Retiming and Re-synthesis

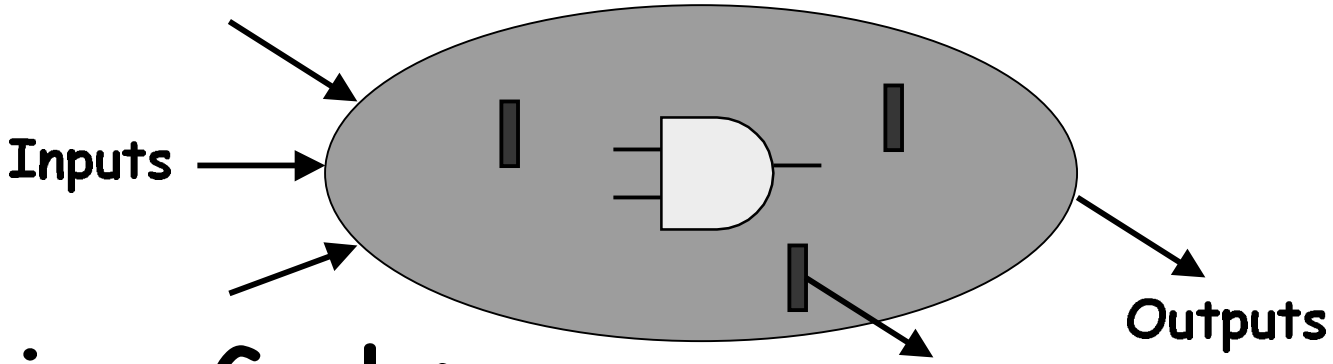
R. K. Brayton
University of California
Berkeley

Outline

- Retiming
- Retiming and Resynthesis (RnR)
- Resynthesis of Pipelines

Optimizing Sequential Circuits by Retiming Netlist of Gates

- Netlist of gates and registers:



- Various Goals:
  - Reduce clock cycle time
  - Reduce area
  - Reduce number of lateches

Retiming

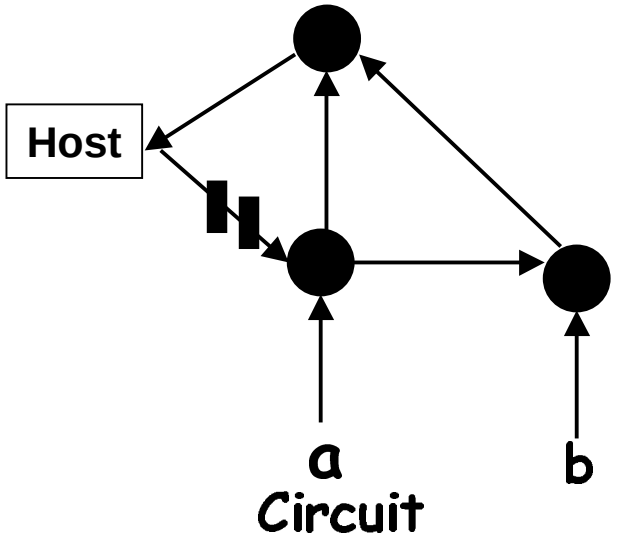
- Problem
  - Pure combinational optimization can be myopic since relations across register boundaries are disregarded
- Solutions
  - Retiming: Move register so that
    - clock cycle decreases, or number of registers decreases and
    - input-output behavior is preserved
  - RnR: Combine retiming with combinational optimization techniques so that can optimize across boundaries

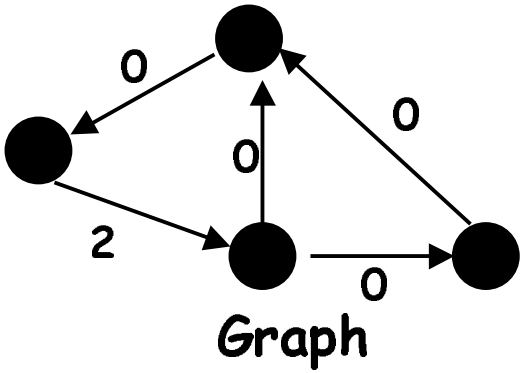
Circuit Represetation

- Leiserson, Rose and Saxe (1983)
- Circuit representation: G(V,E,d,w)
  - V $\leftrightarrow$ set of gates
  - E $\leftrightarrow$ set of wires
  - d(v) = delay of gate/vertex v, ($d(v) \geq 0$)
  - w(e) = number of registers on edge e, ($w(e) \geq 0$)

Circuit Representation

Example: Correlator





Operation	delay
δ	3
+	7

$\delta(x, y) = 1$ if $x=y$
0 otherwise

- Every cycle in Graph has at least one register i.e. no combinational loops.

preliminaries

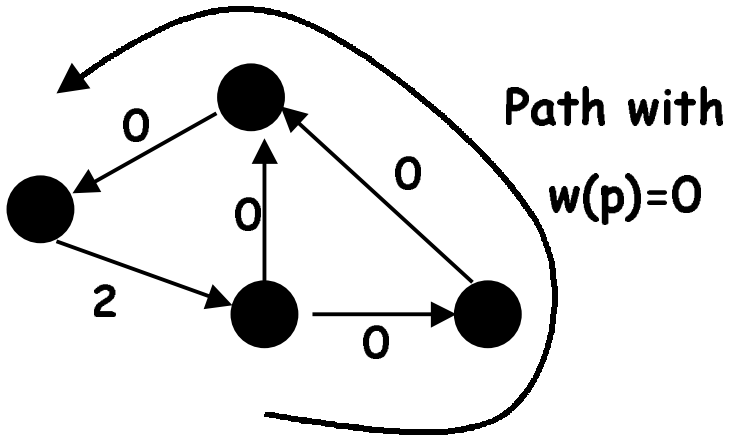
- For a path p: $V_0 \rightarrow$

$$d(p) = \sum_{i=0}^{k} d(v_i) \quad \text{(includes endpoints)}$$

$$w(p) = \sum_{i=0}^{k-1} w(e_i)$$

- Clock cycle

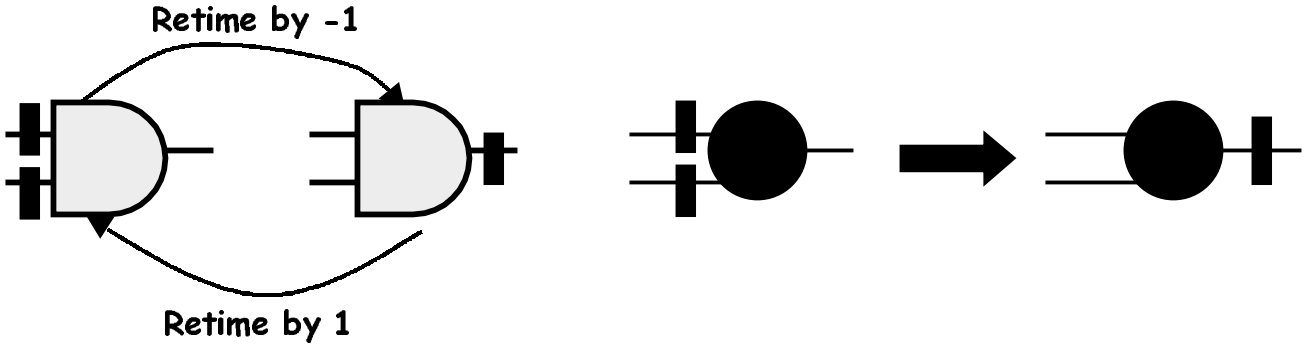
$$c = \max_{p:w(p)=0} \{d(p)\}$$



For correlator c = 13

Basic Operation

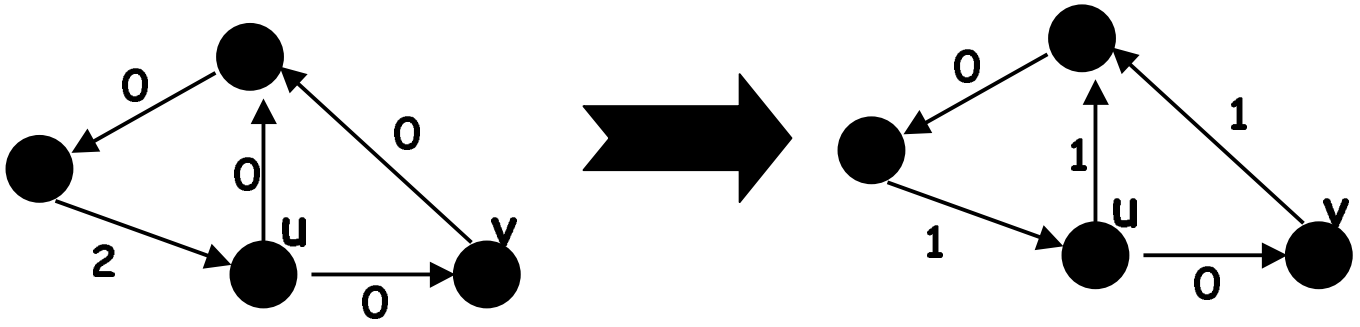
- Movement of registers from input to output of a gate or vice versa



- Does not affect gate functionalities
- A mathematical definition: Retardation
  - r: V $\rightarrow$ Z, an integer vertex labeling
  - Wr(e) =w(e) + r(v) - r(u) for edge e= (u,v)

Basic Operation

- Thus in the example, r(u) = -1, r(v) = -1 results in



- For a path p: $s \rightarrow t$, $W_r(p) = w(p) + r(t) - r(s)$
- A mathematical definition: Retardation
  - r: $V \rightarrow Z$, an integer vertex labeling
  - $W_r(e) = w(e) + r(v) - r(u)$ for edge e= (u,v)
  - A retiming r is legal if $W_r(e) \geq 0$, $\forall_e \in E$

Retiming for minimum clock cycle

- Problem Statement: (Minimum cycle time)
- Given G(V, E, d, w), find a Legal retiming r so that

$$c = \max_{p: w_r(p)=0} \{d(p)\}$$

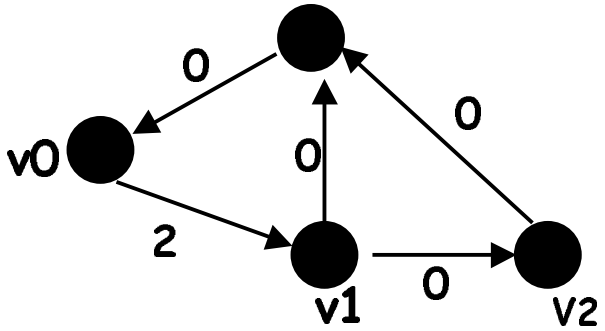
  is minimized
- Retiming: 2 important matrices
  - Register weight matrix

$$W(u,v) = \min\{w(p) : u \xrightarrow{p} v\}$$

  - Delay matrix

$$D(u,v) = \max\{d(p) : u \xrightarrow{p} v, w(p) = W(u,v)\}$$

Retiming for minimum clock cycle



W

	V0	V1	V2	V3
V0	0	2	2	2
V1	0	0	0	0
V2	0	2	0	0
V3	0	2	2	0

D

	V0	V1	V2	V3
V0	0	3	6	13
V1	13	3	6	13
V2	10	13	3	10
V3	7	10	13	7

$C \leq \alpha \Leftrightarrow \forall p$, if $d(p) > \alpha$ then $w(p) \geq 1$

Conditions for Retiming

- Assume that we are asked to check if a retiming exists for a clock cycle α
- Legal retiming: $w_r(e) \geq 0$ for all e. Hence
  $w_r(e) = w(e) = r(v) - r(u) \geq 0$ or
  $r(u) - r(v) \leq w(e)$
- For all paths p: u $\rightarrow$ v such that $d(p) \geq \alpha$, we require $w_r(p) \geq 1$
  - Thus

$$1 \leq w_r(p) = \sum_{i=0}^{k-1} w_r(e_i)$$

$$= \sum_{i=0}^{k-1} [w(e_i) + r(v_{i+1}) - r(v_i)]$$

$$= w(p) + r(v_k) - r(v_0)$$

$$= w(p) + r(v) - r(u)$$

Or take the least w(p) (tightest constraint) $r(u)-r(v) \leq W(u,v)-1$

Note: this is independent of the path from u to v, so we just need to apply it to u, v such that $D(u,v) > \alpha$

Solving the constraints

- All constraints in difference of 2 variable form
- Related to shortest path problem

  Correlator: α = 7

W

	V0	V1	V2	V3
V0	0	2	2	2
V1	0	0	0	0
V2	0	2	0	0
V3	0	2	2	0

D

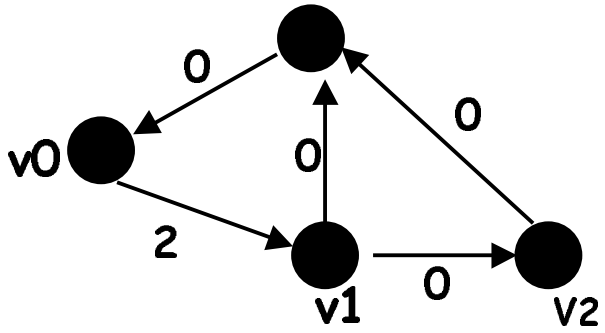
	V0	V1	V2	V3
V0	0	3	6	13
V1	13	3	6	13
V2	10	13	3	10
V3	7	10	13	7

Legal: $r(u)-r(v) \le w(e)$

$r(v_0)-r(v_1) \le 2$
$r(v_1)-r(v_2) \le 0$
$r(v_1)-r(v_3) \le 0$
$r(v_2)-r(v_3) \le 0$
$r(v_3)-r(v_0) \le 0$

D>7: $r(u)-r(v) \le W(u,v)-1$

$r(v_0)-r(v_3) \le 1$
$r(v_1)-r(v_0) \le -1$
$r(v_1)-r(v_3) \le -1$
$r(v_2)-r(v_0) \le -1$
$r(v_2)-r(v_1) \le 1$
$r(v_2)-r(v_3) \le -1$
$r(v_3)-r(v_1) \le 1$
$r(v_3)-r(v_2) \le 1$



Solving the constraints

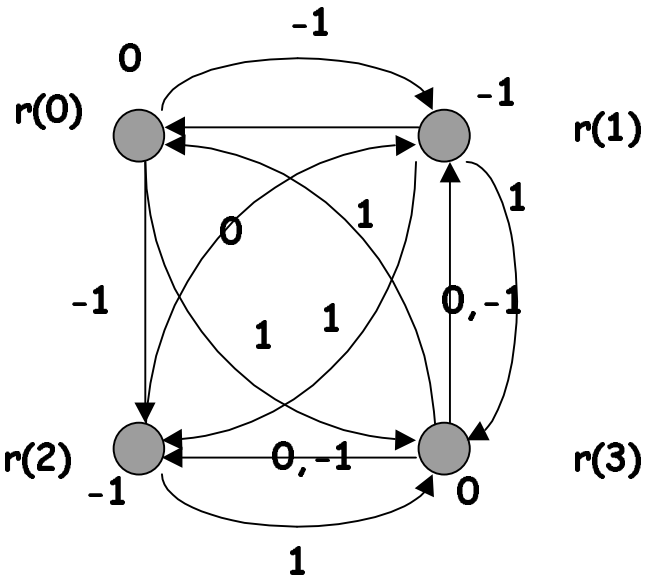
- Do shortest path on constraint graph: ($O(|V|^3)$).
- A solution exists if and only if there exists no negative weighted cycle.

Legal: $r(u)-r(v) \leq w(e)$

$r(v_0)-r(v_1) \leq 2$
$r(v_1)-r(v_2) \leq 0$
$r(v_1)-r(v_3) \leq 0$
$r(v_2)-r(v_3) \leq 0$
$r(v_3)-r(v_0) \leq 0$

D>7:
$r(u)-r(v) \leq W(u,v)-1$

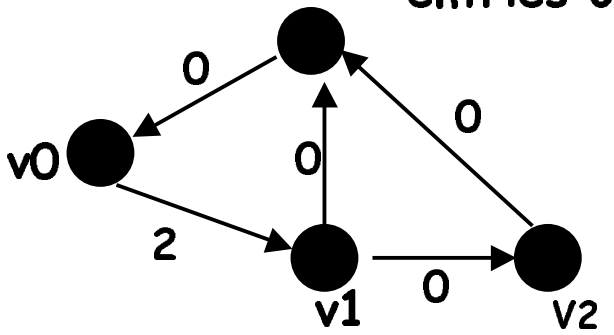
$r(v_0)-r(v_1) \leq 1$
$r(v_1)-r(v_0) \leq -1$
$r(v_1)-r(v_3) \leq -1$
$r(v_2)-r(v_0) \leq -1$
$r(v_2)-r(v_1) \leq 1$
$r(v_2)-r(v_3) \leq -1$
$r(v_3)-r(v_1) \leq 1$
$r(v_3)-r(v_2) \leq 1$



A solution is $r(v_0) = r(v_3) = 0$, $r(v_1) = r(v_2) = -1$

Retiming

To find the minimum cycle time, do a binary search among the entries of the D matrix $O(|V|^3 \log |V|)$



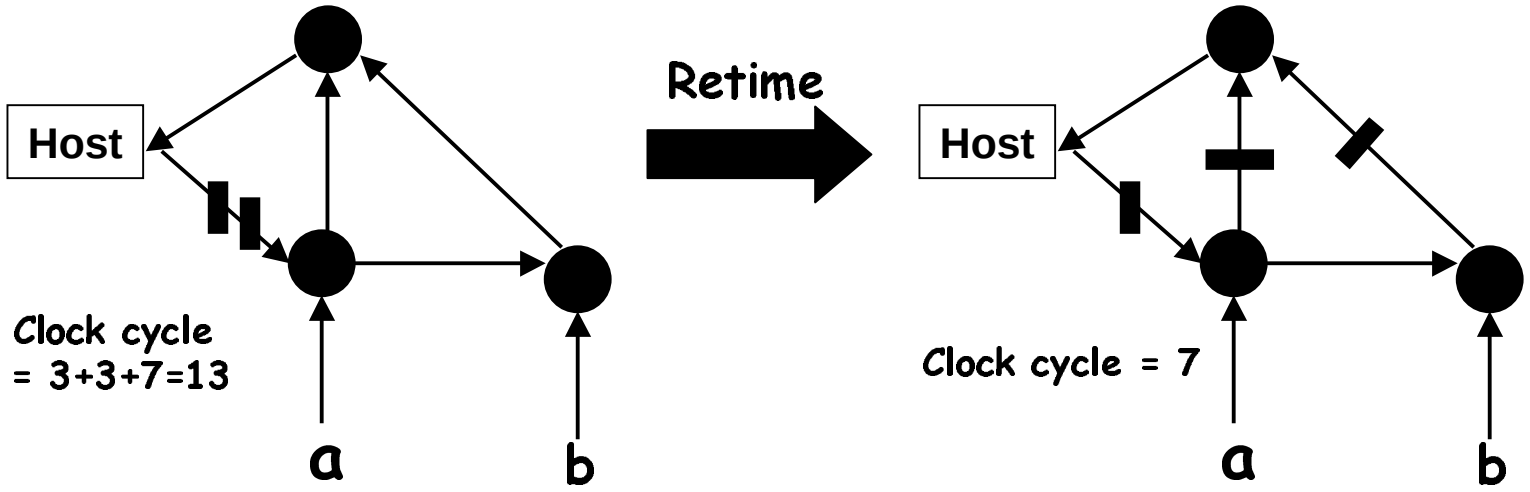
W

	V0	V1	V2	V3	
V0	0	2	2	2	V0
V1	0	0	0	0	V1
V2	0	2	0	0	V2
V3	0	2	2	0	V3

D

	V0	V1	V2	V3	
V0	0	3	6	13	V0
V1	13	3	6	13	V1
V2	10	13	3	10	V2
V3	7	10	13	7	V3

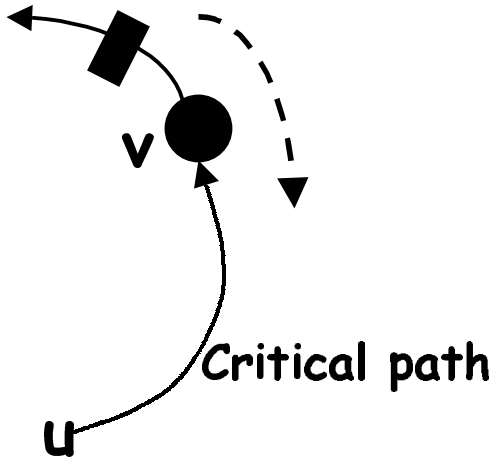
Retimed correlator:



Retiming: 2 more algorithms

- Relaxation based: Repeatedly find critical path; retime vertex at end of of path by +1

$$O(|V||E|\log|V|)$$



- Mixed Integer Linear Program formulation

Retiming for minimum state

Goal: minimize number of registers used

$$
\begin{aligned}
N_r &= \sum_{e \in E} w_r(e) \\
&= \sum_{e:u \to v} (w(e) + r(v) - r(u)) \\
&= \sum_{e \in E} w(e) + \sum_{e:u \to v} (r(v) - r(u)) \\
&= N + \sum_{u \to v} (r(v) - r(u)) \\
&= N + \sum_{v \in V} (r(v)(\#\, fanin(v) - \#\, fanout(v)) \\
&= N + \sum_{v \in V} a_v r(v)
\end{aligned}
$$

Where a_v is a constant.

Minimum registers - formulation

- Minimize:

$$\sum_{v \in V} a_v r(v)$$

Subject to $W_r(e) = w(e) + r(v) - r(u) \geq 0$

- Reducible to a flow problem
- Care to be taken when registers fanout to registers
- Handle multi-way forks with care

Retiming and resynthesis: motivation

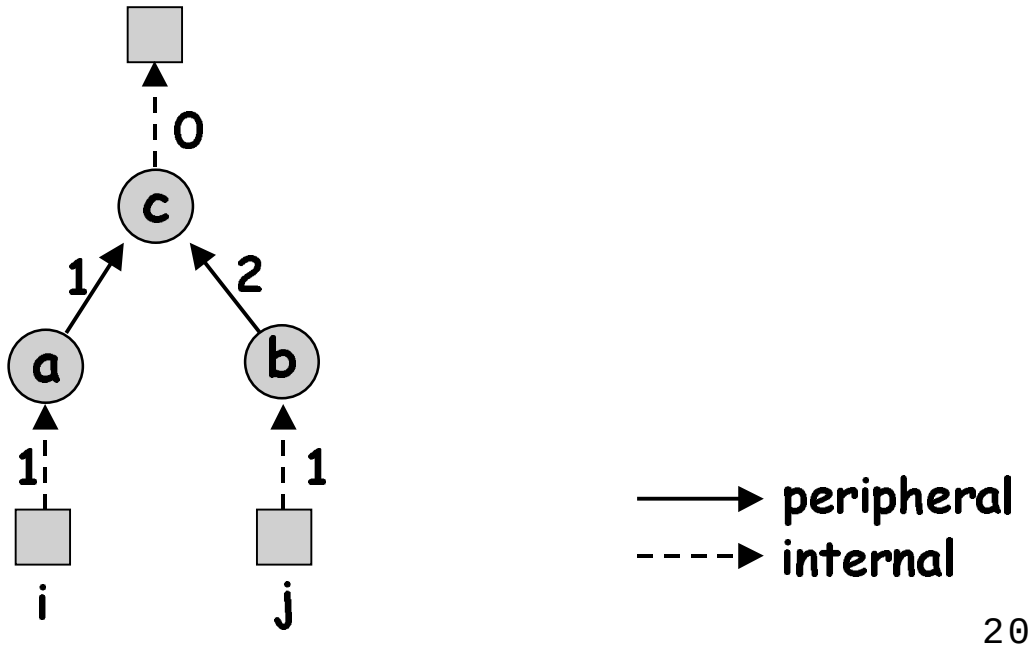
- Goal: incorporate cominational optimazation into sequential optimization
- Naïve approach: carve out combinational regions, do optimization on each region Only local gains made.
- Can we do any better?

RnR: a new approach

- Sentovich, Malik, Brayton andSangiovanni-Vincentelli (89)
- 3 steps approach
  - 1. Move registers to boundary of circuit
  - 2. Optimize network
  - 3. Move registers back in an optimal way

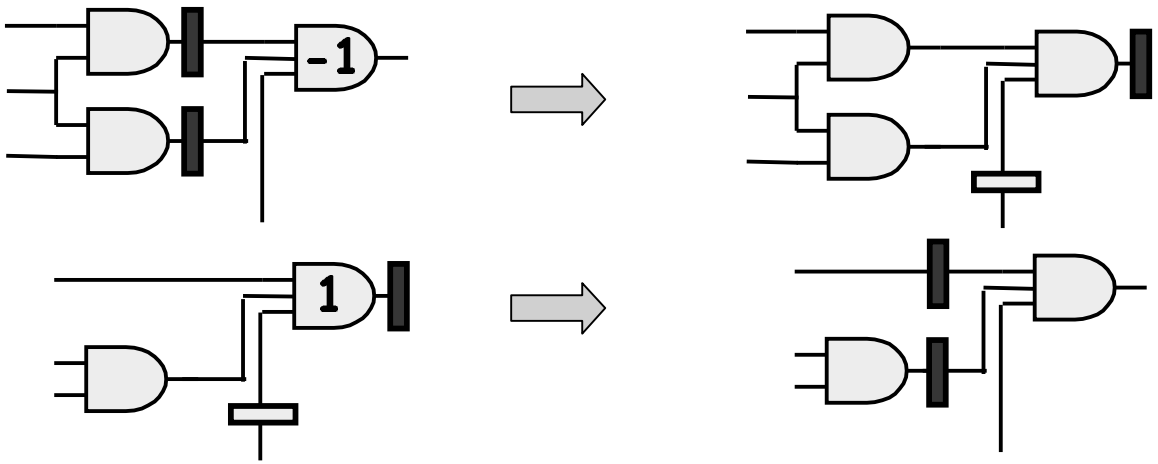
RnR: circuit representation

- Cirsuit representation: communication graph
  - internal/peripheral edges
  - edge-weight = register count



Extended Retiming

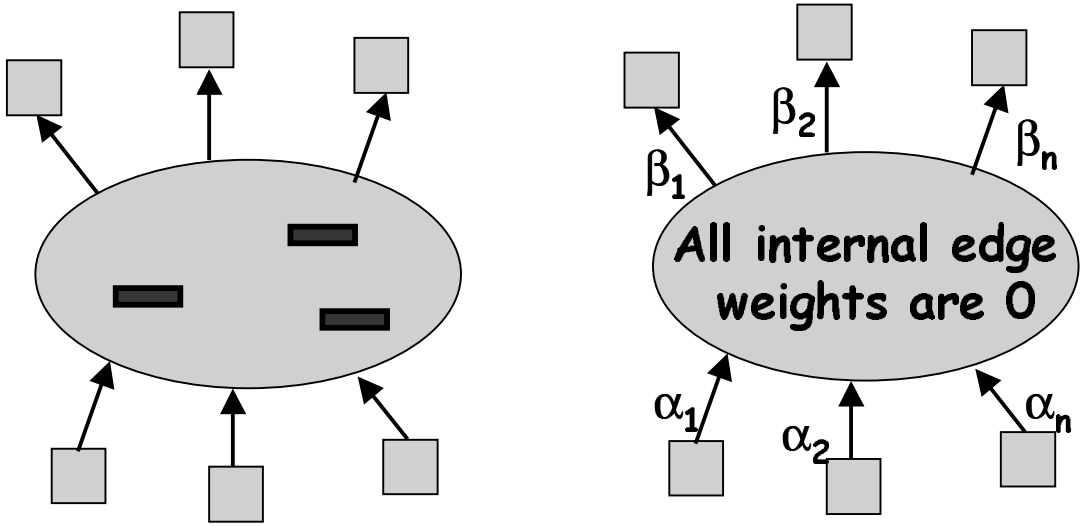
- Move register to the periphery
- Negative edge-weights permitted



- A negative latch has the interpretation that it advances its input by 1 clock cycle instead of delaying it by one clock cycle

Peripheral Retiming

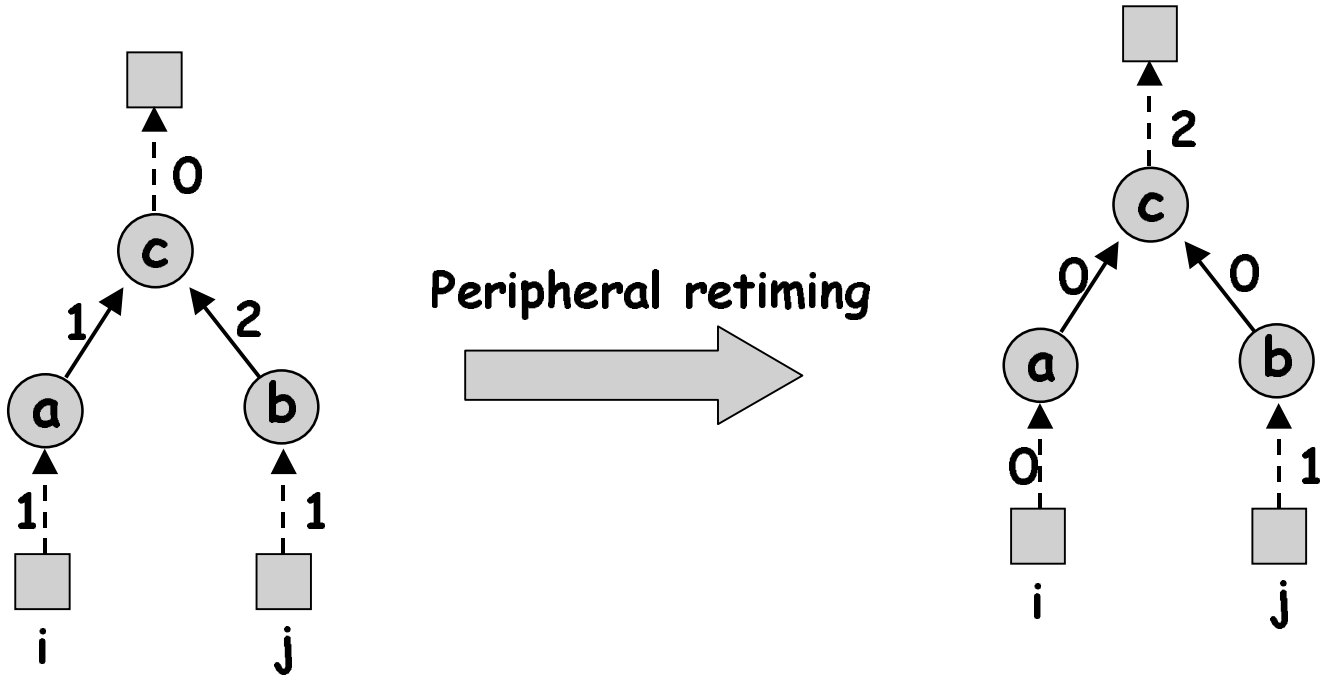
- A retiming is called a peripheral retiming if it results in all internal edges having zero weight
- Peripheral edges can have negative weight



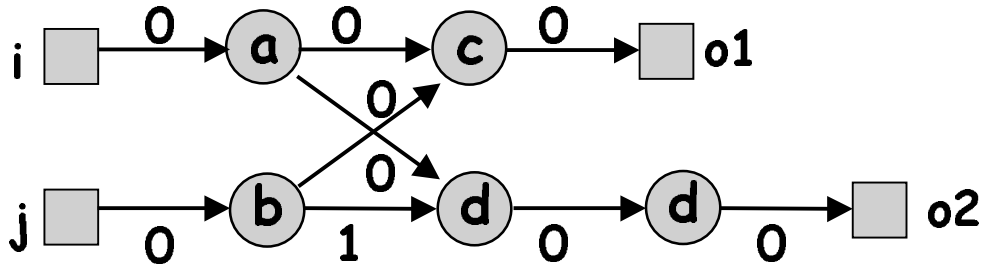
Peripheral Retiming

- A circuit that undergoes peripheral retiming followed by a legal retiming, i.e. one that results in all weights≥0 is functionally equivalent to the original circuit
- Functional equivalence: equivalence of finite state automata (but we have to be careful about the initial conditions and initializing sequences. The resulting circuit may only exhibit equivalence after an appropriate delay. See Singhal et.al ICCAD 1995. This holds even for regular retiming)

Peripheral Retiming - an example

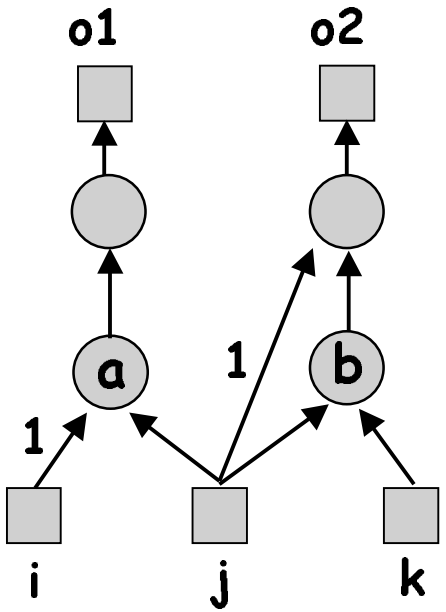


Not possible for all circuits:



Path Weight Matrix (PWM)

- Matrix W
  - row: inputs
  - column: outputs



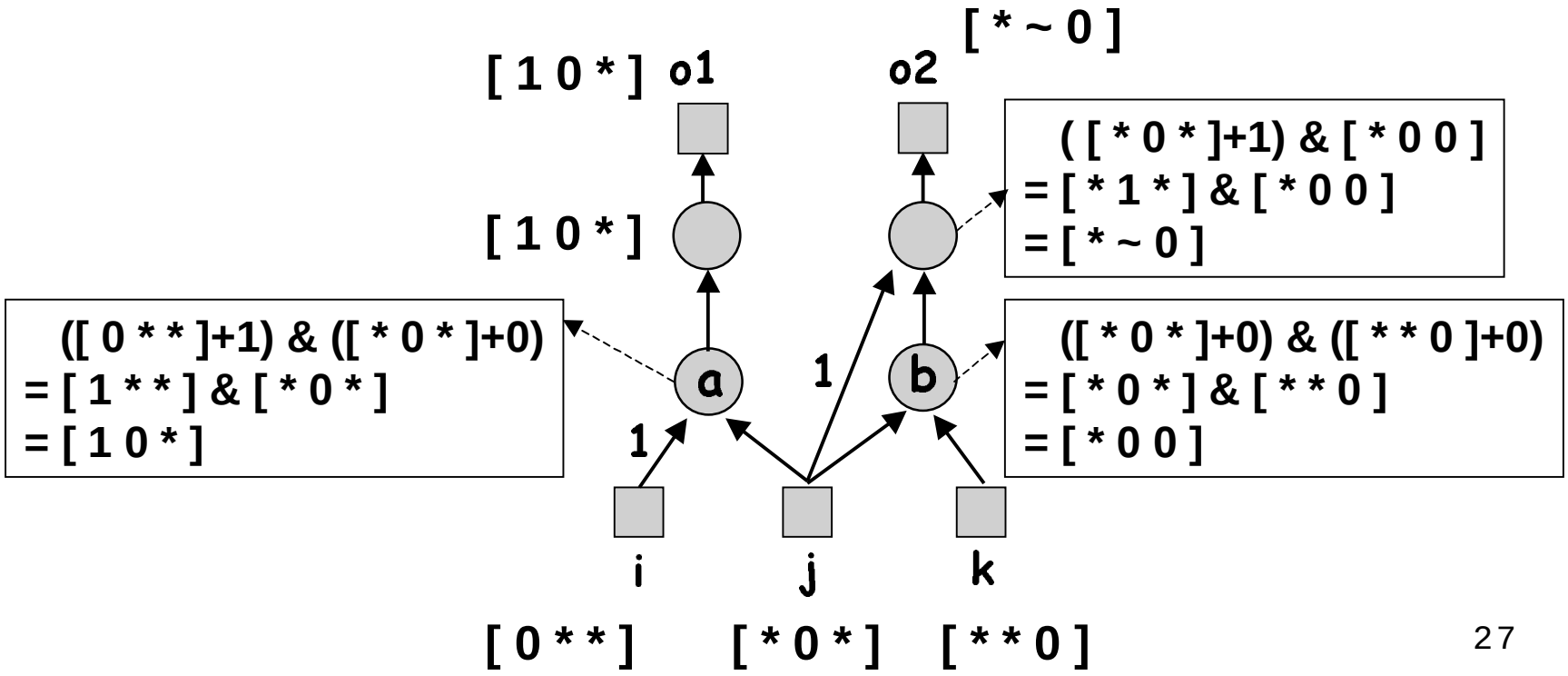
	o1	o2
I	1	*
j	0	~
k	*	0

PWM and Peripheral Retiming

- **Satisfiable path weight matrix:**
  - $W_{ij} \neq$ ~, $\forall i$, $\forall j$ and
  - $\exists \alpha_i, \beta_j$, such that $W_{ij} = \alpha_i + \beta_j$, $\forall W_{ij} \neq$ *
- **Peripheral retiming possible $\Leftrightarrow$ matrix is satisfiable**
  - α_i, β_j specify registers on peripheral edge
  - some α_i, β_j can be negative
- **Complexity: linear in size of communication graph**

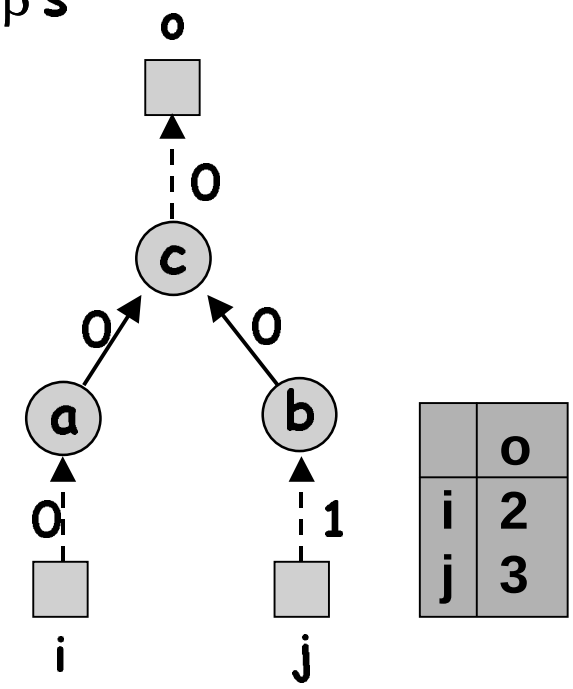
Path Weight Matrix - generation

- DFS skeleton
  - generate one output column of W at a time
- Example:



Computing α, β

- Each constraint of the form $\alpha_i + \beta_j = W_{ij}$
- Procedure
  - set $\alpha_1 = 0$
  - use first row to generate β's
  - determine α's from β's
  - check for consistency
- Example:

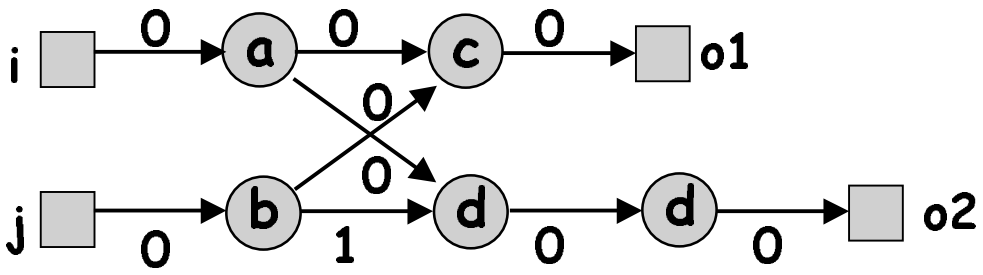


	o
i	2
j	3

$\alpha 1 = 0$
$\alpha 2 = 0$
$\beta 1 = 2$

Computing α, β

- **Example:**



	o1	o2
i	0	0
j	0	1

$\alpha 1 + \beta 1 = 0$ $\alpha 2 + \beta 1 = 0$

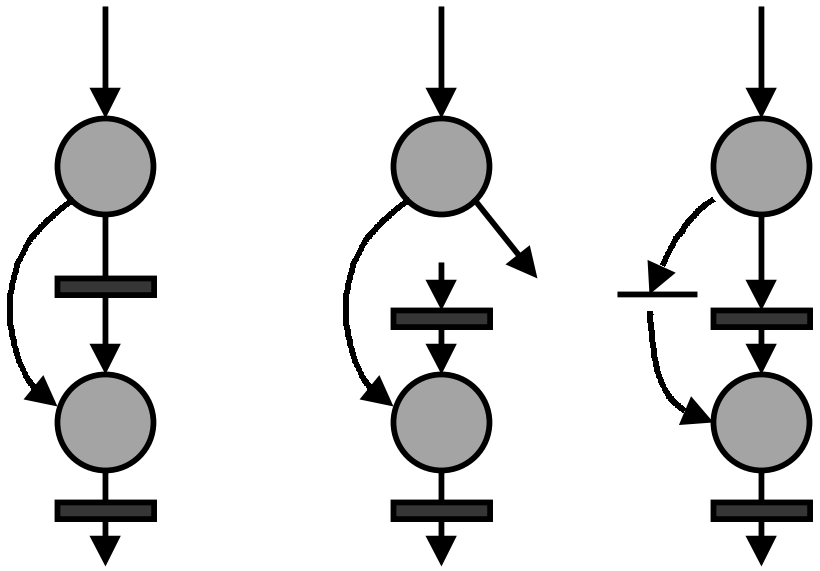
$\alpha 1 + \beta 2 = 0$ $\alpha 2 + \beta 2 = 1$

$\beta 1 - \beta 2 = 0$ $\beta 1 - \beta 2 = -1$

Optimizing Acyclic Sequential Circuits

- Acyclic circuits with satisfiable W
  - Do peripheral retiming putting α_i, β_j registers at the I/O
  - Resynthesize interior
  - Do a legal retiming (move registers in)
    - May not always be possible
- Acyclic circuits with unsatisfiable W
  - Identify maximal sub-circuits with satisfiable W
  - Cut connections
  - Repeat previous procedure

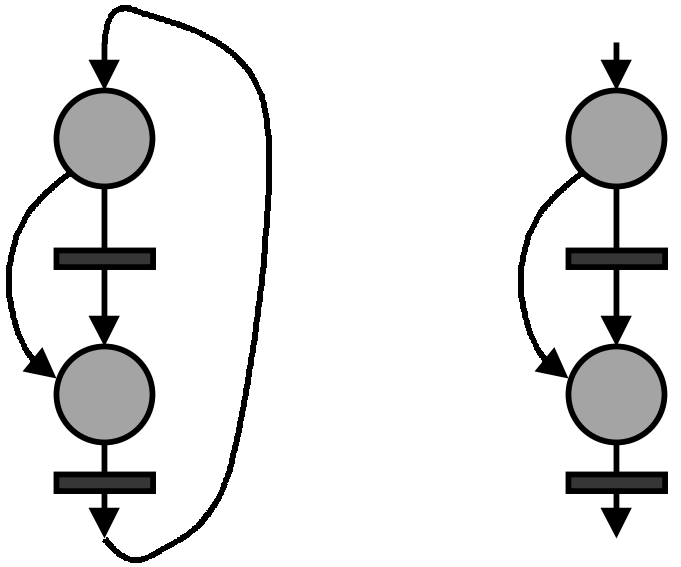
Acyclic Sequential Circuits



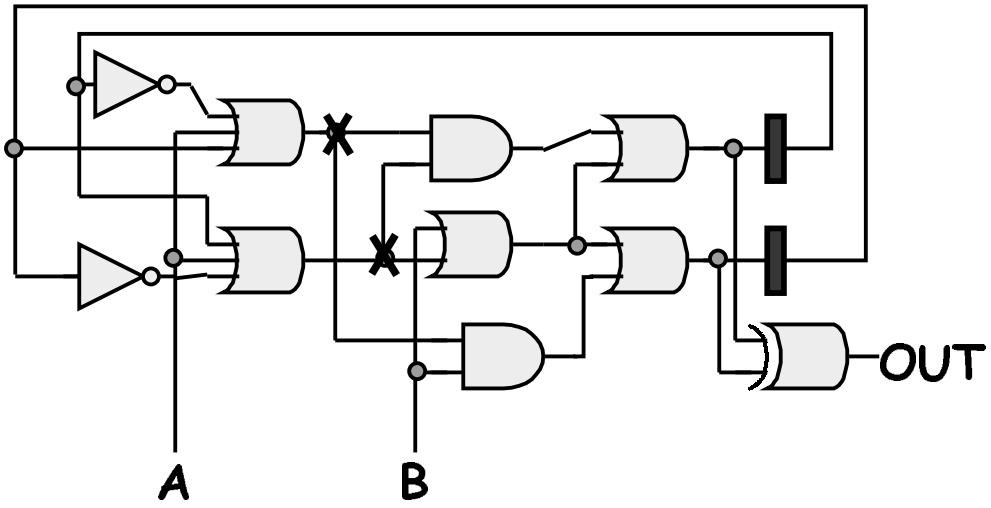
- Note that both of the cut circuits can be peripherally retimed, whereas the first one cannot

Optimizing Cyclic Sequential Circuits

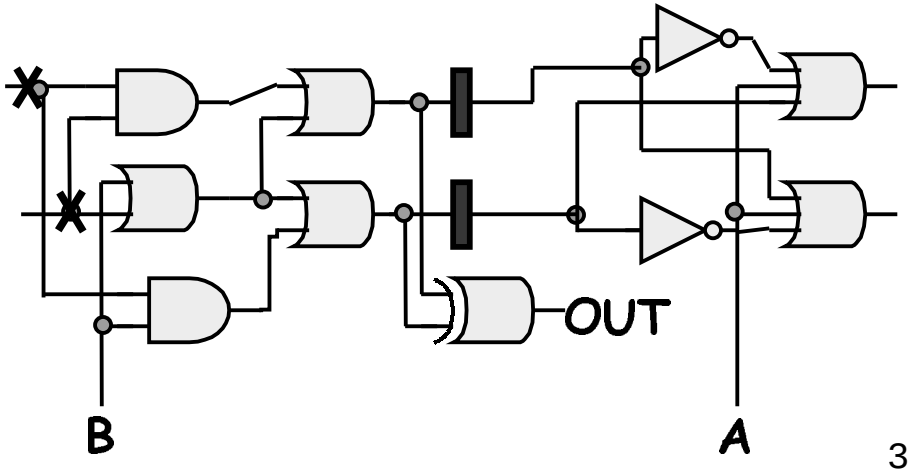
- Cyclic circuits
  - Make circuit acyclic by breaking cycles
  - Different feedback-cuts give different W's
  - Find sub-circuits with satisfiable W's
  - Repeat procedure (1)



FSM Optimization

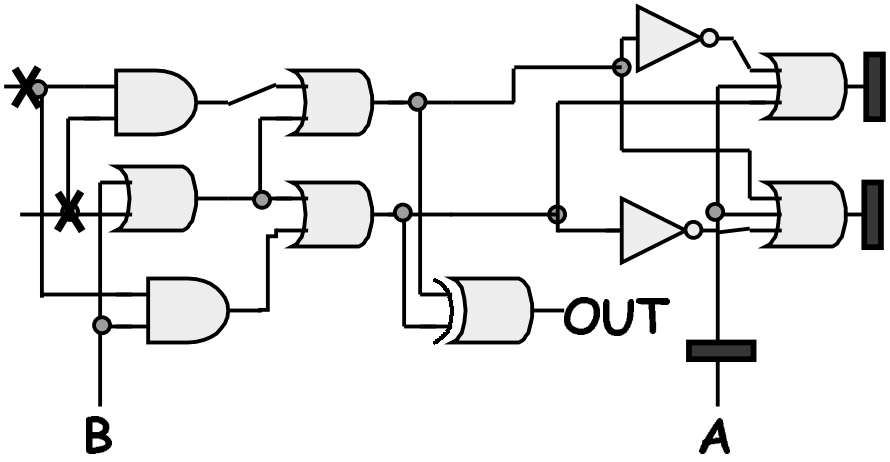


Break feedback

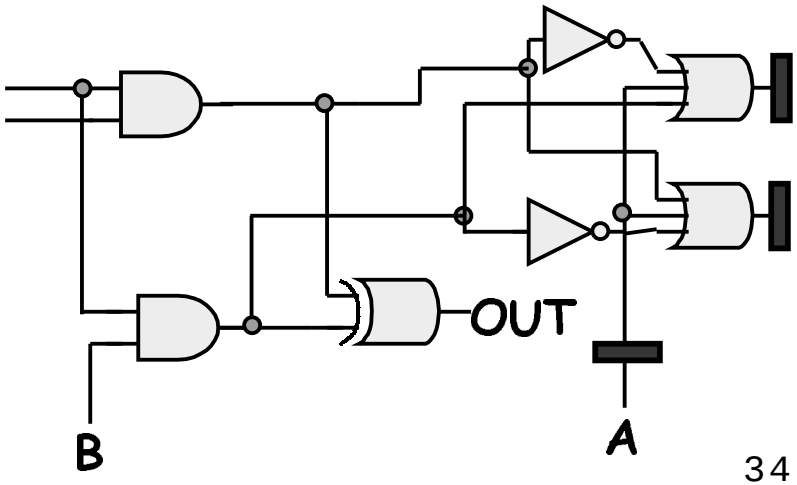


FSM Optimization

Peripheral retiming

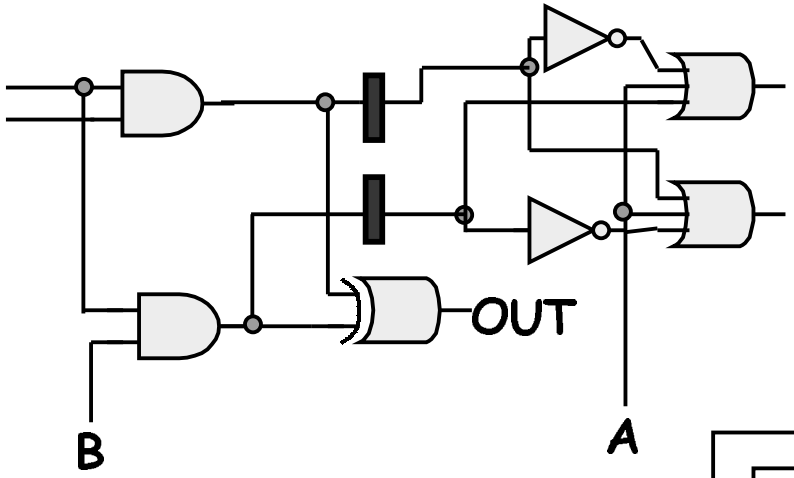


Resynthesize

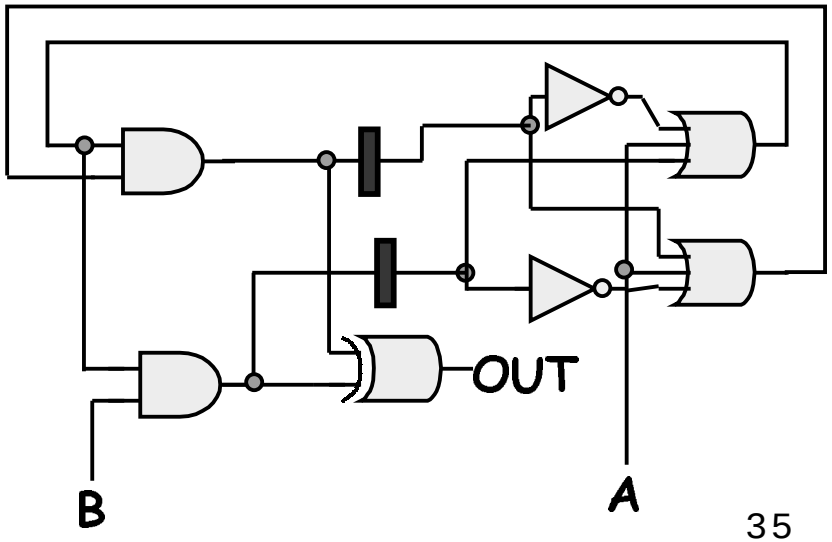


FSM Optimization

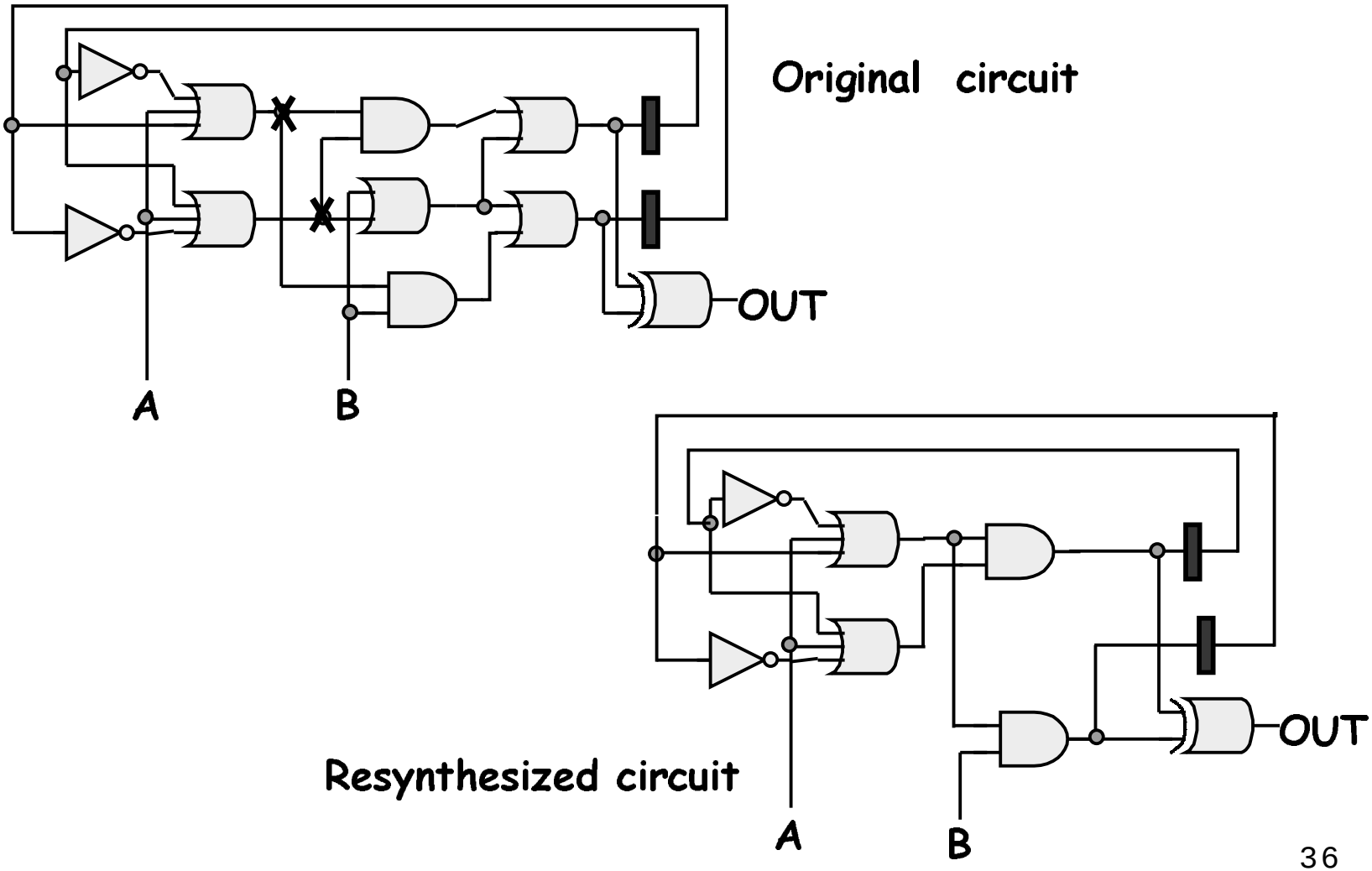
Retime



Reconnect

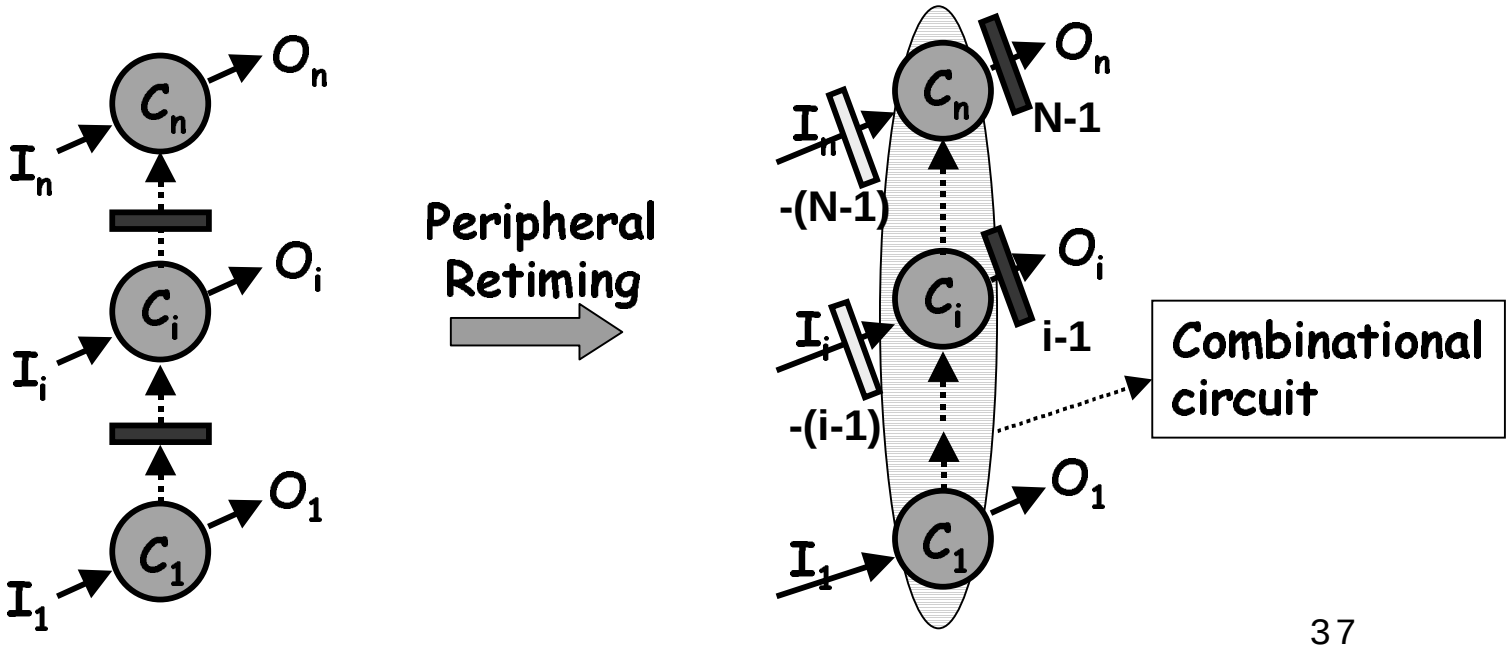


FSM Optimization



Resynthesis of Pipelines

- Goal: Performance optimization of pipeline circuits
- Example: Pipeline circuit C



Resynthesis of Pipelines

- Parameters:
  - A vector of arrival times for inputs
  - R vector of required times for outputs
  - c target clock cycle
  - C circuit
- Pipeline performance problem:

$$P_P(C_P, c_P, A_P, R_P)$$

- Combinational performance problem:

$$P_C(C_C, A_C, R_C)$$

Resynthesis of Pipelines

- **Problem Transformation:**
  - Relax $P_P(C_P, c_P, A_P, R_P)$ to $P'_P(C_P, c+\delta, A_P, R_P)$
    - where δ = largest possible single gate delay
  - Convert: pipeline P'_P to combinational problem P'_C

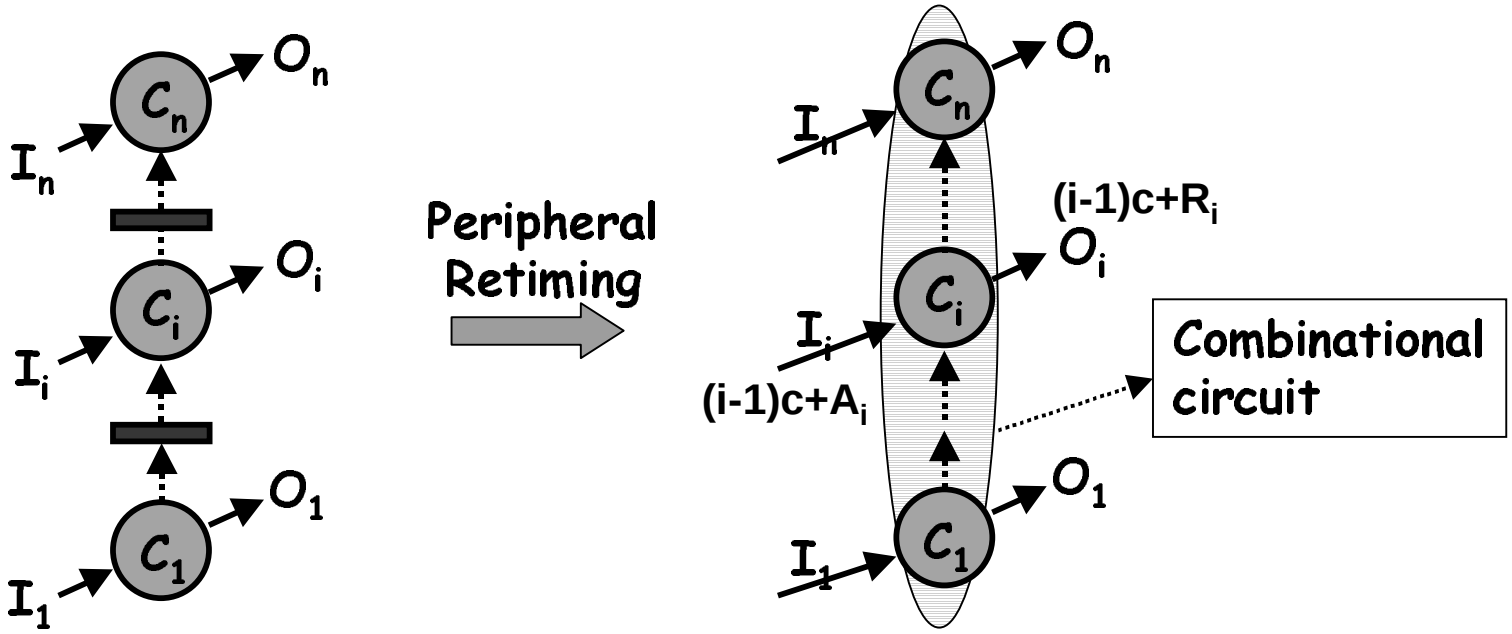
$$C_P \xrightarrow{\text{peripheral retime}} C_C$$

$$R_C^{stage_i} = (i-1) \times c + R_P^{stage_i}$$

$$A_C^{stage_i} = (i-1) \times c + A_P^{stage_i}$$

Perfomance Synthesis of Pipeline

- Arrival and Required Times for P'_C :



- Theoretical Contribution:
  - If P'_C has a solution then retiming yields a solution to P'_P
  - If there is a solution to P_P then peripheral retiming yields a solution to P'_C