CSE4210 Architecture and Hardware for DSP

Lecture 1

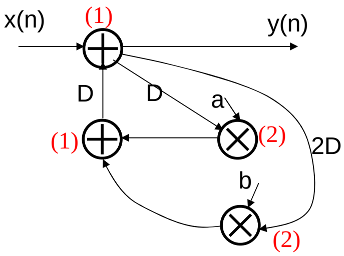
Retiming

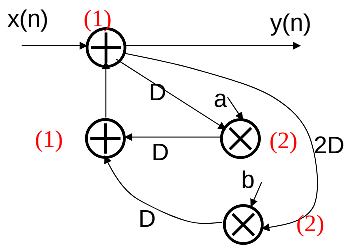
Introduction

- Retiming is a transformation technique that is used to change the locations of delay elements in a circuit without changing its functionality.
- Can be used to reduce the number of registers, or the clock cycle
- Could be considered as a generalization of the pipelining technique studies earlier

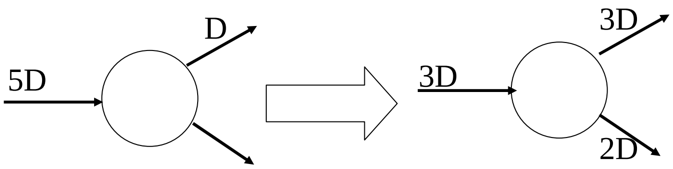
Retiming

$y(n)=ay(n-2) + by(n-3) +x (n)$

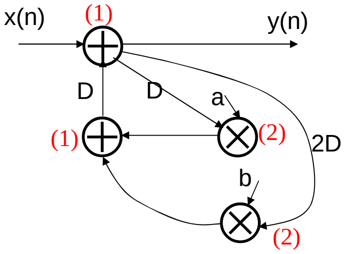


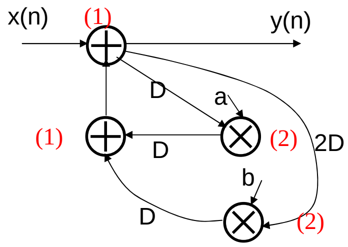


Introduction



$y(n)=ay(n-2) + by(n-3) +x (n)$





Definitions

- Mapping G to G_r
- A retiming solution is a value r(V) for every node in the graph.
- w(e) is the original weight of the edge e
- $w_r(e)$ is the weight in the retimed graph
- Edge e is from U → V

$$w_r(e)=w(e)+r(V)-r(U)$$

- A solution is feasible if all $w_r \geq 0$
- For previous example, r(1)=r(3)=r(4)=0, r(2)=-1

Definitions

- The weight of a path from a to b is

$$w_r(p)=w(p)+r(b)-r(a)$$

- Retiming does not change the number of delays in a cycle.
- Retiming does not change iteration bound
- Adding a constant to all the r(V) produce the same circuit

Solving Systems of Inequalities

- Draw the constraints graph
  - Draw node 1 to N from the graph + node N+1
  - For each inequality $r_i - r_j \le k$, (k is integer) draw an edge from node j → i with weight k
  - For each node i=1,2,..N draw an edge N+1 → i with weight 0
- Solve
  - The system has a solution if the constraints graph has no negative cycle.
  - One solution is the min. length from node N+1 to i

Example

1

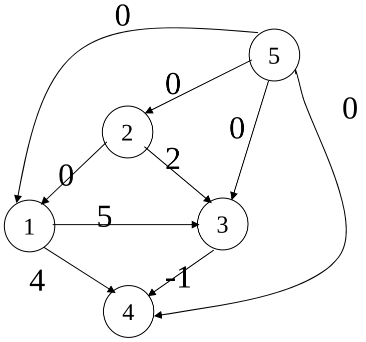
$r_1 - r_2 \le 0$

$r_3 - r_1 \le 5$

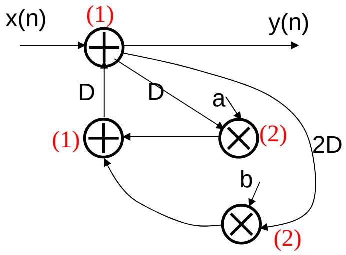
$r_4 - r_1 \le 4$

$r_4 - r_3 \le -1$

$r_3 - r_2 \le 2$



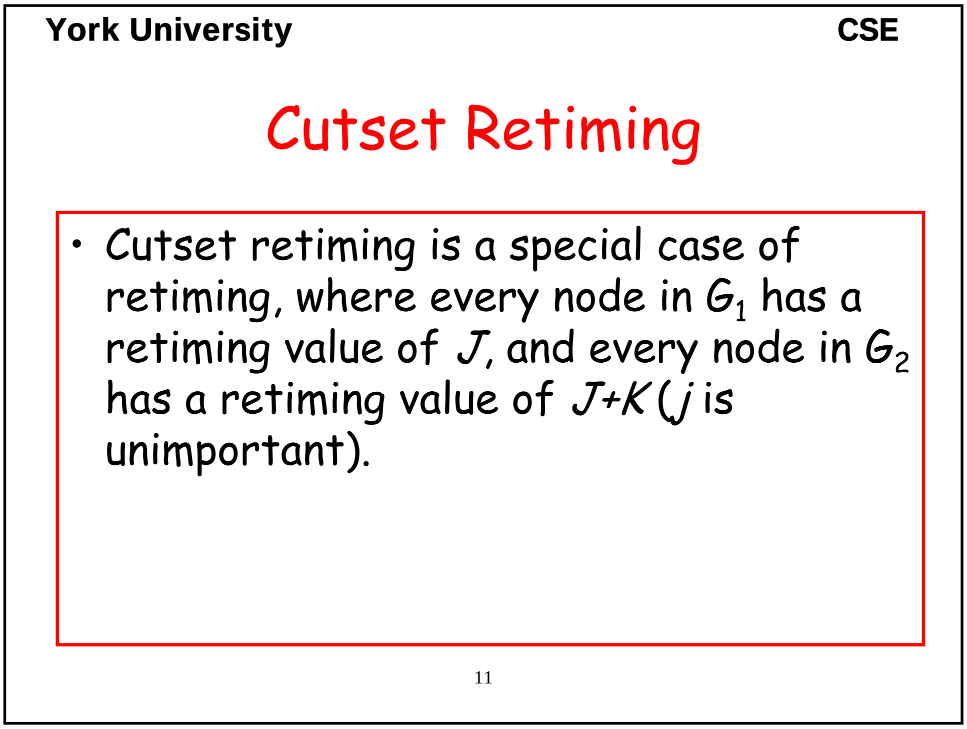
Example



Cutset Retiming

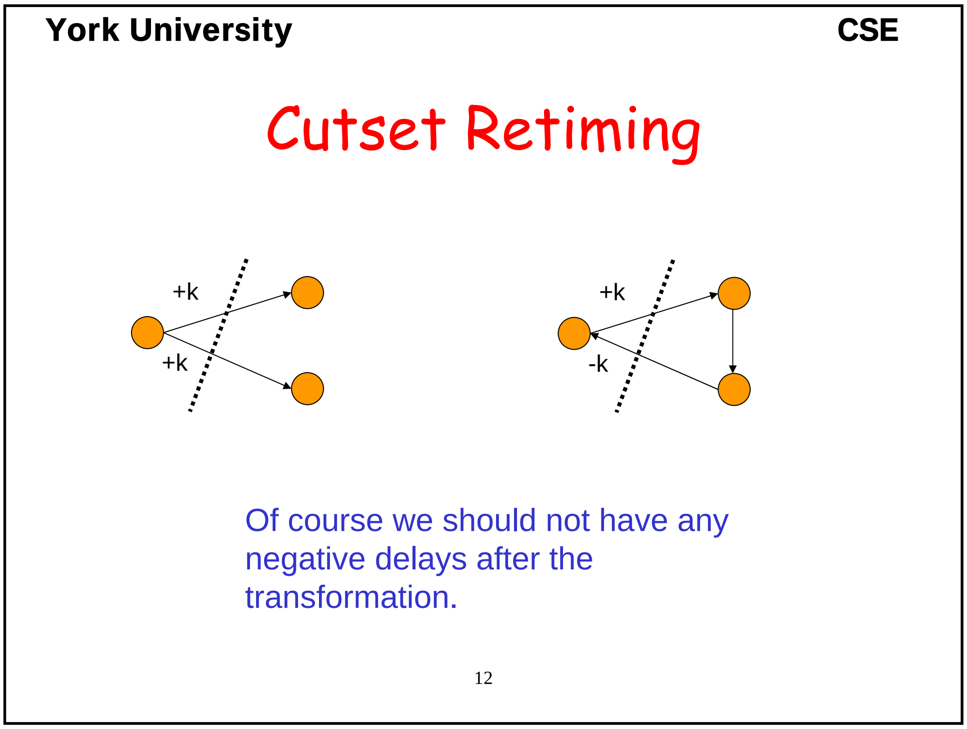
- Cutset: A set of edges if removed, the graph G is partitioned into 2 graphs G_1,G_2 .
- Cutset retiming is done by adding k delays to all the edges in the cutset from G_1 to G_2, and removing k delays from the edges from G_2 to G_1

$$-\min_{G_1 \xrightarrow{e} G_2} \{w(e)\} \le k \le \min_{G_2 \xrightarrow{e} G_1} \{w(e)\}$$



Cutset Retiming

- Cutset retiming is a special case of retiming, where every node in G_1 has a retiming value of J, and every node in G_2 has a retiming value of $J+K$ (j is unimportant).

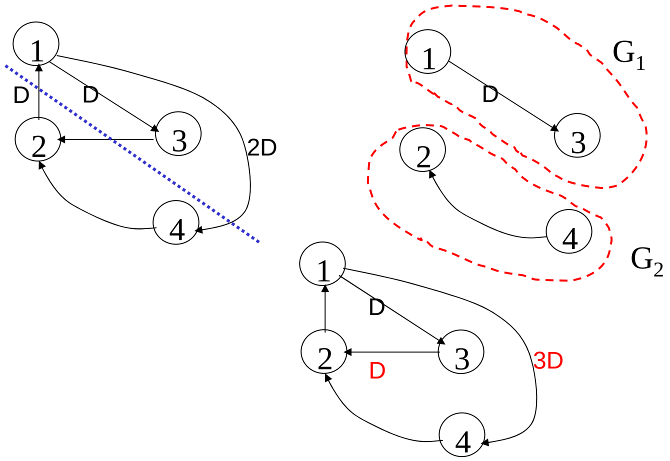


Cutset Retiming

Of course we should not have any negative delays after the transformation.

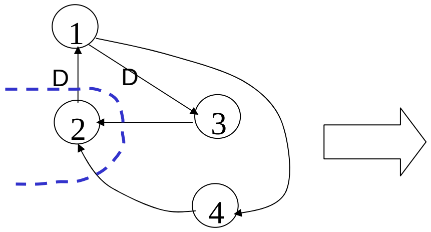
York University CSE

Cutset



York University CSE

Cutset

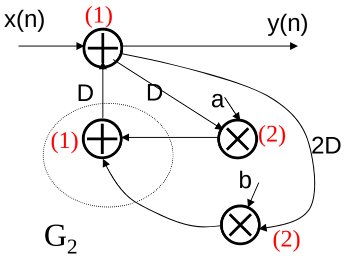


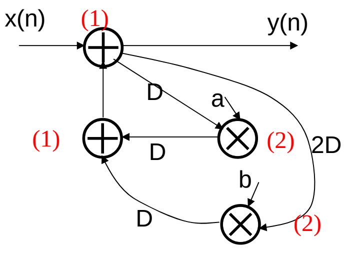
Cutset Retiming

- Pipelining is a special case where there are no nodes from G_2 to G_1 (no loops).
- Cutset is combined with *slow-down*, where first an N-slow-down version of the graph is created by changing every D to ND, then retiming is used.
- With the N-slow-down version, the input is slowed down too (N-1 null operation or 0 samples must be interleaved with input data

Example

$y(n)=ay(n-2) + by(n-3) +x (n)$

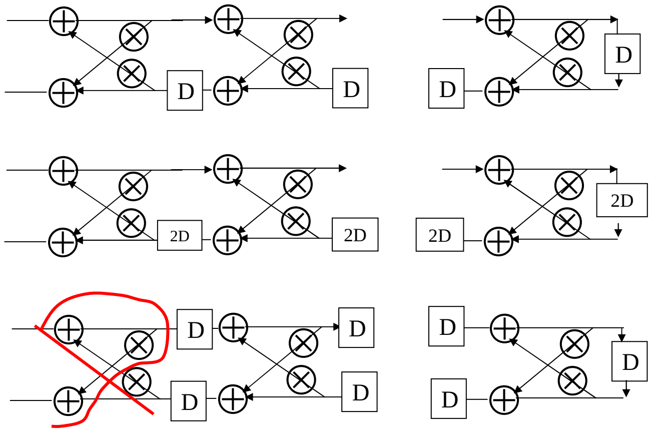




Clock Cycle ?

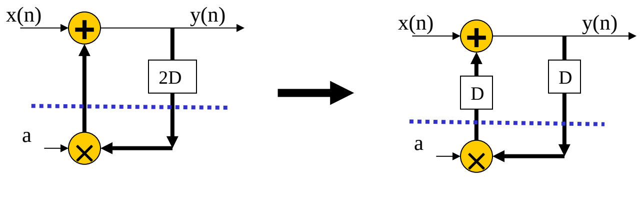
100 stage lattice filter

2-Slow

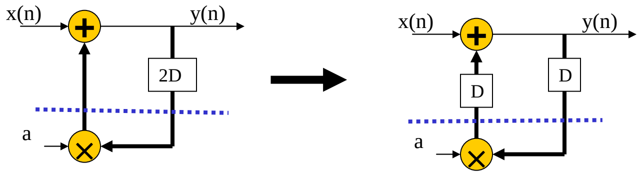


Critical path is reduced from 101 addition + 2 mul to 2 add and 2 mul.

Example

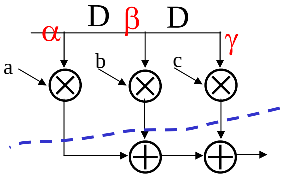


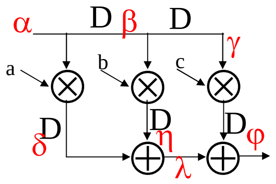
$T_{mul}=T_{add}=1$



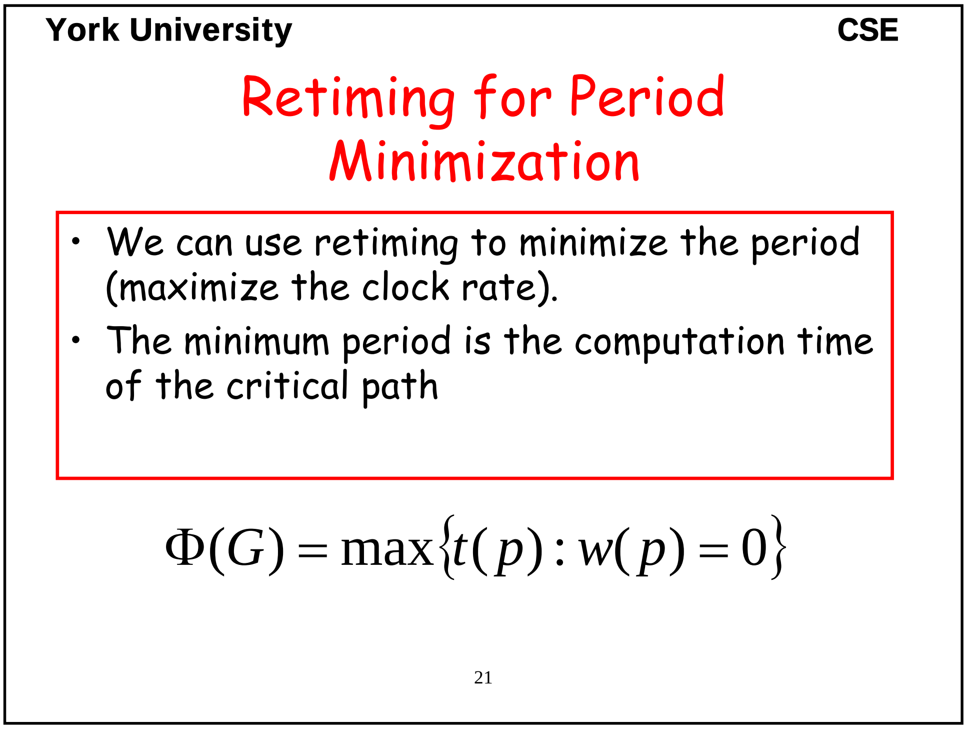
Time	Action	Time	Action	Action
1	y(1)=ay(-1)+x(1)	1	y(1)=x(1)+w(0)	w(1)=ay(0)
2		2	y(2)=x(2)+w(1)	w(2)=ay(1)
3	y(2)=ay(0)+x(2)	3	y(3)=x(3)+w(2)	w(3)=ay(2)
4		4	y(4)=x(4)+w(3)	w(4)=ay(3)
5	y(3)=ay(1)+x(3)	5	y(5)=x(5)+w(4)	w(5)=ay(4)
6		6	y(6)=x(6)+w(5)	w(6)=ay(5)
7	y(4)=ay(2)+x(4)	7	y(7)=x(7)+w(6)	w(7)=ay(6)
8				

Example





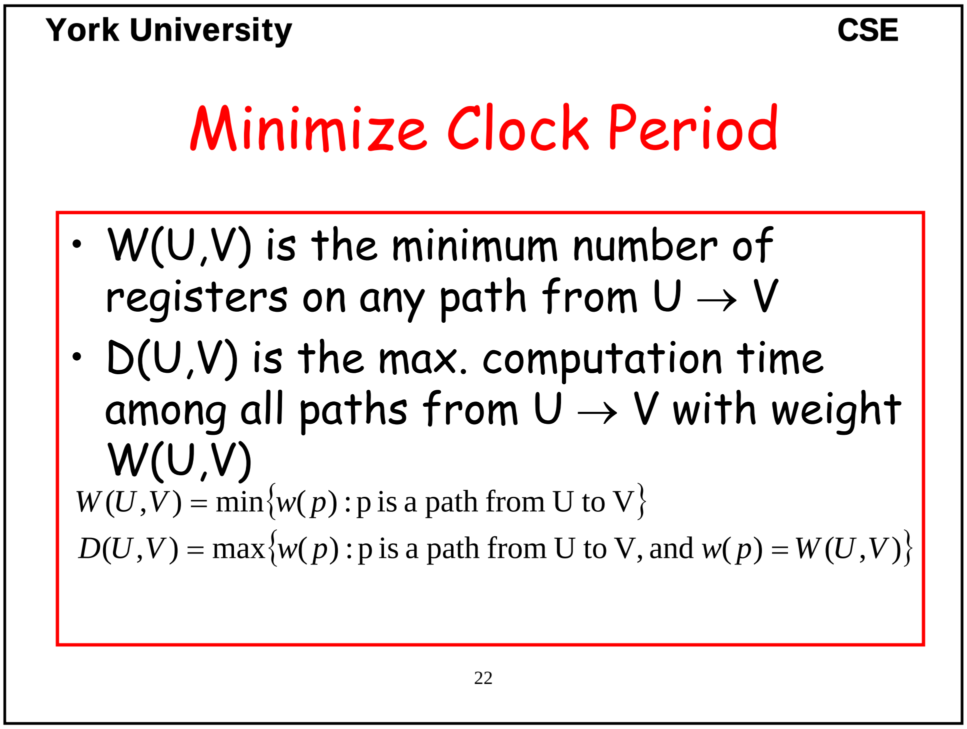
ARE THESE EQUIVALENT?



Retiming for Period Minimization

- We can use retiming to minimize the period (maximize the clock rate).
- The minimum period is the computation time of the critical path

$$\Phi(G) = \max\{t(p) : w(p) = 0\}$$



Minimize Clock Period

- W(U,V) is the minimum number of registers on any path from U → V
- D(U,V) is the max. computation time among all paths from U → V with weight W(U,V)

$$W(U,V) = \min\{w(p) : \text{p is a path from U to V}\}$$

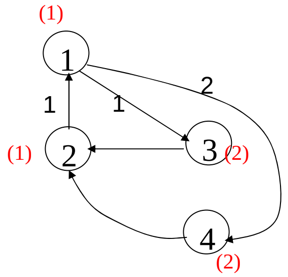
$$D(U,V) = \max\{w(p) : \text{p is a path from U to V, and } w(p) = W(U,V)\}$$

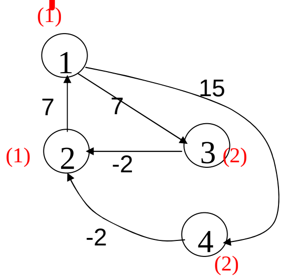
Minimize Clock Period

1. Let $M=n\, t_{max}$, where t_{max} is the max. computation time of any node, n =number of nodes
2. Form a new graph *G'* which is the same as *G* except the edge weights are replaced by $w'(e)=Mw(e)-t(U)$ $(e=U \rightarrow V)$
3. Solve for all-pairs shortest path on *G'* (S_{UV})
   1. If $U \neq V$, then $W(U,V)=\lceil S_{UV}/M \rceil$ and $D(U,V)=MW(U,V)-S_{UV}+t(V)$
   2. IF $U=V$, $W(U,V)=0$, $D(U,V)=t(U)$

Example

$M=nt_{max} = 4*2=8$





$$S_{UV} = \begin{bmatrix} 12 & 5 & 7 & 15 \\ 7 & 12 & 14 & 22 \\ 5 & -2 & 12 & 20 \\ 5 & -2 & 12 & 20 \end{bmatrix}$$

Example

$$S_{UV} = \begin{bmatrix} 12 & 5 & 7 & 15 \\ 7 & 12 & 14 & 22 \\ 5 & -2 & 12 & 20 \\ 5 & -2 & 12 & 20 \end{bmatrix} \qquad W(U,V) = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 1 & 0 & 2 & 3 \\ 1 & 0 & 0 & 3 \\ 1 & 0 & 2 & 0 \end{bmatrix}$$

$$\begin{cases} U \neq V & W(U,V) = \left\lceil \frac{S_{UV}}{M} \right\rceil, D(U,V) = MW(U,V) - S_{UV} + t(V) \\ U = V & W(U,V) = 0, D(U,V) = t(U) \end{cases} \qquad D(U,V) = \begin{bmatrix} 1 & 4 & 3 & 3 \\ 2 & 1 & 4 & 4 \\ 4 & 3 & 2 & 6 \\ 4 & 3 & 6 & 2 \end{bmatrix}$$

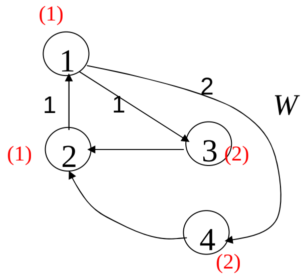
Minimize Clock Period

- After getting W(U,V),D(U,V) construct the following set of constraints
- Feasibility constraints r(U)-r(V) ≤ w(e) for every edge in G
- Critical path constraint r(U)-r(V) ≤ W(U,V)-1for all nodes U,V in G such that D(U,V)>c (cycle time).
- Solve to get r(.) (retiming values)

$$w_r(p)=w(p)+r(b)-r(a)$$

Example

c=3



$$W(U,V)=\begin{bmatrix}0&1&1&2\\1&0&2&3\\1&0&0&3\\1&0&2&0\end{bmatrix}\quad D(U,V)=\begin{bmatrix}1&\underline{4}&3&3\\2&1&\underline{4}&\underline{4}\\\underline{4}&3&2&\underline{6}\\\underline{4}&3&\underline{6}&2\end{bmatrix}$$

Feasibility Constraints

r(1)-r(2) ≤ 1
r(1)-r(4) ≤ 2
r(2)-r(1) ≤ 1
r(1)-r(2) ≤ 1
r(3)-r(2) ≤ 0
r(4)-r(2) ≤ 0

Critical Path Constraints

r(1)-r(2) ≤ 0
r(2)-r(3) ≤ 1
r(2)-r(4) ≤ 2
r(3)-r(1) ≤ 0
r(4)-r(1) ≤ 0
r(4)-r(3) ≤ 1

Retiming for Register Minimization

- The number of registers required to implement the output edges of node v

$$R_V = \max_{V \xrightarrow{e} ?} \{W_r(e)\}$$

- Minimize

$COST = \sum R_V$

$R_V \geq w_r(e)$ for all V and all edges $V \xrightarrow{e} ?$

$r(U) - r(V) \leq w(e)$ for every edge $U \xrightarrow{e} V$

$r(U) - r(V) \leq W(U,V) - 1$ for every vertices U,V such that $D(U,V) > c$