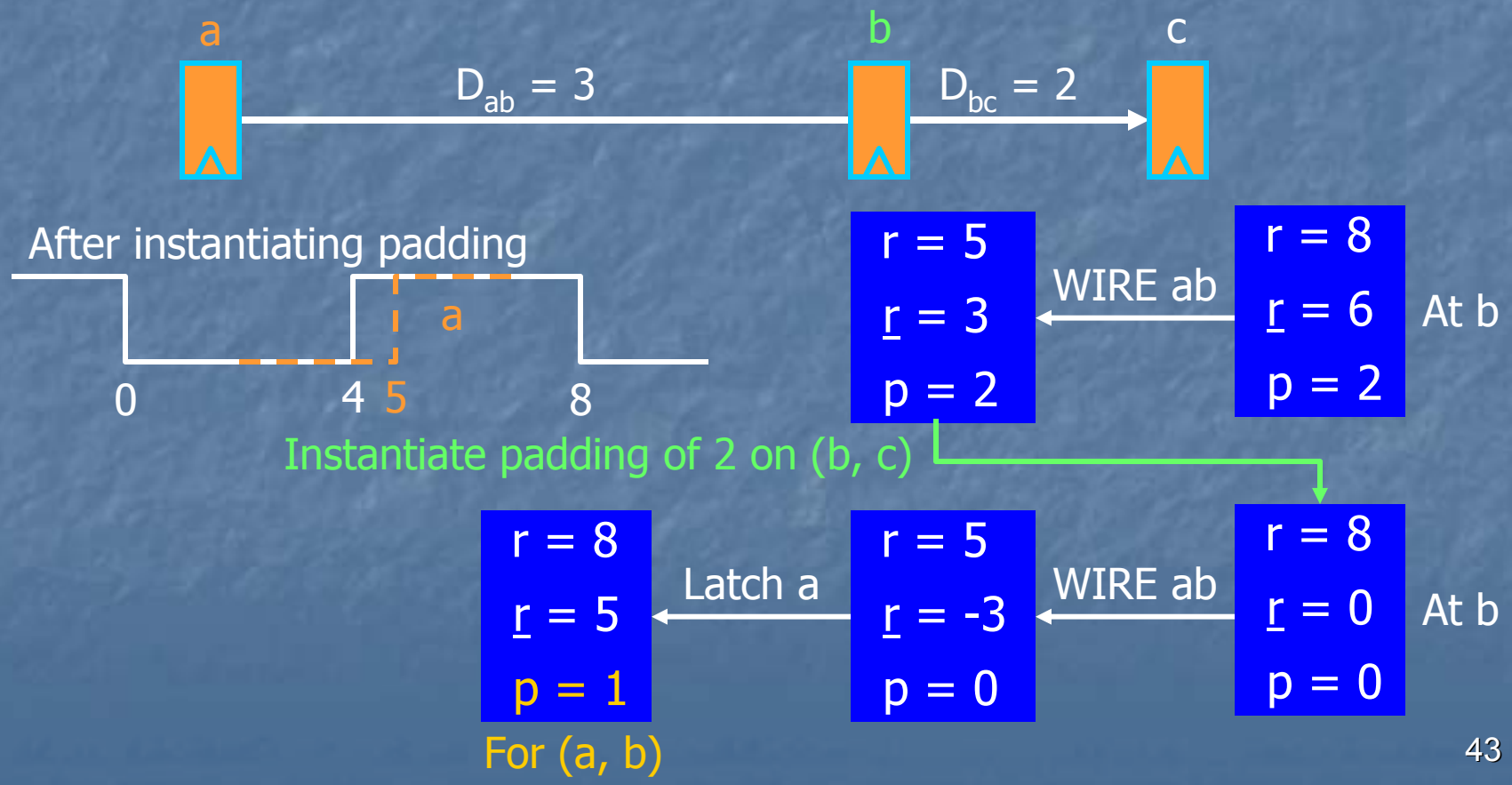




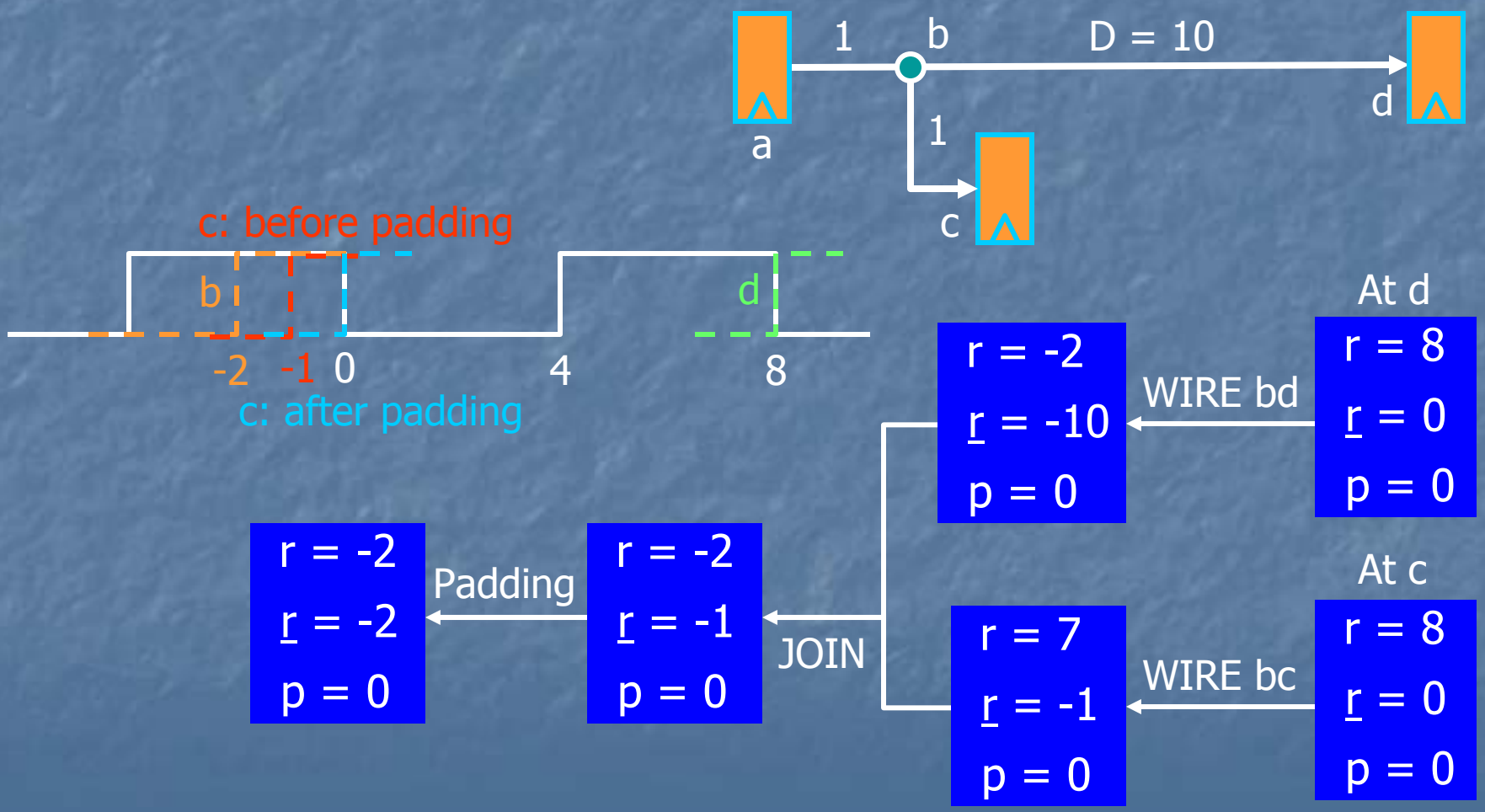
Instantiate padding of 1 on either (a, b) or (b, c)

DDP: Large Earliest RAT 2

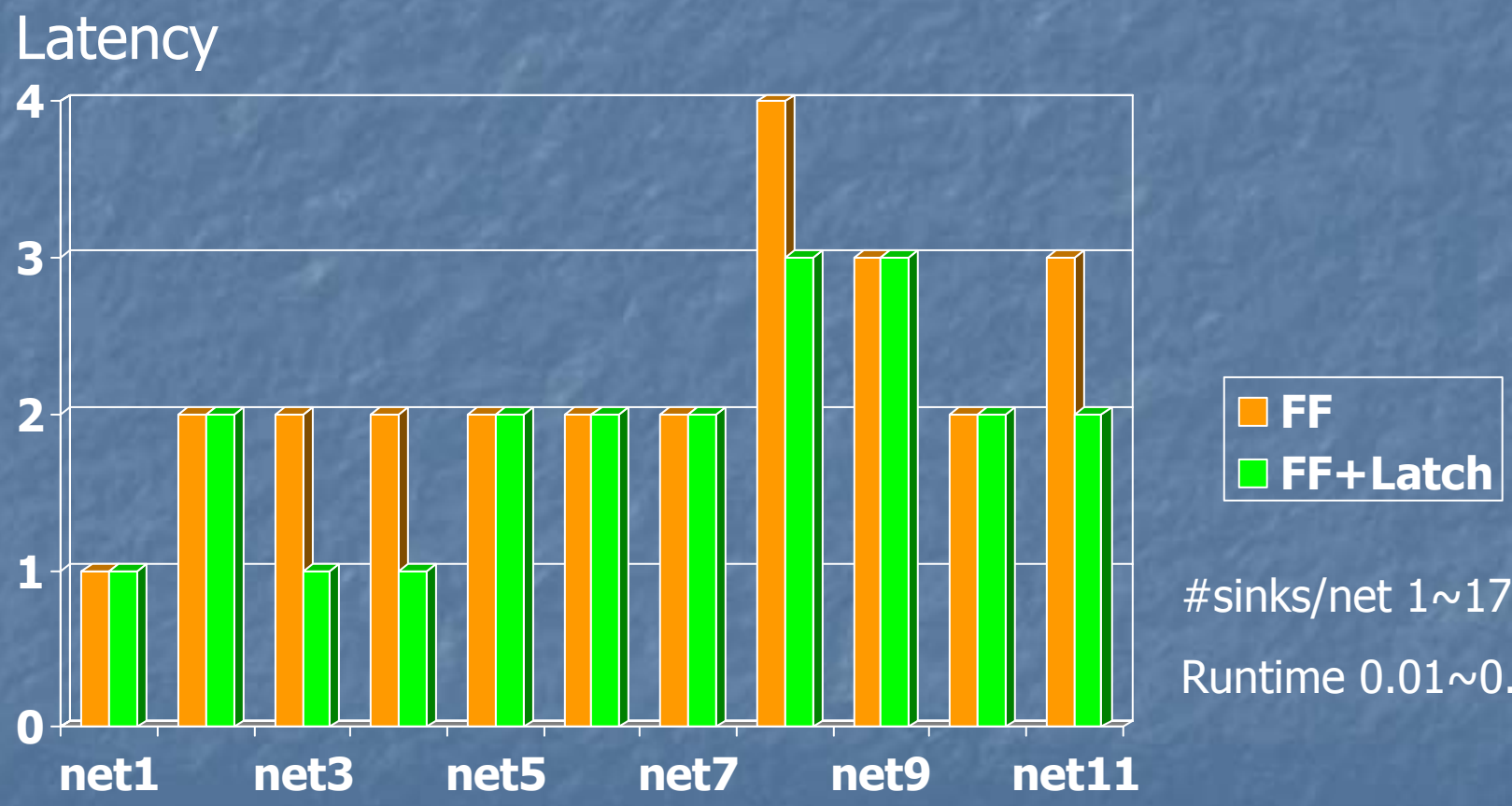
- If $\underline{r} > p > 0$, instantiate padding of amount p ! reset p and $\underline{r}$



DDP: Multi-fanout Tree



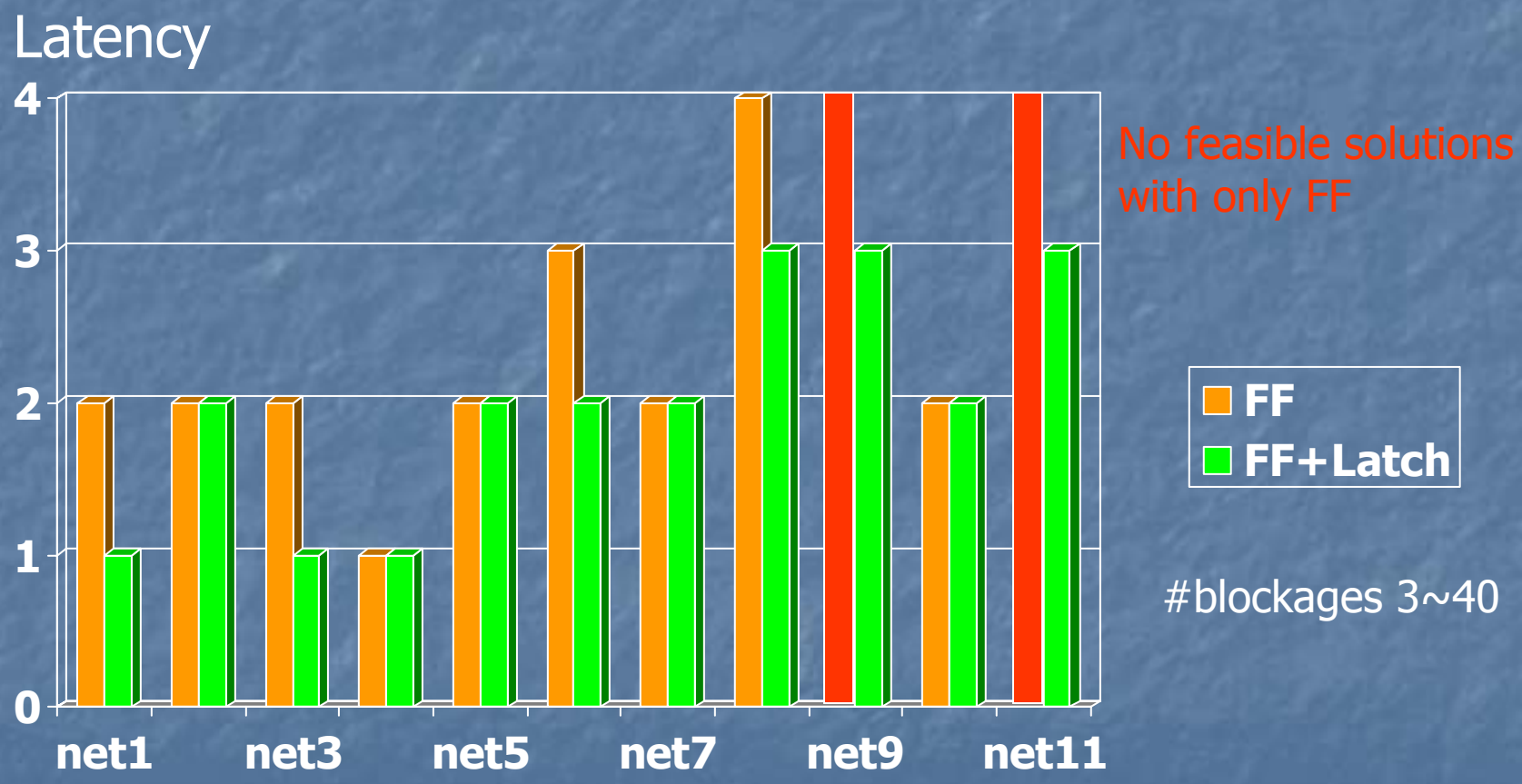
Experiment: MiLa without Blockages



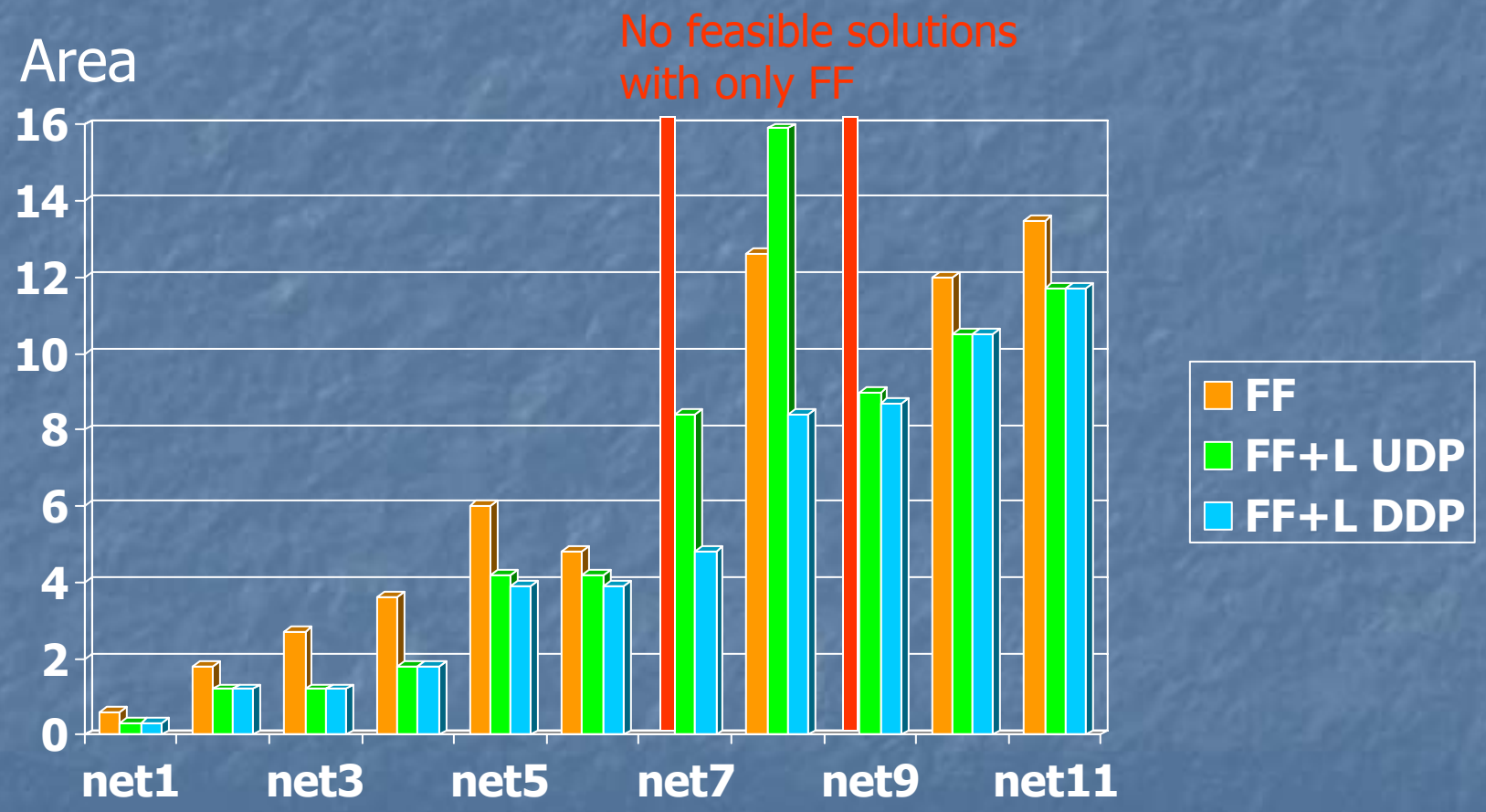
#sinks/net 1~17

Runtime 0.01~0.07 s

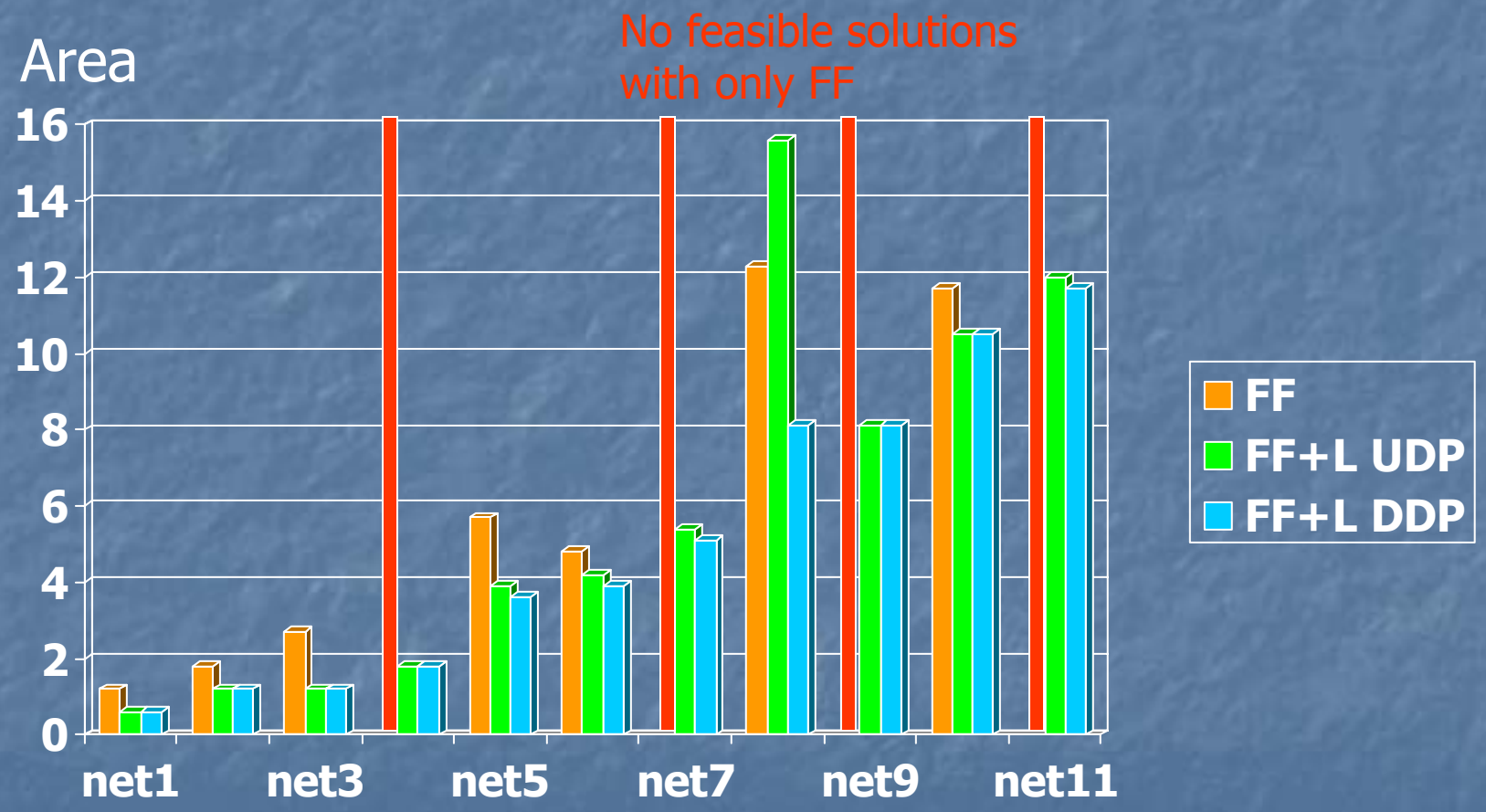
Experiment: MiLa with Blockages



Experiment: GiLa without Blockages



Experiment: GiLa with Blockages



Conclusions

- Wire pipelining becomes a necessity and requests more on timing and power/area
- Advantages of using latches: area and timing flexibility
- Short path constraint can be solved
  - Mixed latch and flip-flop
  - Proper delay padding
- Timing advantage of latch can be traded to variation tolerance

Acknowledgement

- Collaboration with *Dr. Zhao* of Freescale
- Code implementation and experiment by *Mr. V. Seth* of Texas A&M University
- Helpful discussions with *Dr. Cocchini* of Intel