國立交通大學電子工程學系

PushPull: Short Path Padding for Timing Error Resilient Circuits

YU-MING YANG
IRIS HUI-RU JIANG
SUNG-TING HO

IRIS Lab
National Chiao Tung University

Outline

- Introduction
- Problem Formulation
- Algorithm - PushPull
- Experimental Results
- Conclusion

Outline

Introduction

Problem Formulation

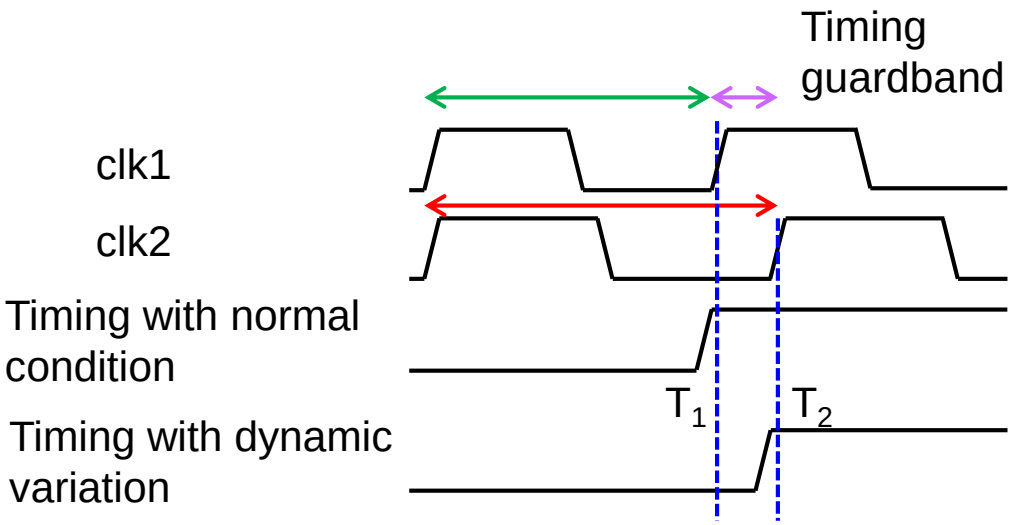
Algorithm - PushPull

Experimental Results

Conclusion

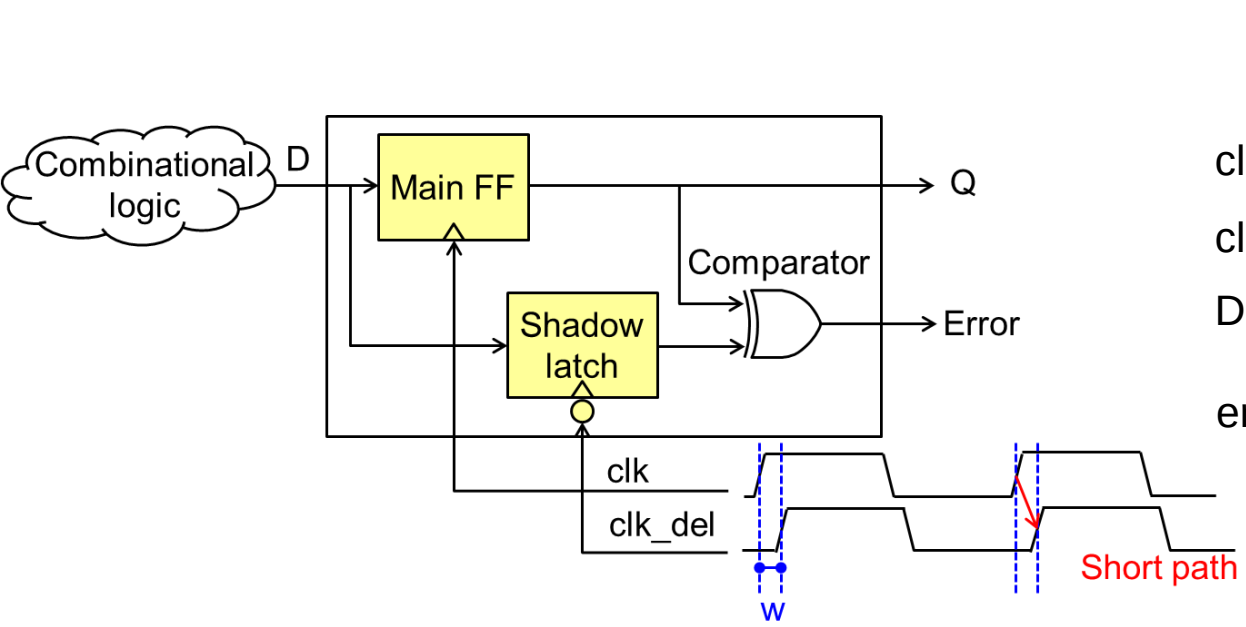
Introduction

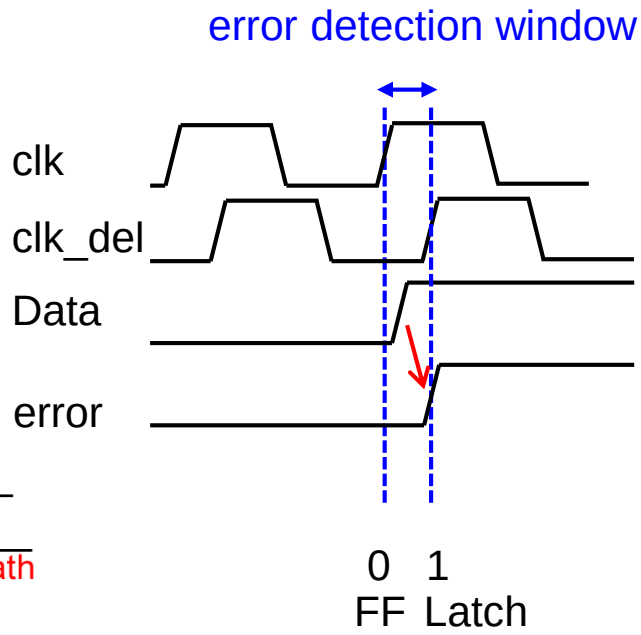
- **Timing characterization is difficult due to a wide range of dynamic variations**
  - Supply voltage droops, process variations, etc.
- **A timing guardband is reserved to ensure correct functionality**
  - Degrade circuit performance, i.e., limit the clock frequency



Resilient Circuit

- **Eliminate the guardband by error detection and correction**
  - Ex: Razor flip-flop (one error-detection circuit)
  - Correction through instruction replay
  - Cons: short path issue

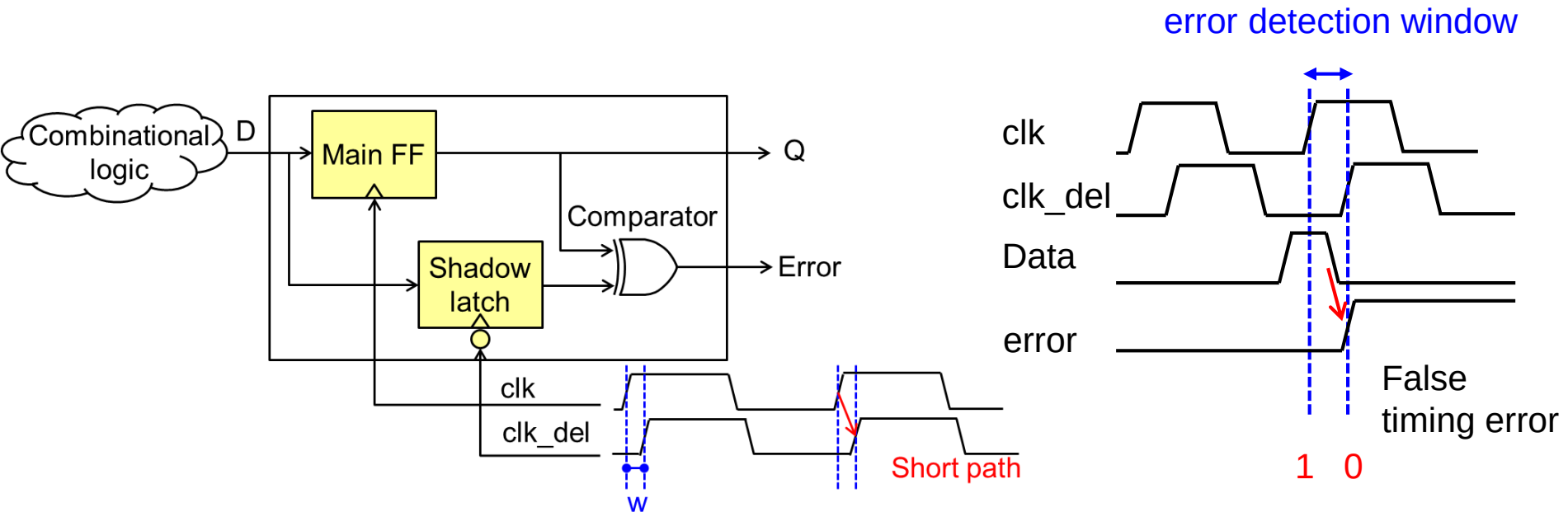




D. Ernst *et al.*, "Razor: a low-power pipeline based on circuit-level timing speculation" *MICRO*, 2003

Short Path Issue

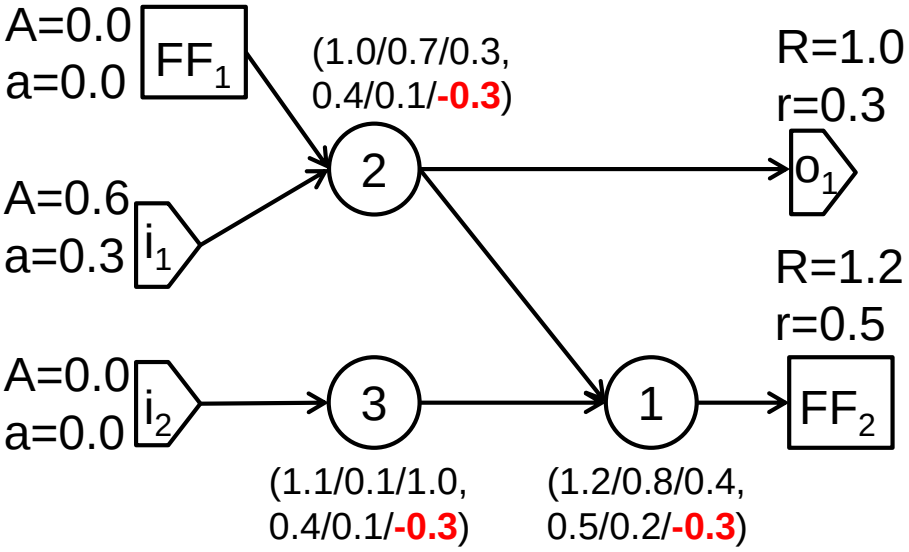
- **Short paths should exceed the error detection window**
  - Otherwise, may detect false timing errors
- **Require a significant hold time margin for short paths**
- **Focus on short path padding in this paper**



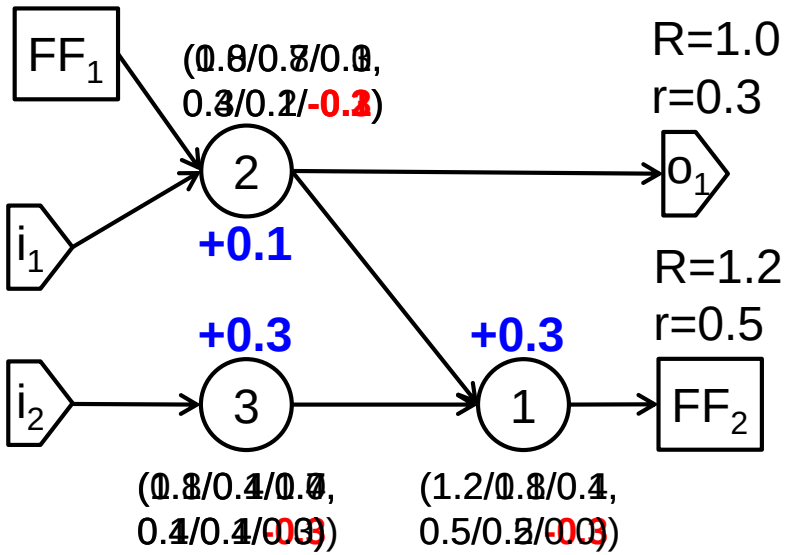
Previous Work

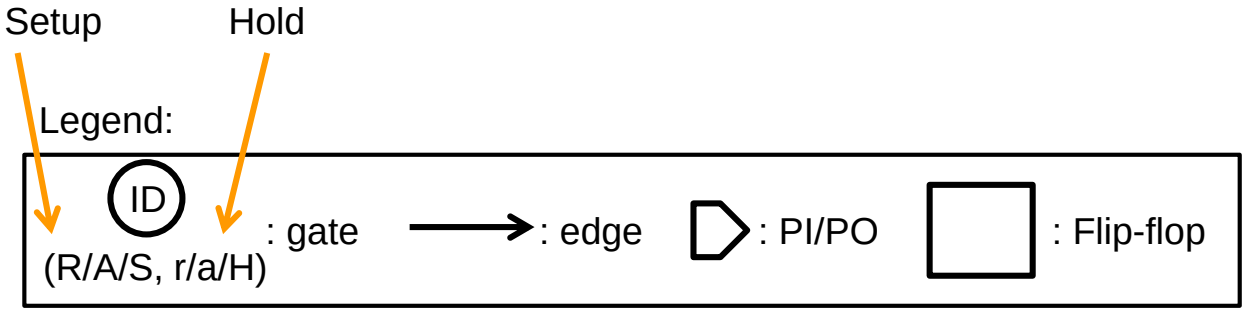
- **Determine the padding delay path by path**
  - Combine with clock skew scheduling to minimize the clock period at logic synthesis stage
    - J.P. Fishburn, IEEE TC, 1990
    - R.B. Deokar and S.S. Sapatnekar, ISCAS, 1994
    - S.H. Huang et al., DAC, 2007
- **Determine the gate/wire to insert padding delay**
  - Solve by linear programming
    - N. V. Shenoy et al., ICCAD, 1993
  - Heuristics
    - Pad the gate with the largest setup slack
      - US Patent 6,990,646,2006 and 7,278,126,2007
    - Pad the gate passed by most hold violating paths
      - Y. Liu et al., DAC, 2011

Short Path Padding Example



The largest setup slack

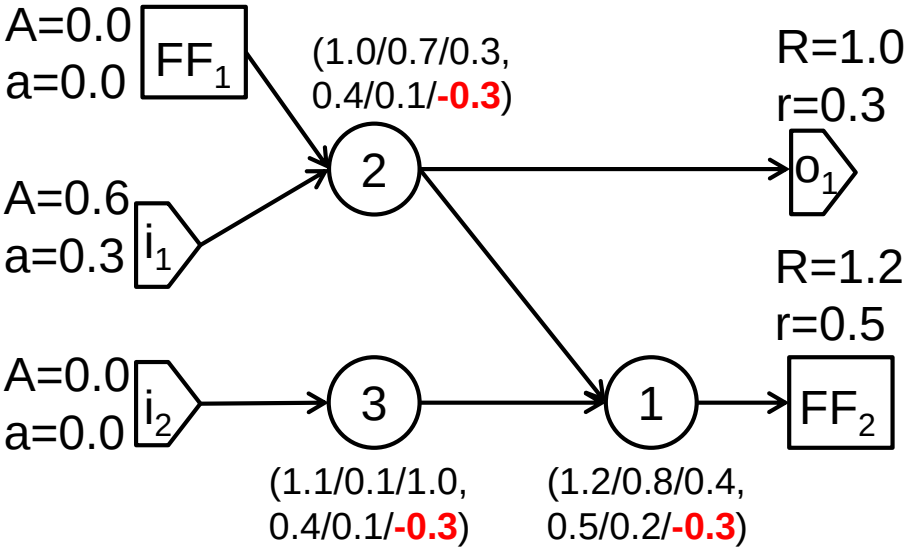




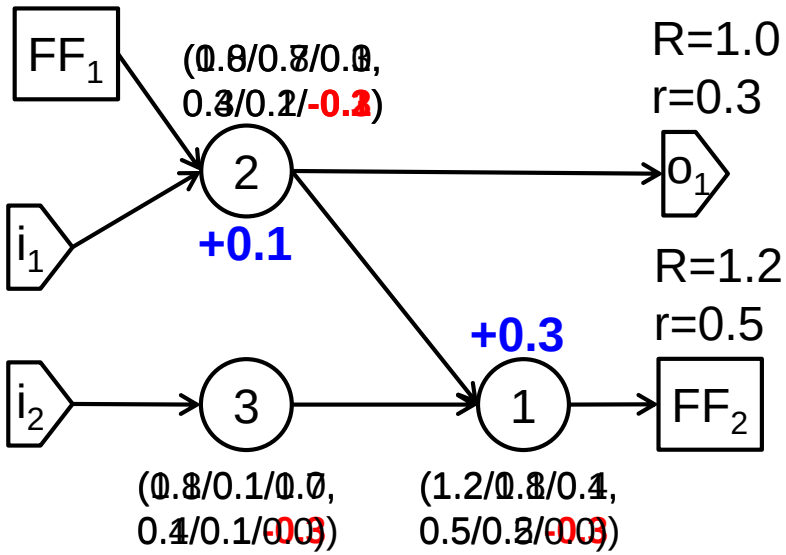
Unfix

ID	1	2	3
Delay	0.1	0.1	0.1

Short Path Padding Example (cont'd)

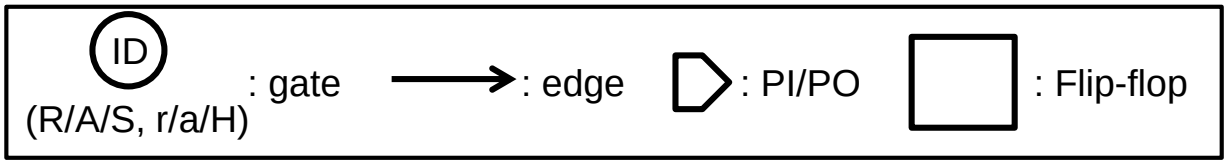


The most hold violating path



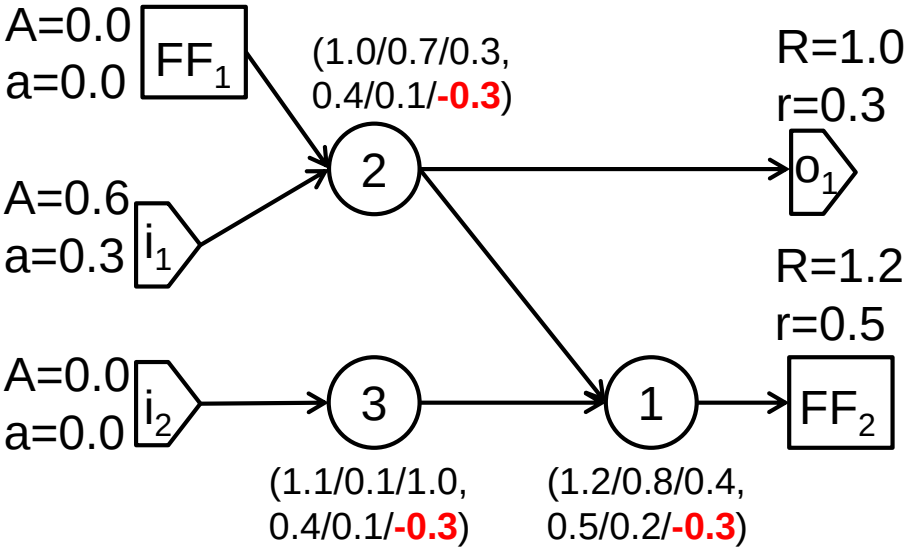
Unfix

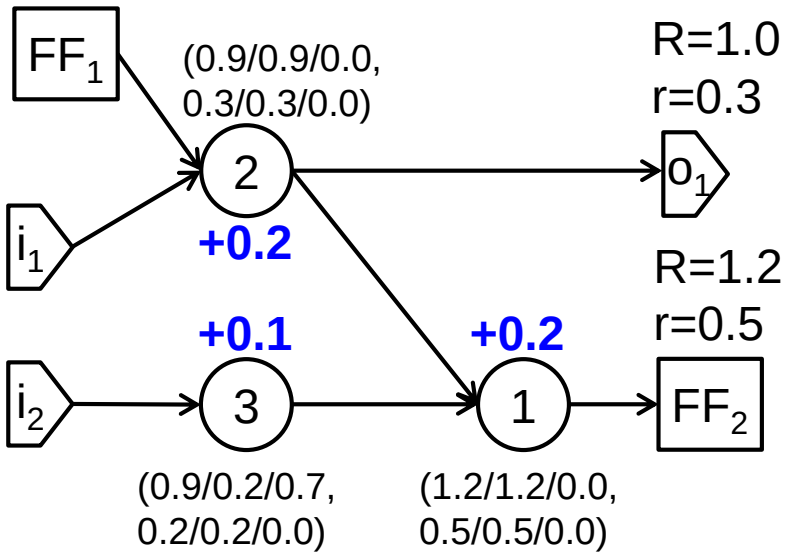
Legend:



ID	1	2	3
Delay	0.1	0.1	0.1

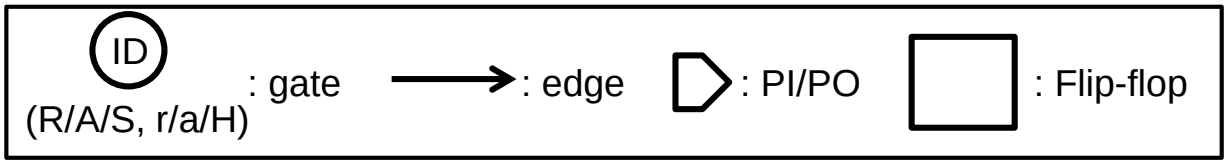
Short Path Padding Example (cont'd)





Optimal padding delay = 0.5

Legend:



ID	1	2	3
Delay	0.1	0.1	0.1

Our Contributions

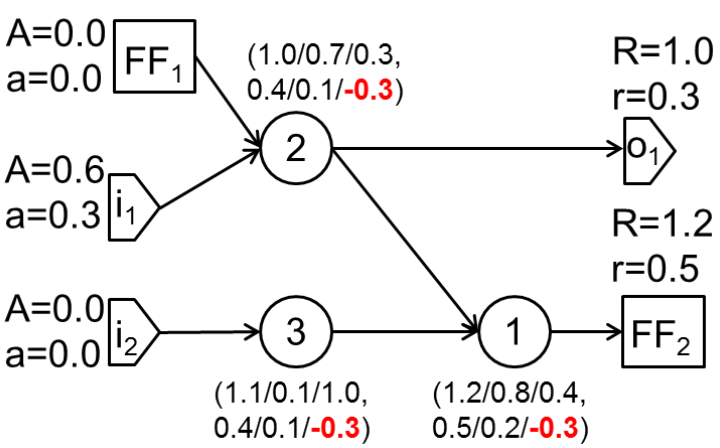
- **Find the padding values and locations with a global view**
  - Compute the fanout padding flexibility of each gate
- **Realize delay padding at the post-layout stage**
  - Coarse-grained delay padding: by spare cells
  - Fine-grained delay padding: by dummy metal

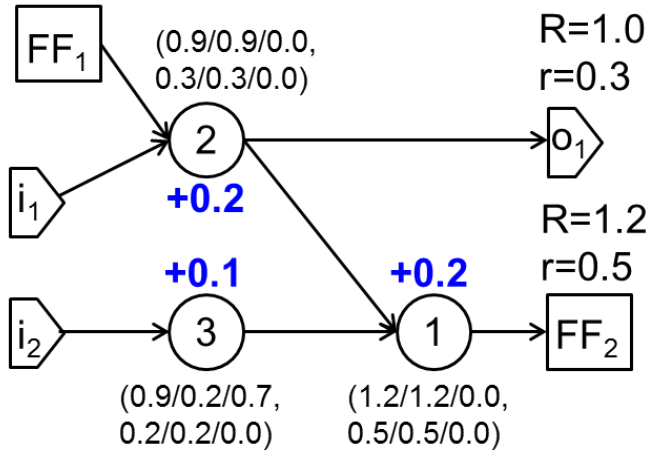
Outline

- Introduction
- **Problem Formulation**
- Algorithm - PushPull
- Experimental Results
- Conclusion

Problem Formulation

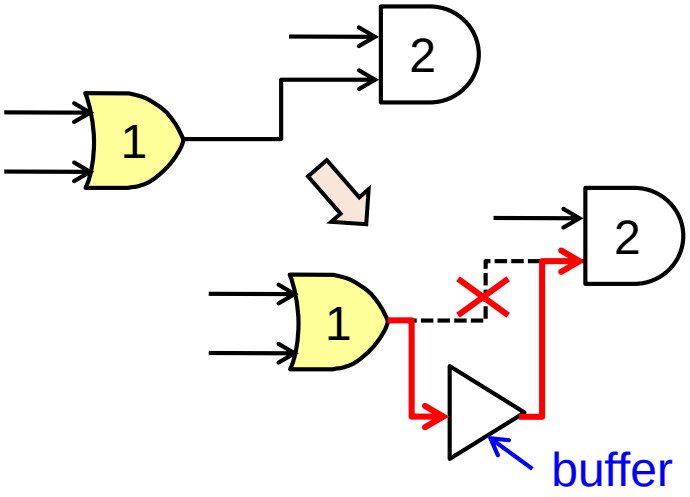
- **Given**
  - Placed and routed resilient design
  - Cell library
  - Spare cells and dummy metal information
  - Target clock period and error detection window
- **Goal**
  - Pad short paths
  - Satisfy Setup/hold timing constraints
  - Minimize the padding overhead





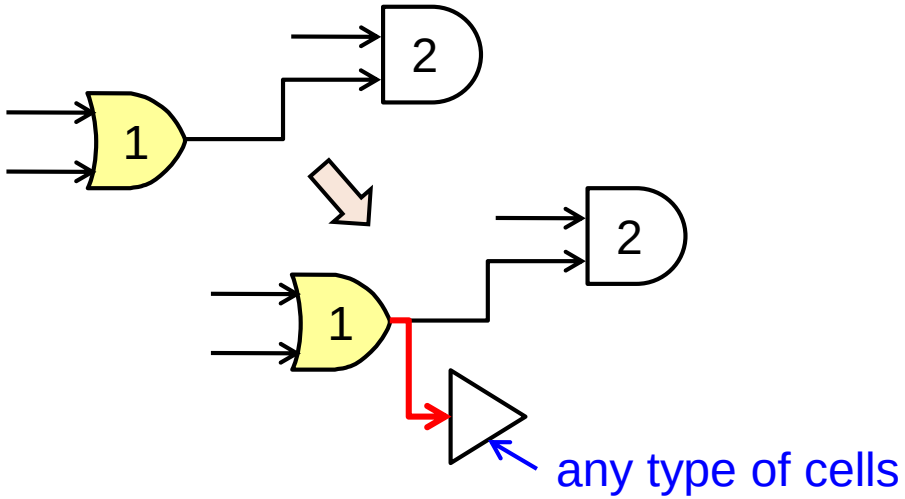
Delay Padding

- **Lengthen short paths by buffer insertion and extra capacitance hook-up**



Buffer insertion

Padding wire



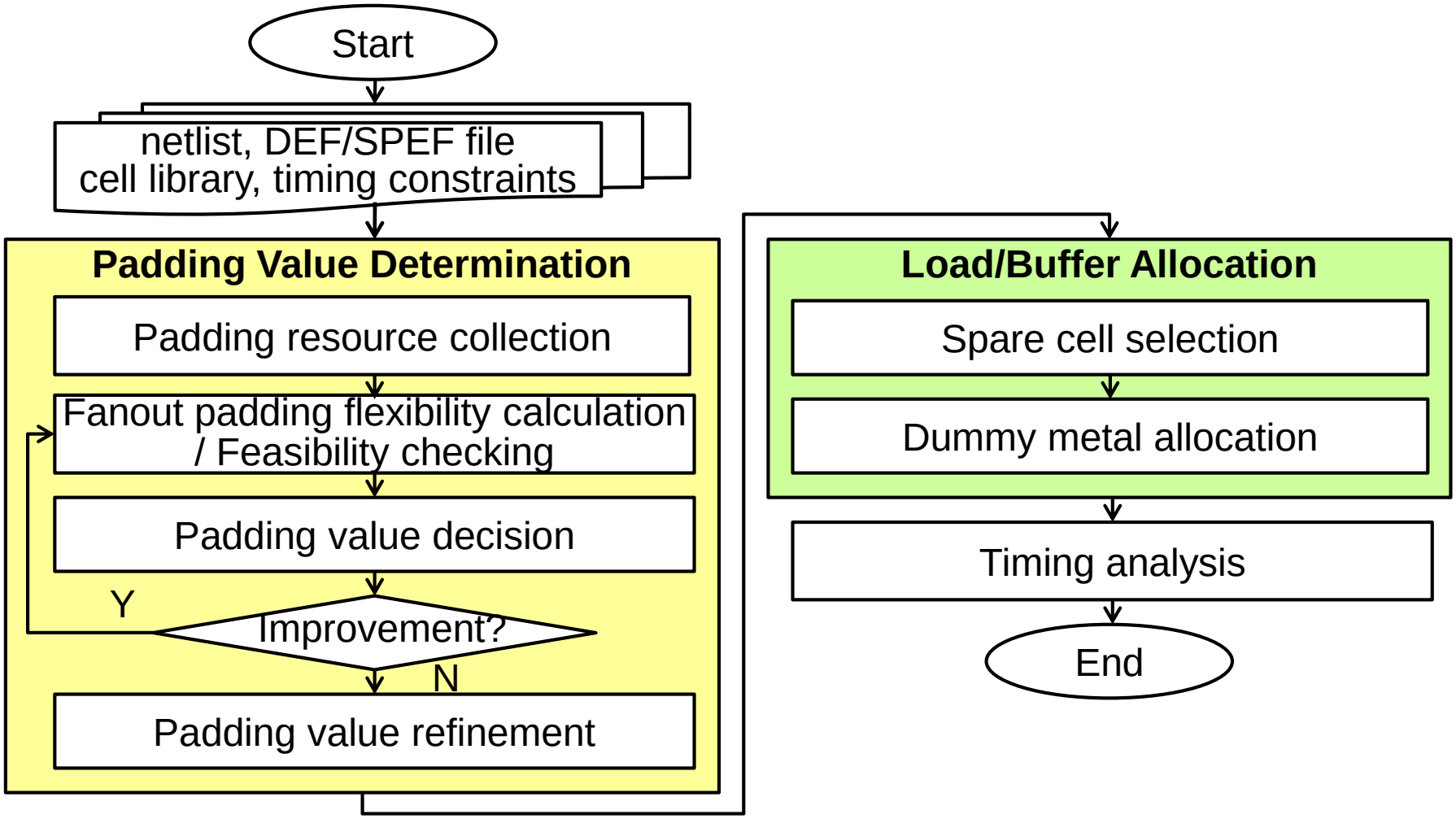
Extra cap hook-up

Padding gate

Outline

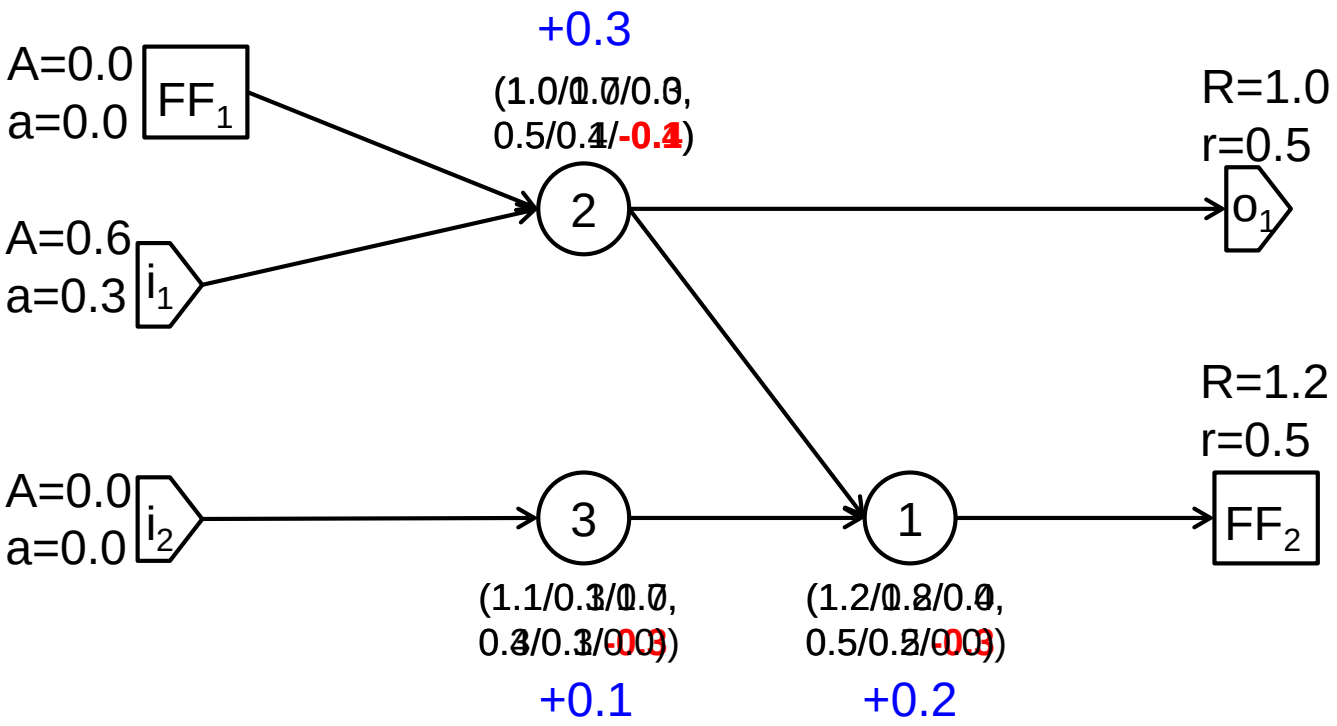
- Introduction
- Problem Formulation
- **Algorithm - PushPull**
- Experimental Results
- Conclusion

Overview of PushPull



Example: Padding Value Determination

First, pad gates



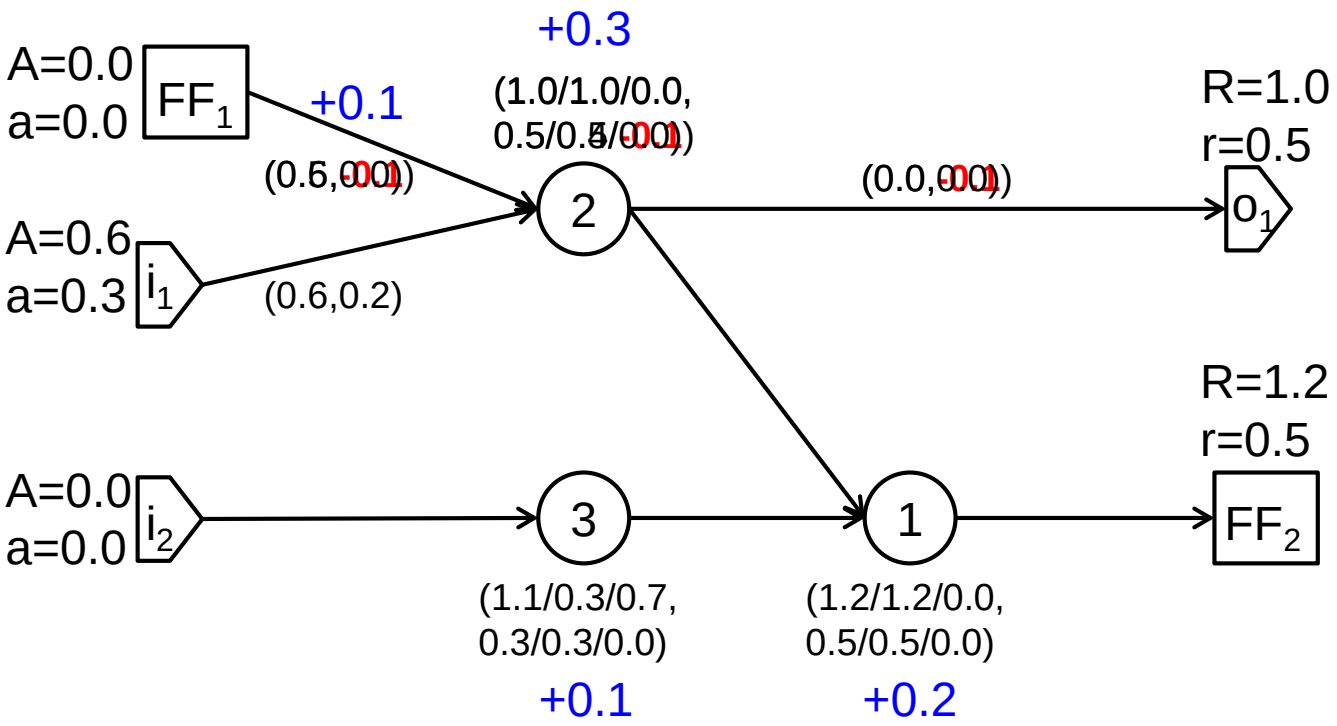
Legend:

ID (R/A/S, r/a/H) : gate → : edge : PI/PO : Flip-flop

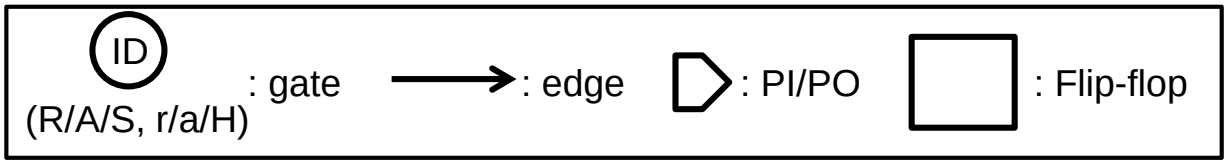
ID	1	2	3
Delay	0.1	0.1	0.1

Example: Padding Value Determination

Second, pad wires

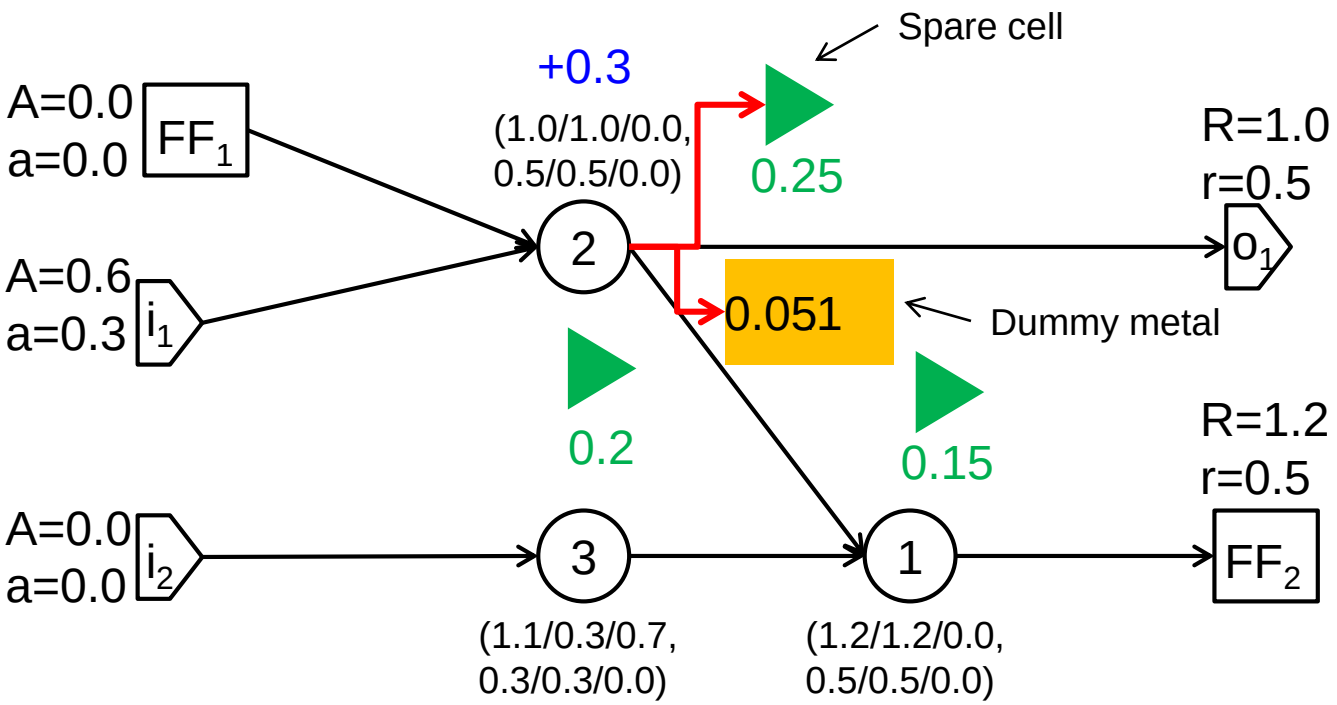


Legend:

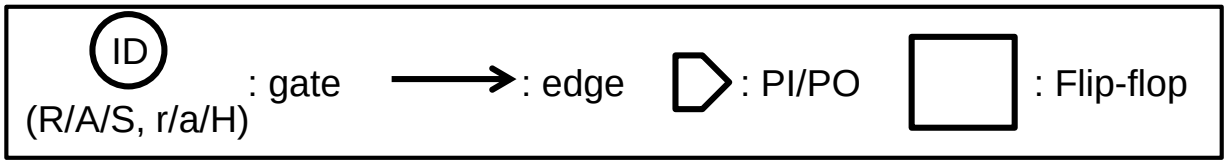


ID	1	2	3
Delay	0.1	0.1	0.1

Example: Load/Buffer Allocation

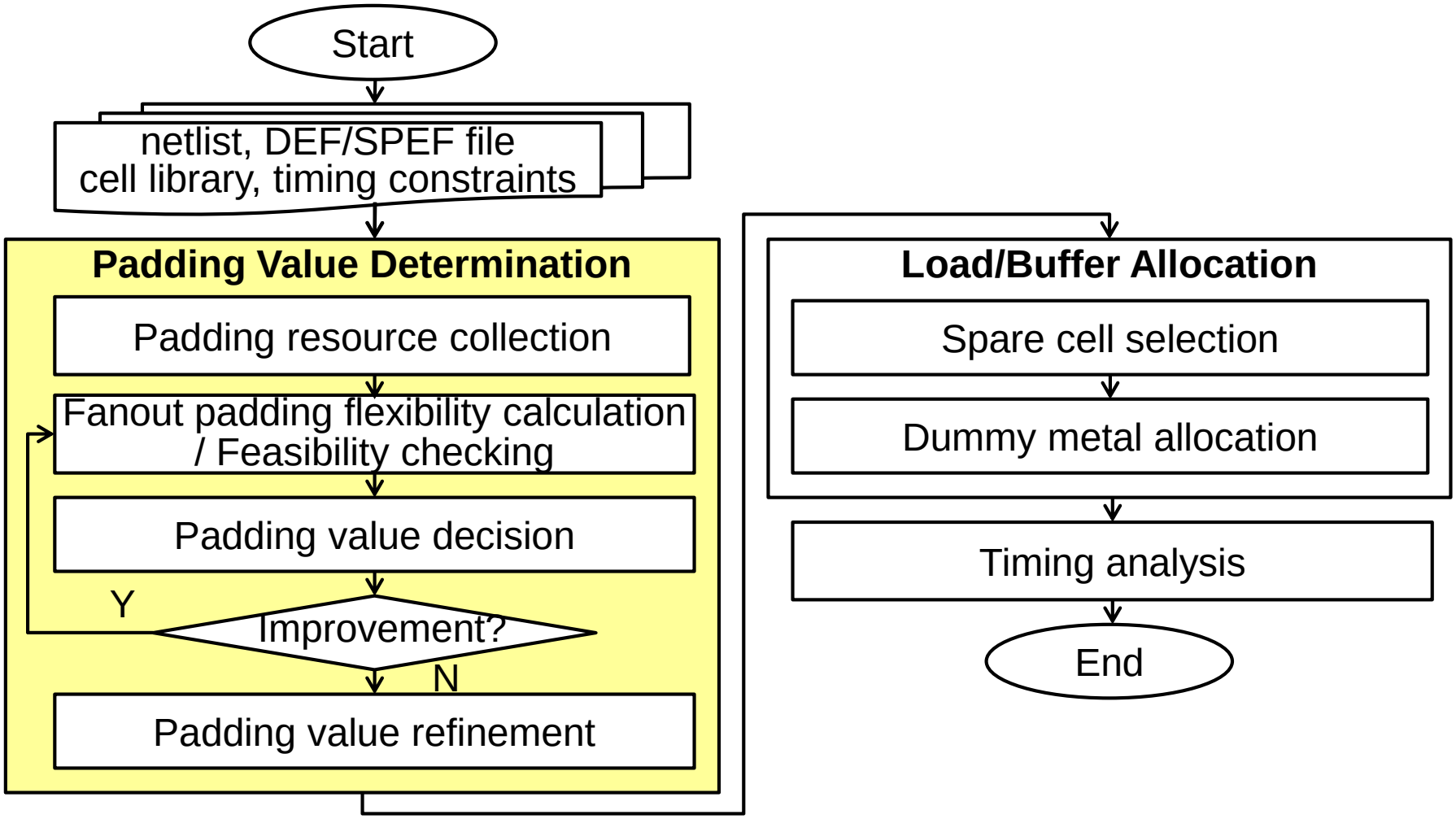


Legend:



ID	1	2	3
Delay	0.1	0.1	0.1

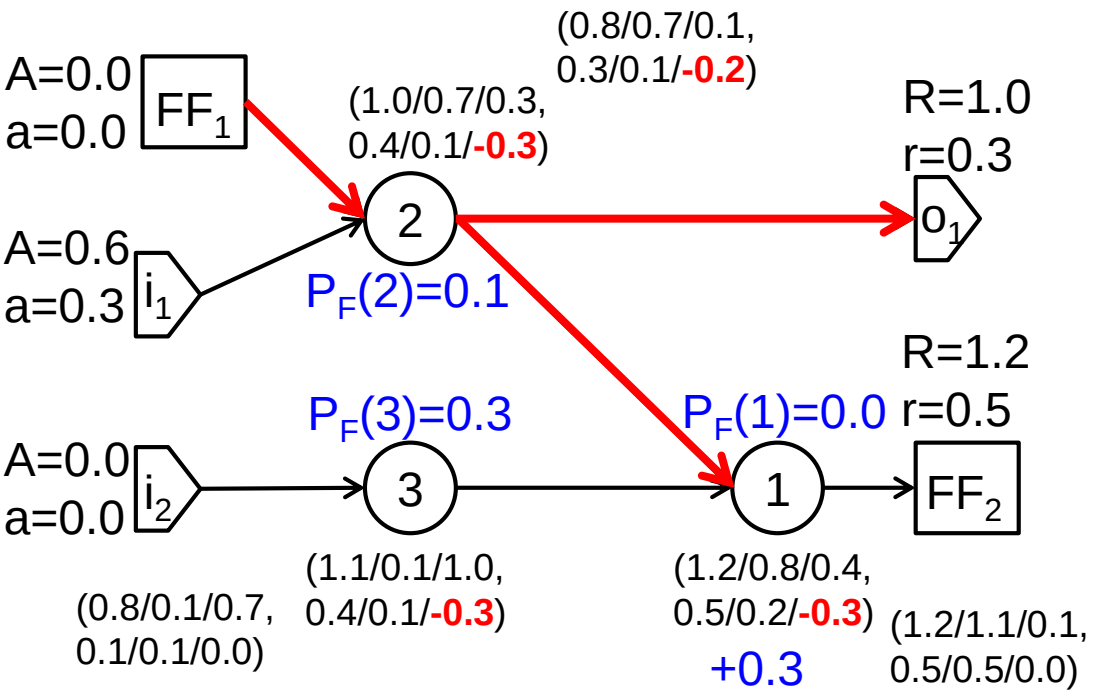
Padding Value Determination



Fanout Padding Flexibility Calculation

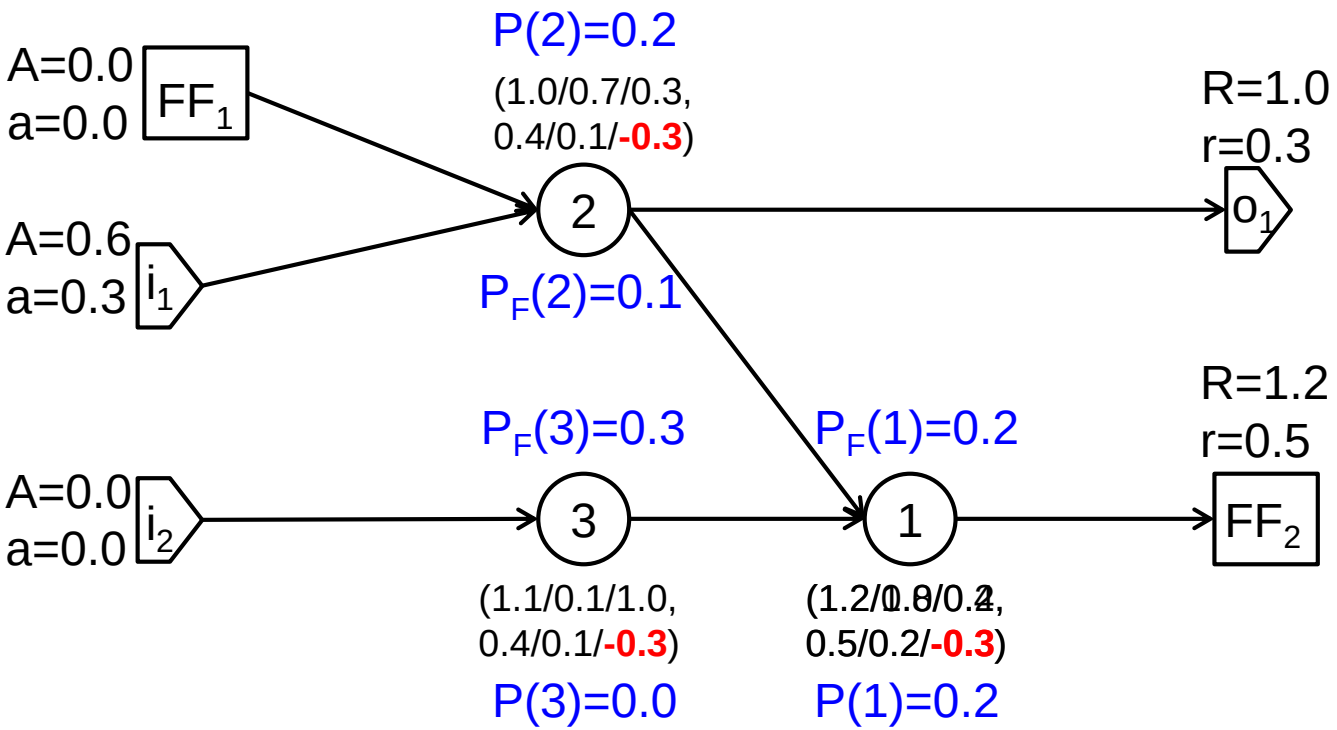
- **Fanout padding flexibility $P_F(i)$ of gate i**
  - Reflect the maximum padding value allowed on its whole fanout cone with a global view

$$P_F(i,j) = \begin{cases} 0, if\ g_i \in PO\ or\ H(i) \geq 0; \\ \min\{0, \min\{H'(i,j) | e(i,j) \in E\}\} - H(i), otherwise. \end{cases}$$



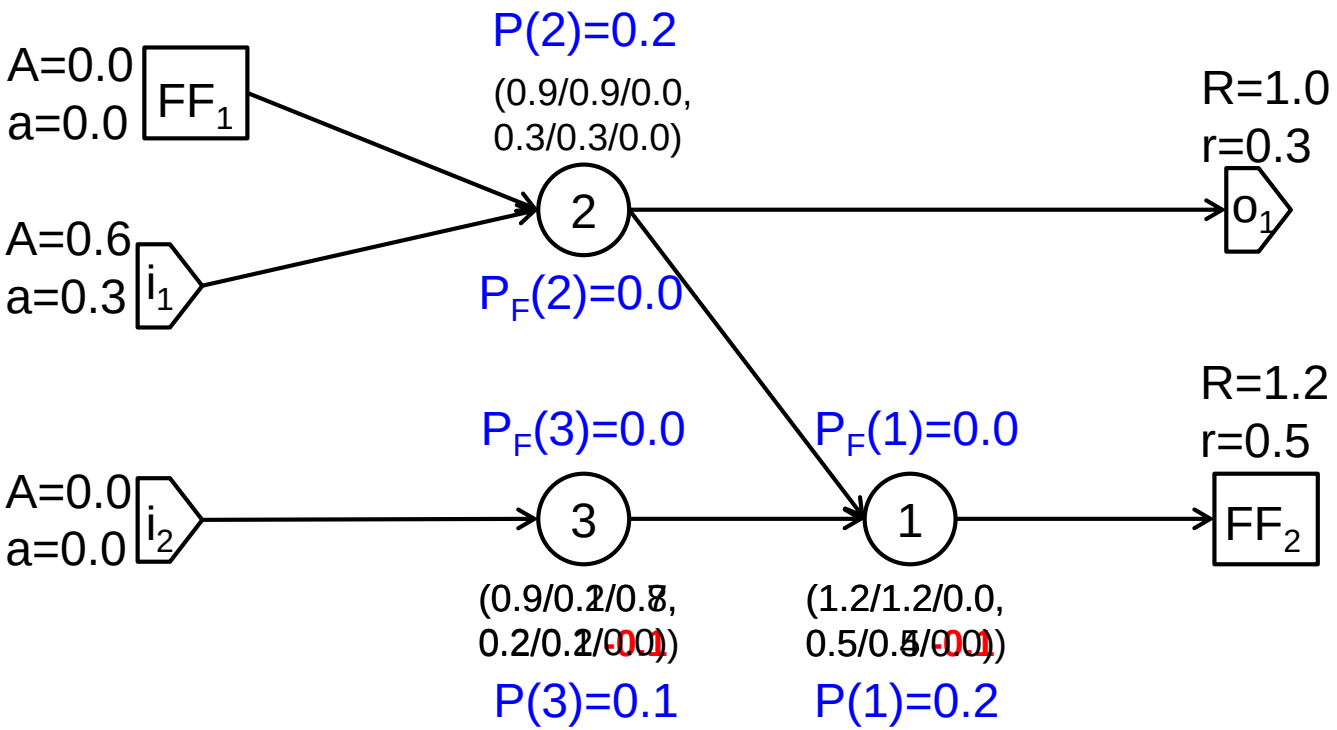
Padding Value Decision

- **Decide the padding value of each gate in the topological order**
  - Only pad the remaining slack
    - $P(i) = \max\{P_{saf}(i) - P_F(i), 0\}$



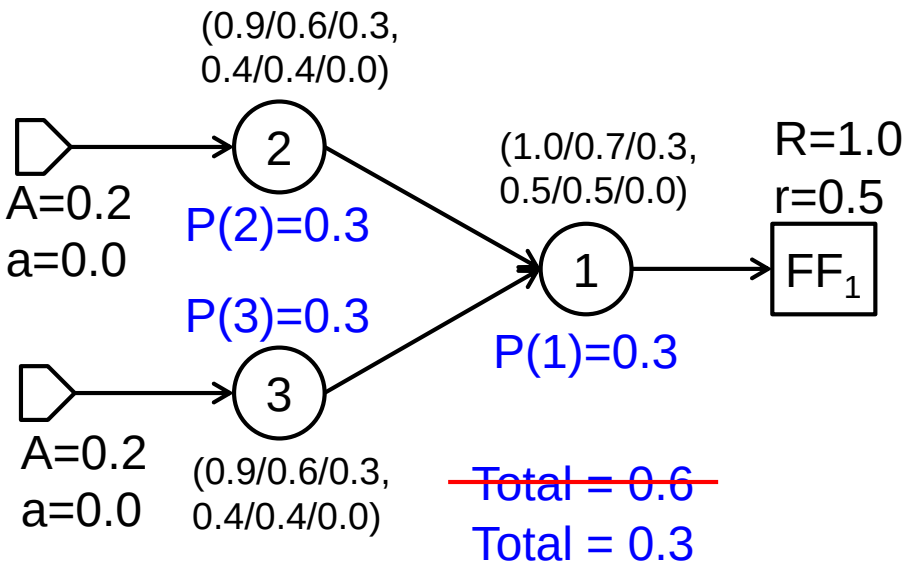
Padding Value Decision

- **Iterate until no more improvement**

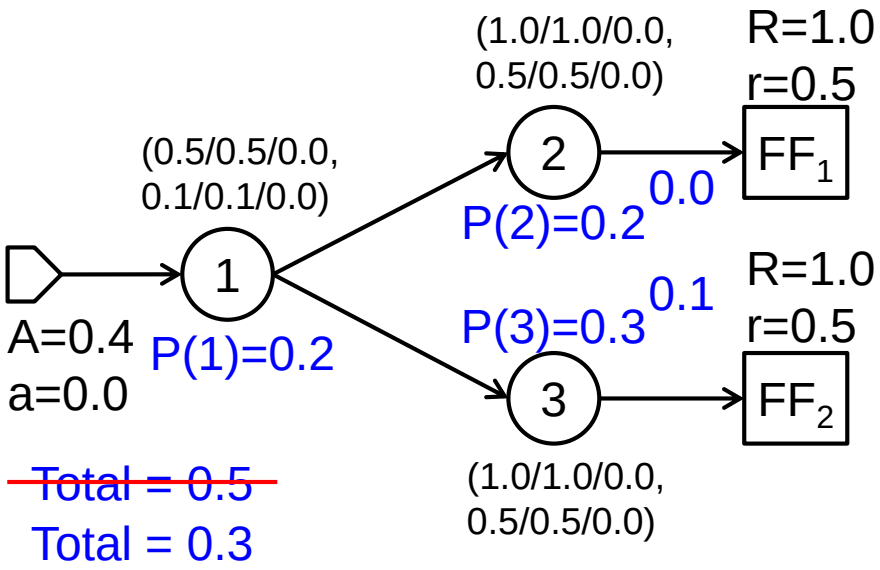


Padding Value Refinement

- **Further reduce the total padding values with forked path**
  - Push the padding values toward the gates where two or more paths fork



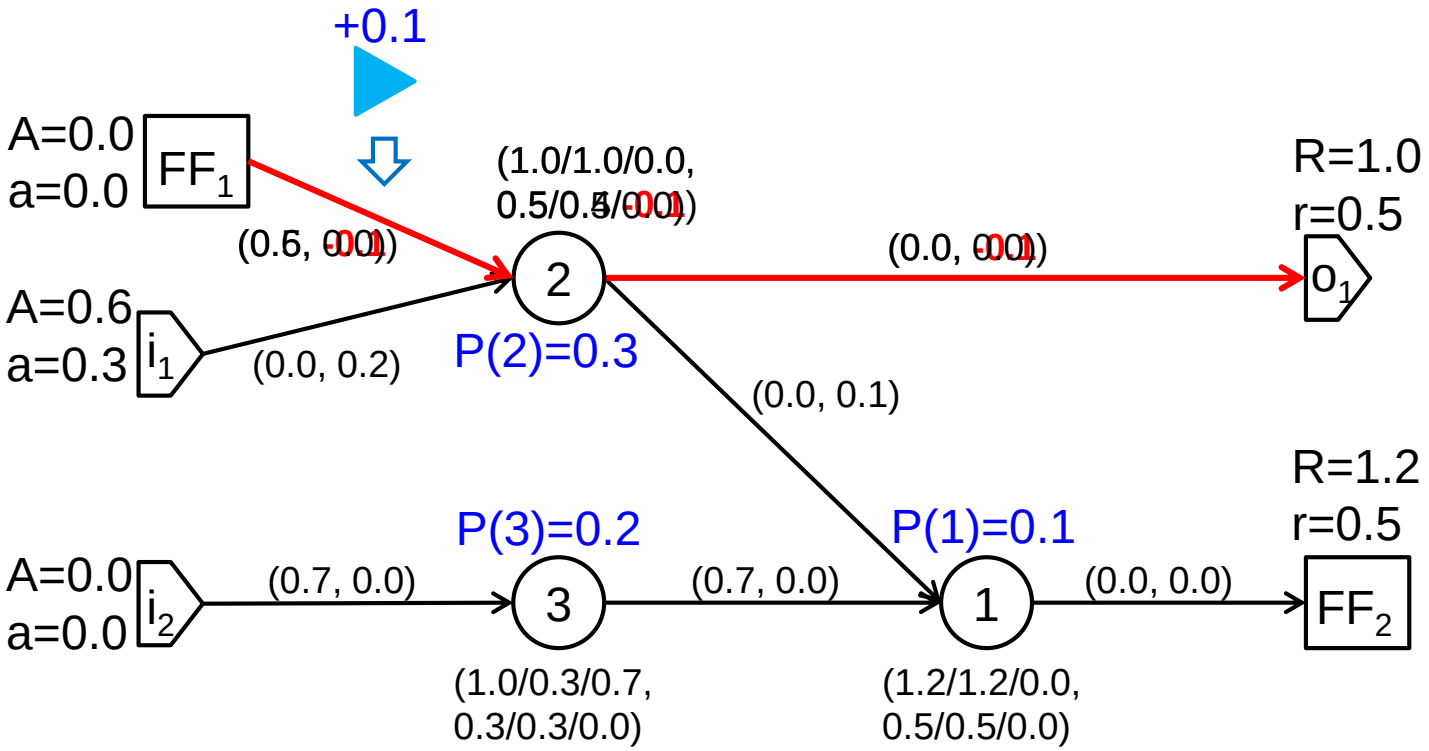
Joined short paths



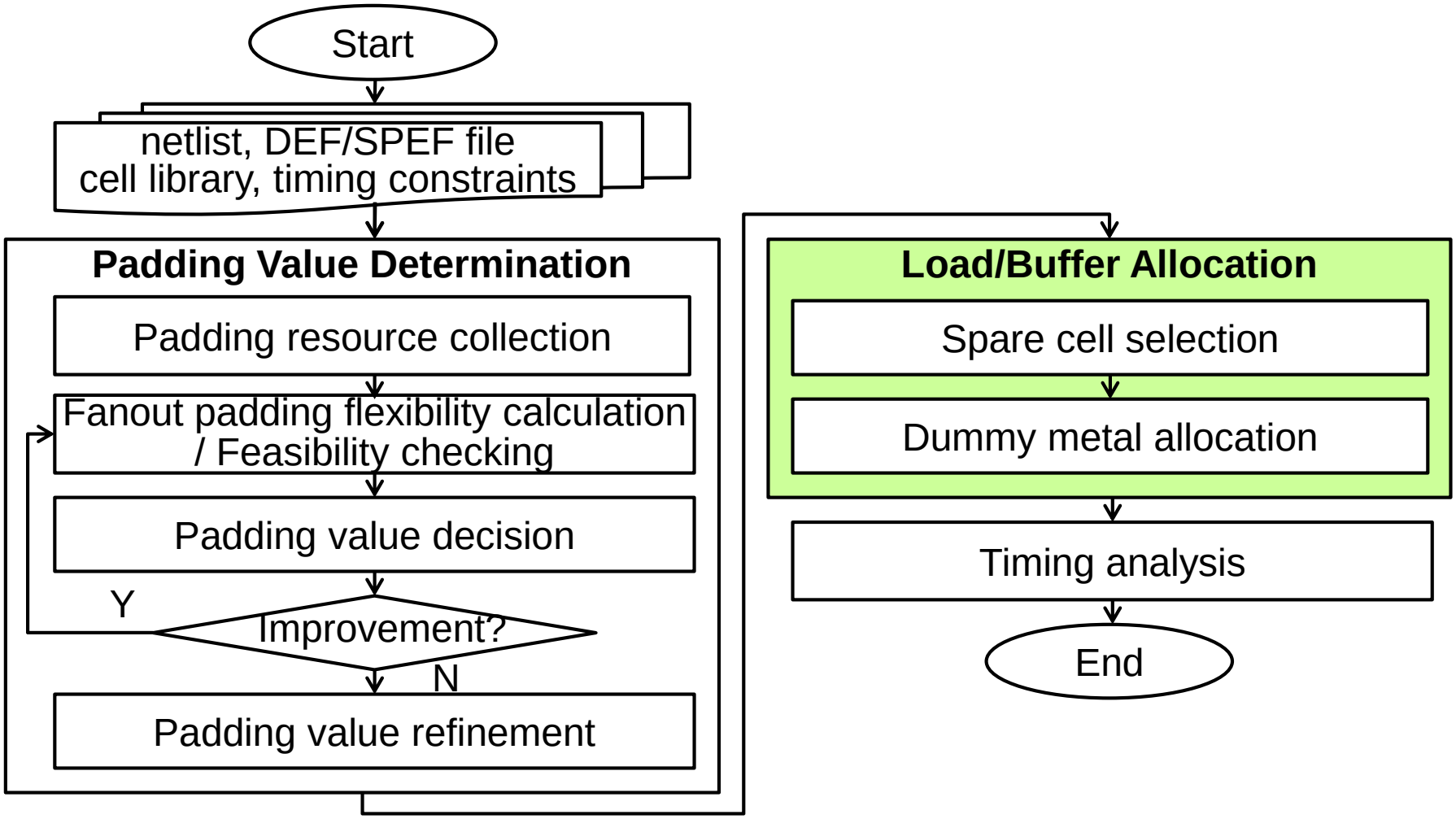
Forked short paths

Buffer Insertion

- **When hold violations cannot be fully cleaned by padding on gates**
  - Padding on wires (buffer insertion)



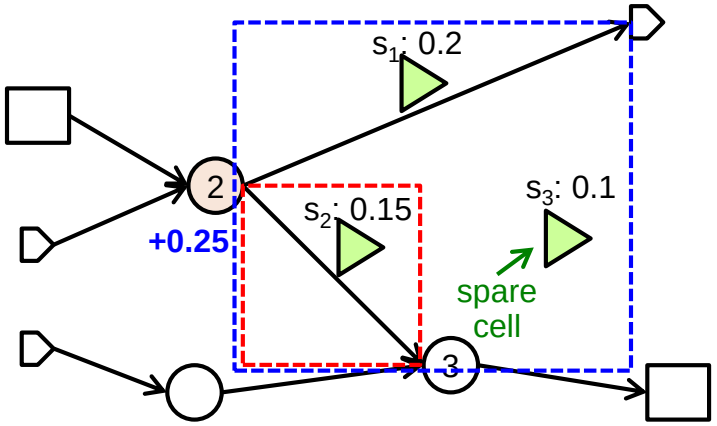
Load/Buffer Allocation



Spare Cell Selection (1/2)

- **Extract the available spare cells located within the bounding box of its fanout net**

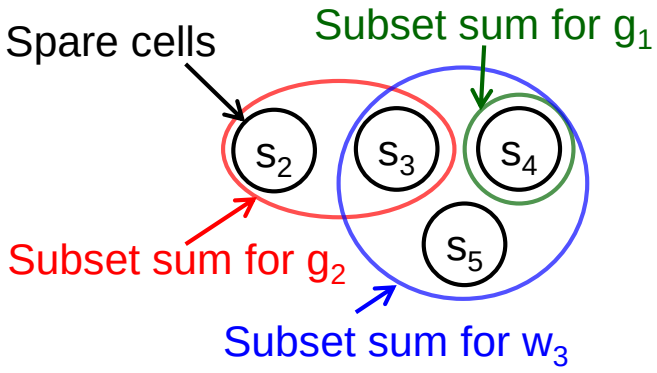
- **Find suitable spare cell candidates for gate/wire**
  - Reduce to the subset sum problem

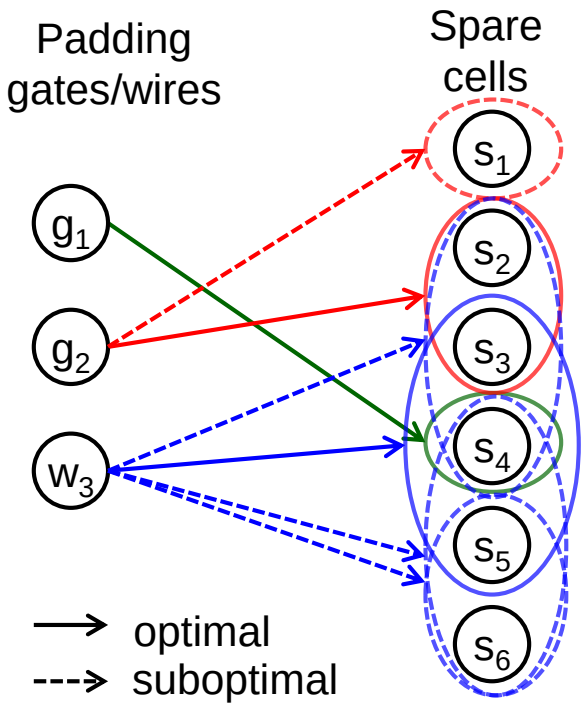


		Padding value					
		0.00	0.05	0.10	0.15	0.20	0.25
Spare cells	∅	0.00 ∅	0.00 ∅	0.00 ∅	0.00 ∅	0.00 ∅	0.00 ∅
	$\{s_1\}$	0.00 ∅	0.00 ∅	0.00 ∅	0.00 ∅	0.20 $\{s_1\}$	0.20 $\{s_1\}$
	$\{s_1, s_2\}$	0.00 ∅	0.0 ∅	0.00 ∅	0.15 $\{s_2\}$	0.20 $\{s_1\}$	0.20 $\{s_1\}$
	$\{s_1, s_2, s_3\}$	0.00 ∅	0.0 ∅	0.10 $\{s_3\}$	0.15 $\{s_2\}$	0.20 $\{s_1\}$	0.25 $\{s_2, s_3\}$

Spare Cell Selection (2/2)

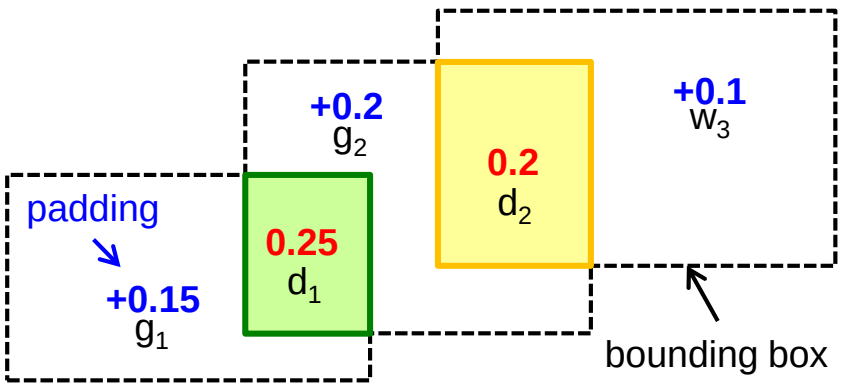
- **Solve the competition problem between different padding gates/wires**
  - Record multiple subset sum solutions
  - Sort the gates/wires in ascending order of the number of spare cell solutions
  - Assign the gates/wires to their best solution in the sorted order

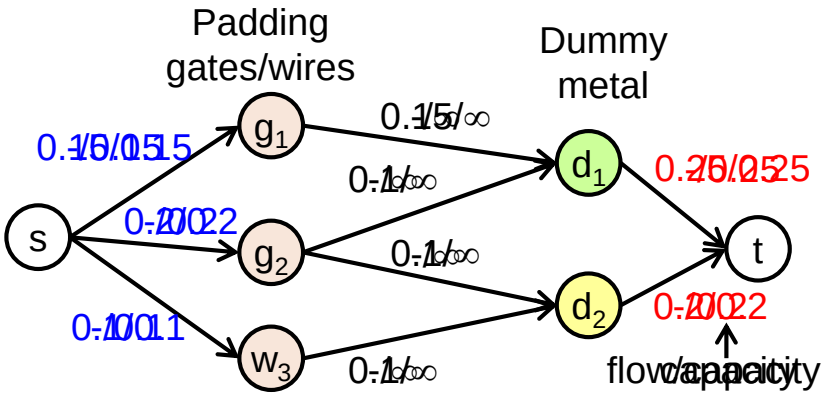




Dummy Metal Allocation

- **Fix remaining padding by dummy metal insertion**
  - Convert the delay to an amount of capacitance
  - Assign un-overlapped metal resource
  - Resort the unfixed padding to maximum network flow





Outline

- Introduction
- Problem Formulation
- Algorithm - PushPull
- **Experimental Results**
- **Conclusion**

Experimental Results

- **Platform: Intel® Xeon® CPU E5620 @ 2.40GHz with 16GB memory, CentOS 5.5**
- **Compare with two greedy heuristics**

Circuit	Ours: PushPull					Greedy 1: Largest setup first [2]					Greedy 2: Most path sharing first [1][2][3]				
	TNS_1 (ps)	THS_1 (ps)	Padding delay (ps)	#Ite.	Runtime (s)	TNS_1 (ps)	THS_1 (ps)	Padding delay (ps)	#Ite.	Runtime (s)	TNS_1 (ps)	THS_1 (ps)	Padding delay (ps)	#Ite.	Runtime (s)
s1196	0.0	0.0	152.5	1	0.02	0.0	0.0	338.4	5	0.02	0.0	0.0	173.8	3	0.01
s1423	0.0	0.0	4,746.9	2	0.05	0.0	43.5	4,968.7	56	0.45	0.0	43.5	4,802.8	35	0.24
s5378	0.0	0.0	3,722.7	2	0.08	0.0	79.3	4,678.1	60	0.79	0.0	79.3	4,555.4	43	0.58
s9234	0.0	0.0	1,647.5	1	0.05	0.0	12.0	1,878.3	33	0.30	0.0	12.0	2,158.5	27	0.26
s13207	0.0	0.0	371.2	2	0.07	0.0	34.5	762.3	22	0.21	0.0	34.5	621.5	18	0.21
s15850	0.0	0.0	1,510.8	2	0.04	0.0	0.0	1,662.8	43	0.25	0.0	0.0	2,161.5	41	0.27
s38584	0.0	0.0	143,764.1	2	1.04	0.0	0.0	225,026.0	780	115.19	0.0	85.8	130,703.0	512	80.37
des_perf	0.0	0.0	583,579.0	2	9.46	0.0	19,270.5	795,022.0	3,414	6,401.86	0.0	3,864.1	595,088.0	2,257	4,202.46
b19	0.0	0.0	1,675,688.0	4	21.68	0.0	108,078.0	6,128,710.0	21,902	37,473.30	0.0	65,746.7	1,729,230.0	6,220	10,978.40

TNS_1: total negative setup slack after padding value determination.
THS_1: total negative hold slack after padding value determination.

[1] S. Yoshikawa, “Hold time error correction method and correction program for integrated circuits,” US Patent 6,990,646, 2006.
[2] Y. Sun et al., “Method and apparatus for fixing hold time violations in a circuit design,” US Patent 7,278,126, 2007.
[3] Y. Liu et al., “Re-synthesis for cost-efficient circuit-level timing speculation,” DAC, pp. 158–163, 2011.

Experimental Results (cont'd)

- **Compare with LP solutions**

Circuit	#Gate	#FF	#SFF	Conservative clock period (ns)	THS (ps)	LP				Ours: PushPull					
						TNS_1 (ps)	THS_1 (ps)	Padding delay (ps)	Runtime (s)	TNS_1 (ps)	THS_1 (ps)	Padding delay (ps)	Normalized to LP	#Ite.	Runtime (s)
s1196	301	19	1	1.0	152.5	0.0	0.0	152.5	0.03	0.0	0.0	152.5	1.00	1	0.02
s1423	486	74	45	1.0	4,916.9	0.0	0.0	4,126.8	0.15	0.0	0.0	4,746.9	1.15	2	0.05
s5378	739	162	37	1.0	3,852.8	0.0	0.0	3,661.3	0.06	0.0	0.0	3,722.7	1.02	2	0.08
s9234	555	132	24	1.0	1,929.0	0.0	0.0	1,459.6	0.05	0.0	0.0	1,647.5	1.13	1	0.05
s13207	748	213	14	1.0	371.2	0.0	0.0	371.2	0.03	0.0	0.0	371.2	1.00	2	0.07
s15850	428	128	29	1.0	2,114.6	0.0	0.0	1,305.1	0.03	0.0	0.0	1,510.8	1.16	2	0.04
s38584	7,890	1,159	194	3.4	108,759.0	0.0	0.0	108,476.0	6.96	0.0	0.0	143,764.1	1.33	2	1.04
des_perf	51,349	8,808	1,190	2.9	583,579.0	0.0	0.0	583,579.0	60.59	0.0	0.0	583,579.0	1.00	2	9.46
b19	72,872	5,541	2,737	3.8	1,481,800.0	NA	NA	NA	timeout	0.0	0.0	1,675,688.0	-	4	21.68

TNS_1: total negative setup slack after padding value determination.
THS_1: total negative hold slack after padding value determination.

- **The impact of input slew on padding delay capacitance conversion**
  - The average error rate over all cases is only 0.56%.

LP: N. V. Shenoy et al., "Minimum padding to satisfy short path constraints," ICCAD-93.

Experimental Results (cont'd)

- **Compare our load/buffer allocation method with LP+Mapping**

Circuit	LP+Mapping			Ours: PushPull		
	TNS_2 (ps)	THS_2 (ps)	Runtime (s)	TNS_2 (ps)	THS_2 (ps)	Runtime (s)
s1196	0.0	0.3	0.03	0.0	0.0	0.33
s1423	0.0	826.4	0.09	0.0	0.0	0.43
s5378	0.0	302.3	0.05	0.0	0.0	0.54
s9234	0.0	631.3	0.04	0.0	0.0	0.41
s13207	0.0	161.2	0.03	0.0	0.0	0.61
s15850	0.0	352.0	0.02	0.0	0.0	0.38
s38584	0.0	29,970.7	0.58	0.0	0.0	1.96
des_perf	0.0	51,146.0	41.99	0.0	0.0	12.36
b19	NA	NA	NA	0.0	0.0	43.33

TNS_2: total negative setup slack after load/buffer allocation.
THS_2: total negative hold slack after load/buffer allocation.

LP+Mapping: N. V. Shenoy et al., "Minimum padding to satisfy short path constraints," ICCAD-93.

Conclusion

- **Focus on the severe short path padding problem in resilient circuits**
  - Enable the timing error detection and correction mechanism of resilient circuits
- **Determine the padding values and locations with a global view**
  - vs. the local view adopted by resent prior work
- **Realize the determined padding values at physical implementation**
  - Propose coarse-grained and fine-grained load/buffer allocation by using spare cells and dummy metal

Future Work

- **Allocate spare cell and dummy metal simultaneously**
  - Spare cells: discrete capacitance resource
  - Dummy metal: continuous capacitance resource
- **Adopt Composite Current Source (CCS) timing model**
  - Use more accuracy delay model

Thank You!

Contact info:
Yu-Ming Yang
yuming.yyang@gmail.com

Backup Slides

The Resilient Circuit Design Flow