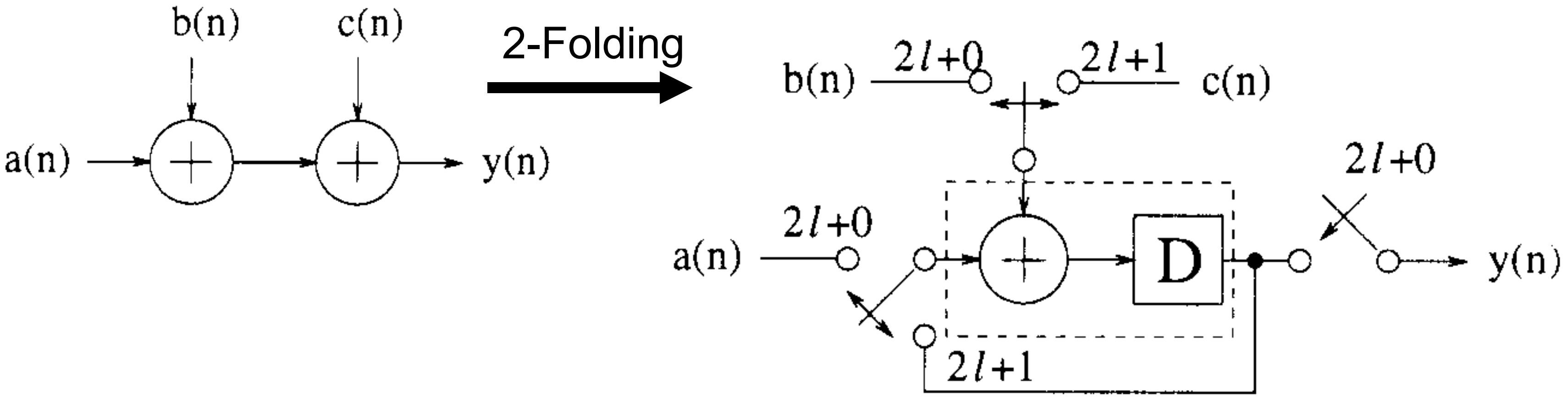
Folding

Shao-Yi Chien

Introduction

- Folding transform is used to systematically determine the control circuits in DSP architectures where multiple algorithm operations are **time-multiplexed** to a single functional unit
  - Trading area for time in a DSP architecture
  - Reducing the number of hardware functional units by a factor of N at the expense of increasing the computation time by a factor of N

An Example of Folding



Scheduling

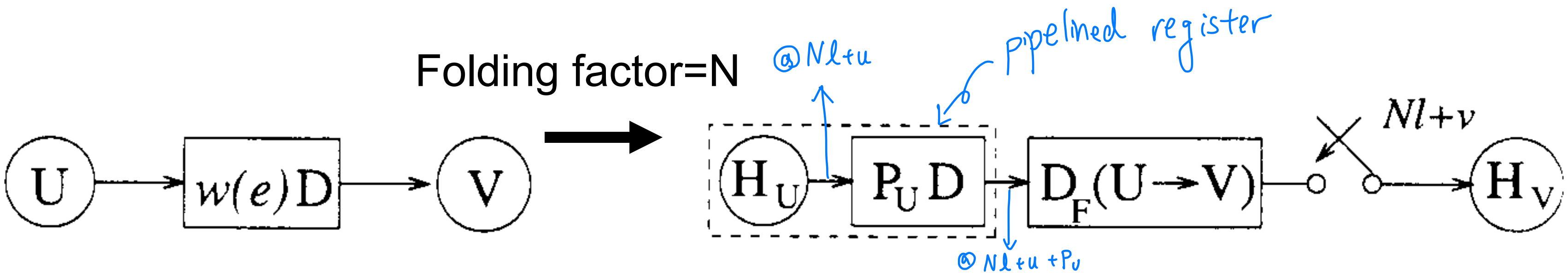
Cycle	Adder Input (left)	Adder Input (top)	System Output
0	$a(0)$	$b(0)$	–
1	$a(0)+b(0)$	$c(0)$	–
2	$a(1)$	$b(1)$	$a(0)+b(0)+c(0)$
3	$a(1)+b(1)$	$c(1)$	–
4	$a(2)$	$b(2)$	$a(1)+b(1)+c(1)$
5	$a(2)+b(2)$	$c(2)$	–

Folding Transform

N: folding factor
u, v are namely folding orders

H_u H_v
@$Nl+u$ @$Nl+v$

$0 \le u, v \le N-1$



$$D_F(U \xrightarrow{e} V) = [N(l+w(e))+v] - [Nl+P_U+u] = Nw(e) - P_U + v - u$$

(1) U is executed in H_U and V is executed in H_V
(2) Data leave H_U at $Nl+u$, and reach H_V at $Nl+v$
(3) H_U is pipelined by P_U stages

Folding Transform

- Folding set
  - Ex: a folding set $S_1 = \{A_1, \phi, A_2\}$ for N=3 for a functional unit means A_1 is executed at time 3l+0 (S_1|0), and A_2 is executed at time 3l+2 (S_1|2)

Folding Transform

$N = |S_1| = |S_2| = 4$

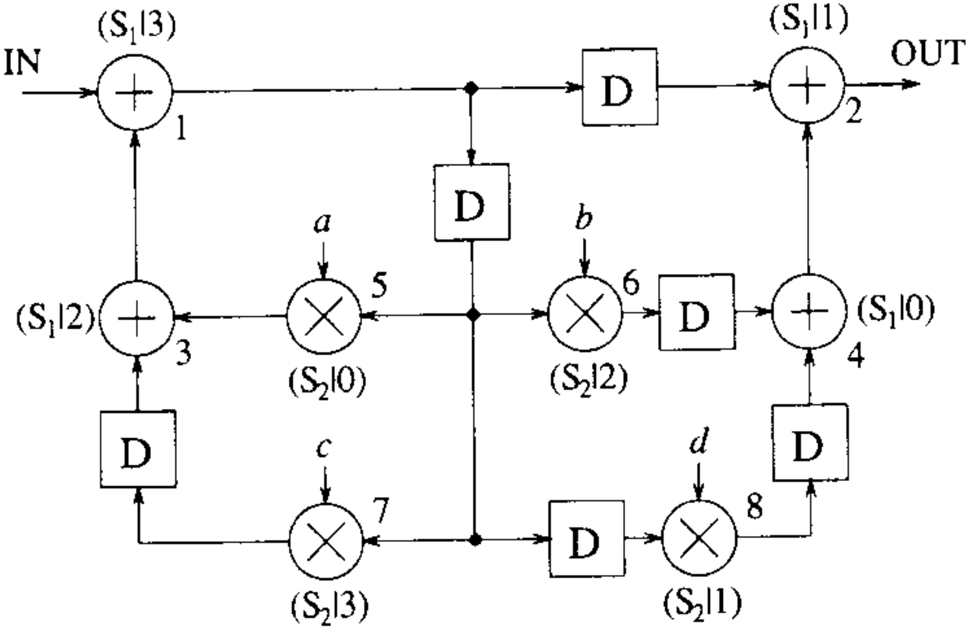
$w(e)$: delay on the path

P_U: Pipeline stages

v: Dest. node index e.g $(S_i|1) = 1$

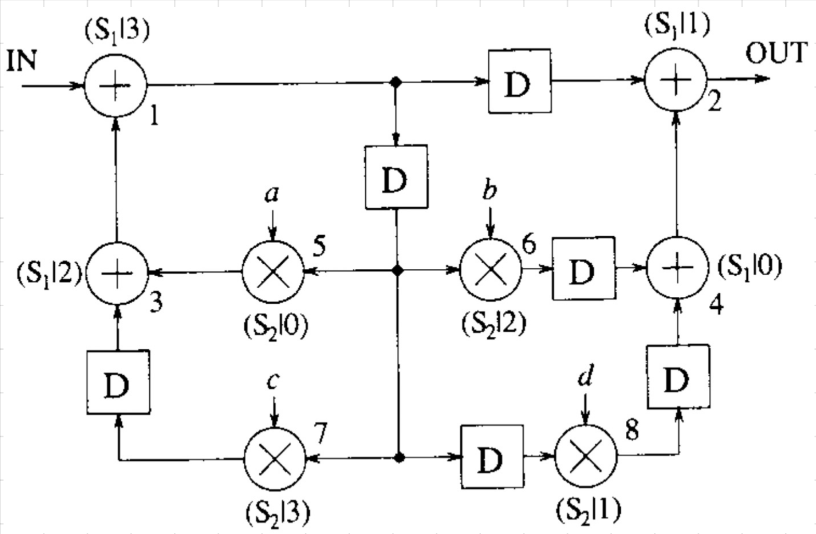
u: Source node index

$$\boxed{D_F(U \xrightarrow{e} V) = [N(l+w(e))+v] - [Nl + P_U + u] = Nw(e) - P_U + v - u}$$



$$\begin{aligned}
D_F(1 \to 2) &= 4(1) - 1 + 1 - 3 = 1 \\
D_F(1 \to 5) &= 4(1) - 1 + 0 - 3 = 0 \\
D_F(1 \to 6) &= 4(1) - 1 + 2 - 3 = 2 \\
D_F(1 \to 7) &= 4(1) - 1 + 3 - 3 = 3 \\
D_F(1 \to 8) &= 4(2) - 1 + 1 - 3 = 5 \\
D_F(3 \to 1) &= 4(0) - 1 + 3 - 2 = 0 \\
D_F(4 \to 2) &= 4(0) - 1 + 1 - 0 = 0 \\
D_F(5 \to 3) &= 4(0) - 2 + 2 - 0 = 0 \\
D_F(6 \to 4) &= 4(1) - 2 + 0 - 2 = 0 \\
D_F(7 \to 3) &= 4(1) - 2 + 2 - 3 = 1 \\
D_F(8 \to 4) &= 4(1) - 2 + 0 - 1 = 1.
\end{aligned}$$

S_1={4,2,3,1} for one adder with 1 stage pipelining P_A=1
S_2={5,8,6,7} for one multiplier with 2 stages pipelining P_M=2
N=4



~~$D_F(1 \to 2) = 4(1) - 1 + 1 - 3 = 1$~~

~~$D_F(1 \to 5) = 4(1) - 1 + 0 - 3 = 0$~~

~~$D_F(1 \to 6) = 4(1) - 1 + 2 - 3 = 2$~~

~~$D_F(1 \to 7) = 4(1) - 1 + 3 - 3 = 3$~~

~~$D_F(1 \to 8) = 4(2) - 1 + 1 - 3 = 5$~~

~~$D_F(3 \to 1) = 4(0) - 1 + 3 - 2 = 0$~~

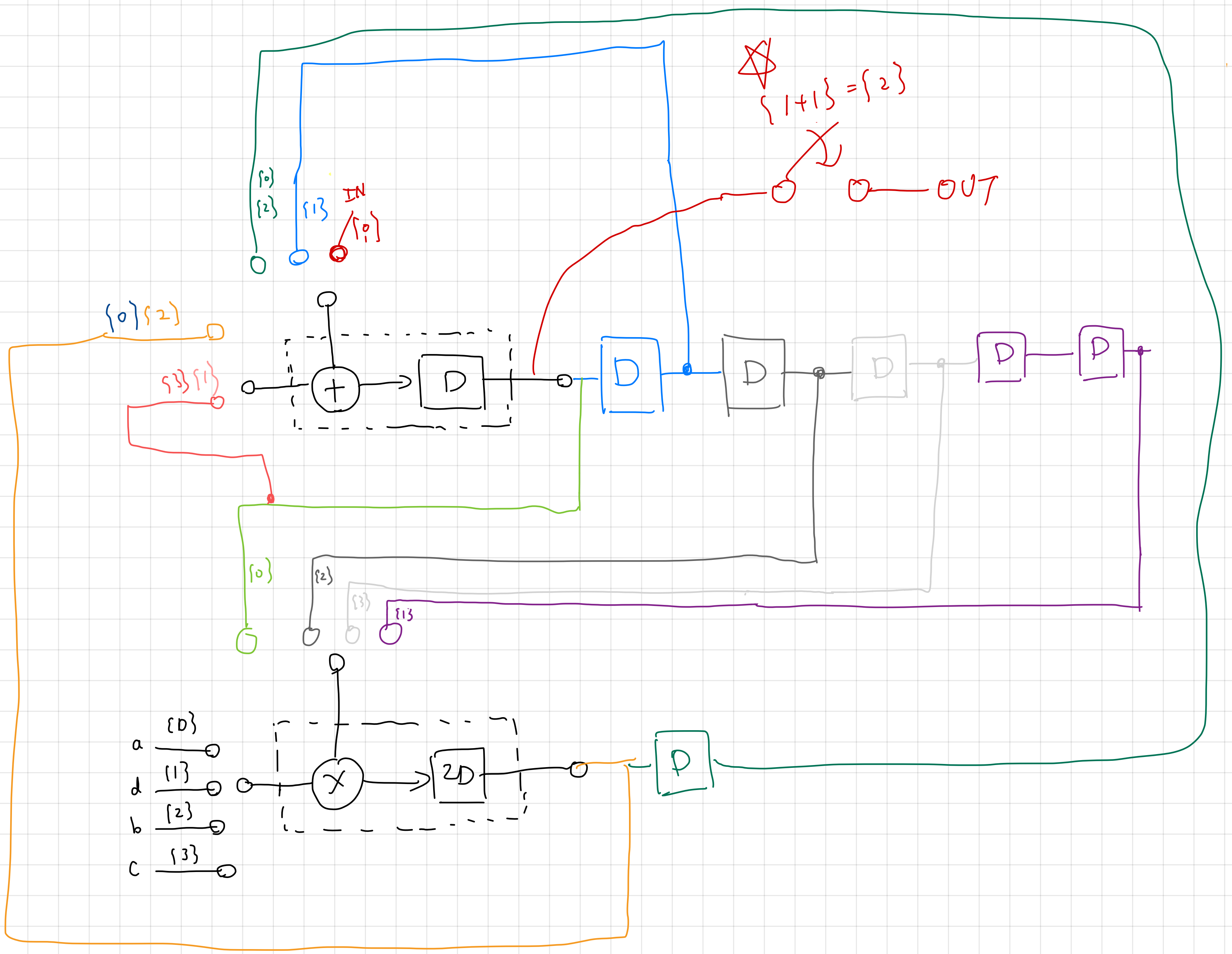
~~$D_F(4 \to 2) = 4(0) - 1 + 1 - 0 = 0$~~

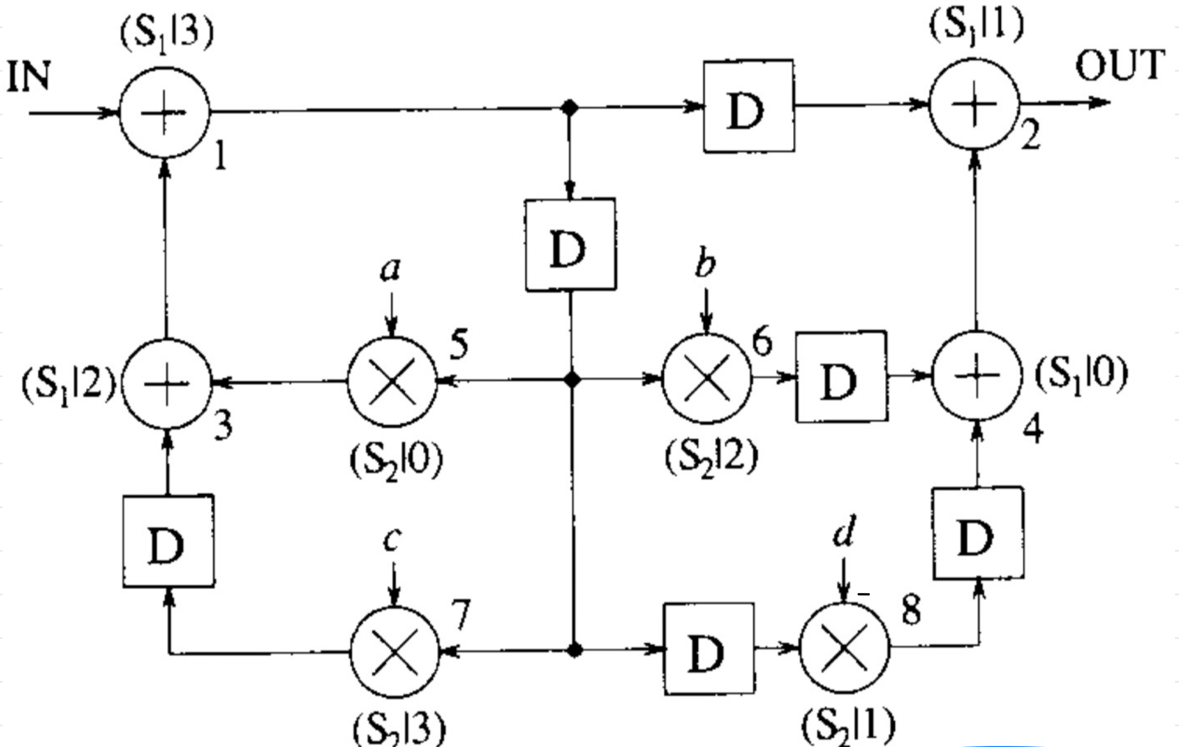
~~$D_F(5 \to 3) = 4(0) - 2 + 2 - 0 = 0$~~

~~$D_F(6 \to 4) = 4(1) - 2 + 0 - 2 = 0$~~

~~$D_F(7 \to 3) = 4(1) - 2 + 2 - 3 = 1$~~

$D_F(8 \to 4) = 4(1) - 2 + 0 - 1 = 1.$





~~$D_F(1 \rightarrow 2) = 4(1) - 1 + 1 - 3 = 1$~~

~~$D_F(1 \rightarrow 5) = 4(1) - 1 + 0 - 3 = 0$~~

~~$D_F(1 \rightarrow 6) = 4(1) - 1 + 2 - 3 = 2$~~

~~$D_F(1 \rightarrow 7) = 4(1) - 1 + 3 - 3 = 3$~~

~~$D_F(1 \rightarrow 8) = 4(2) - 1 + 1 - 3 = 5$~~

~~$D_F(3 \rightarrow 1) = 4(0) - 1 + 3 - 2 = 0$~~

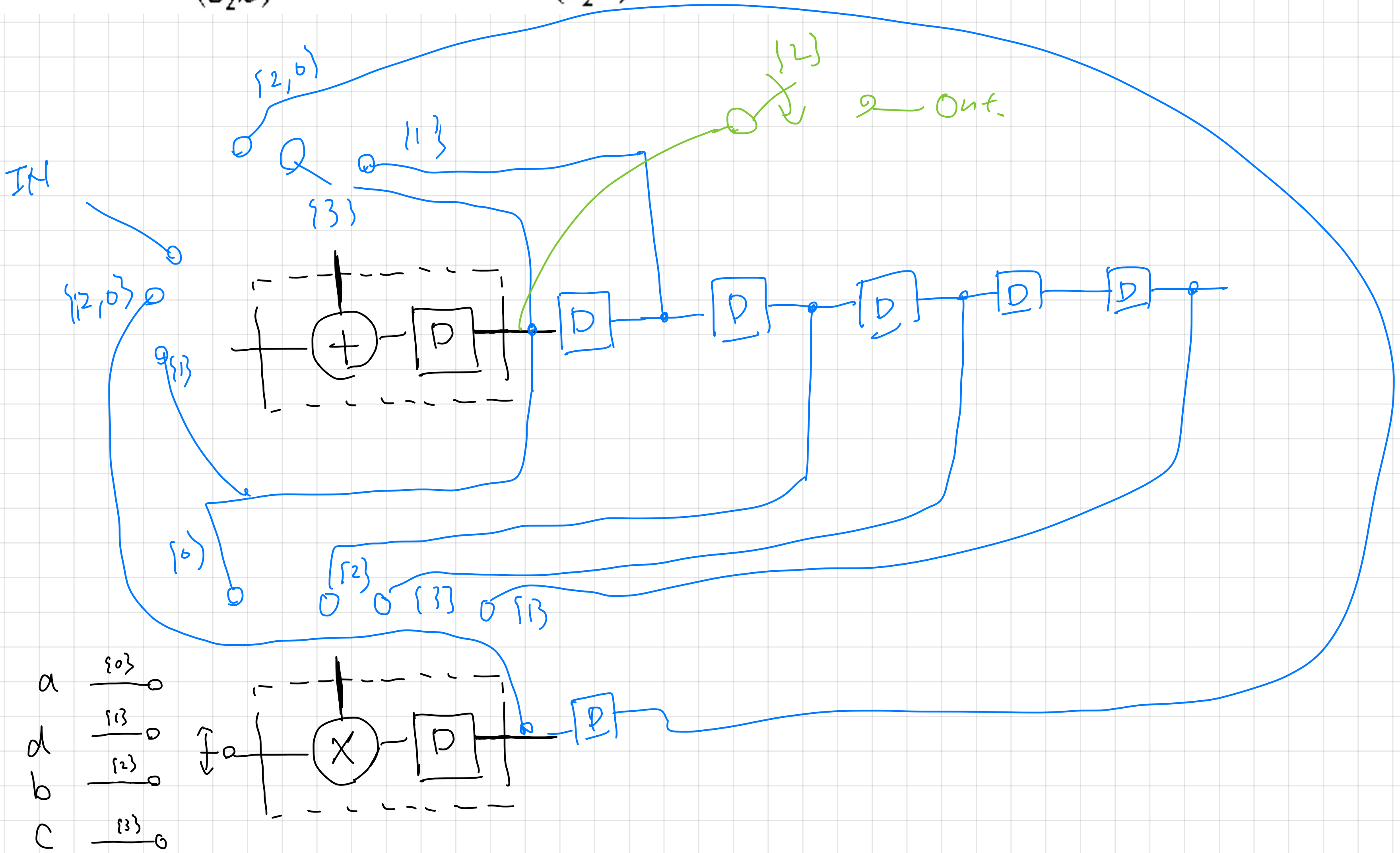
~~$D_F(4 \rightarrow 2) = 4(0) - 1 + 1 - 0 = 0$~~

~~$D_F(5 \rightarrow 3) = 4(0) - 2 + 2 - 0 = 0$~~

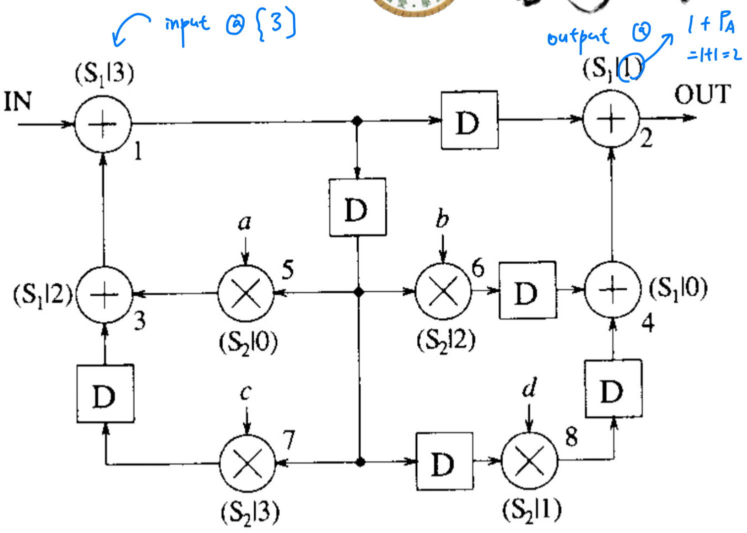
~~$D_F(6 \rightarrow 4) = 4(1) - 2 + 0 - 2 = 0$~~

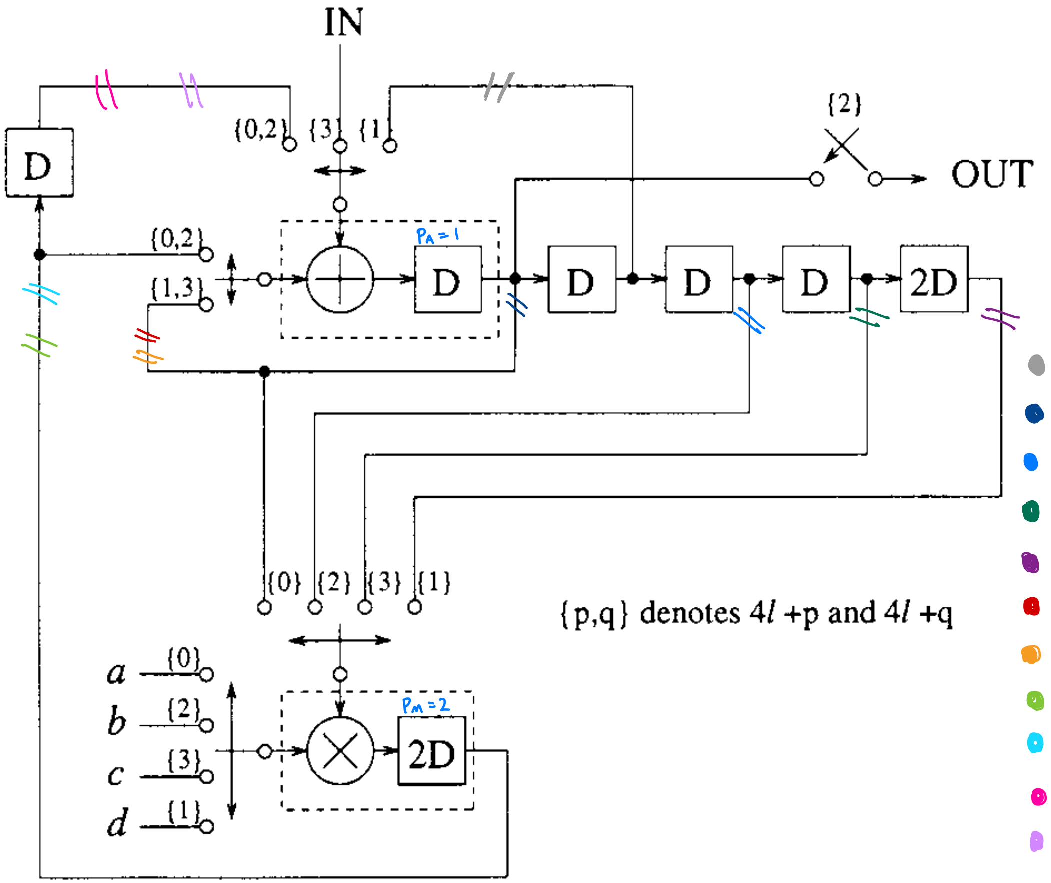
~~$D_F(7 \rightarrow 3) = 4(1) - 2 + 2 - 3 = 1$~~

$D_F(8 \rightarrow 4) = 4(1) - 2 + 0 - 1 = 1.$



Folding Transform





{p,q} denotes 4*l* +p and 4*l* +q

- ● $D_F(1 \rightarrow 2) = 4(1) - 1 + 1 - 3 = 1$
- ● $D_F(1 \rightarrow 5) = 4(1) - 1 + 0 - 3 = 0$
- ● $D_F(1 \rightarrow 6) = 4(1) - 1 + 2 - 3 = 2$
- ● $D_F(1 \rightarrow 7) = 4(1) - 1 + 3 - 3 = 3$
- ● $D_F(1 \rightarrow 8) = 4(2) - 1 + 1 - 3 = 5$
- ● $D_F(3 \rightarrow 1) = 4(0) - 1 + 3 - 2 = 0$
- ● $D_F(4 \rightarrow 2) = 4(0) - 1 + 1 - 0 = 0$
- ● $D_F(5 \rightarrow 3) = 4(0) - 2 + 2 - 0 = 0$
- ● $D_F(6 \rightarrow 4) = 4(1) - 2 + 0 - 2 = 0$
- ● $D_F(7 \rightarrow 3) = 4(1) - 2 + 2 - 3 = 1$
- ● $D_F(8 \rightarrow 4) = 4(1) - 2 + 0 - 1 = 1.$

Retiming for Folding (1/6)

- Realizable folding: $D_F(U \xrightarrow{e} V) \geq 0$
- Once valid folding sets have been assigned, retiming can be used to either satisfy this property or determine that the folding sets are not feasible

Retiming for Folding (2/6)

- Retiming constraints:

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

$$D'_F(U \xrightarrow{e} V) \geq 0$$

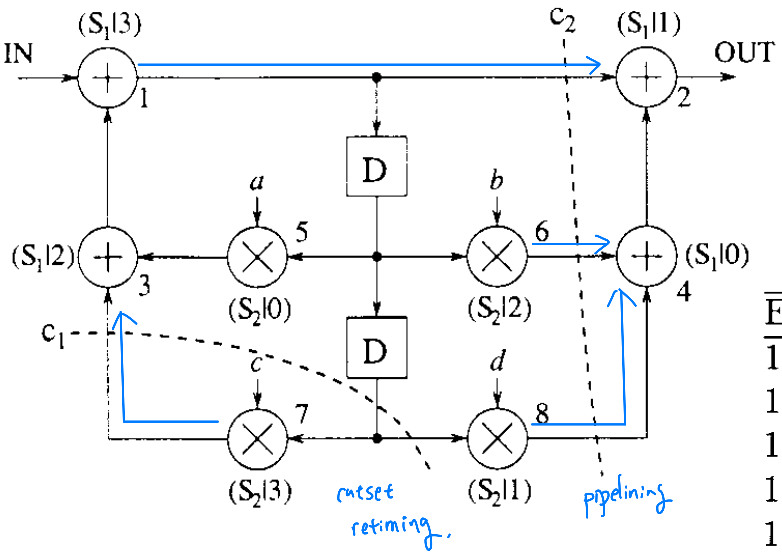
$$N w_r(e) - P_U + v - u \geq 0.$$

$$N(w(e) + r(V) - r(U)) - P_U + v - u \geq 0.$$

$$r(U) - r(V) \leq \frac{D_F(U \xrightarrow{e} V)}{N}.$$

$$\boxed{r(U) - r(V) \leq \left\lfloor \frac{D_F(U \xrightarrow{e} V)}{N} \right\rfloor}$$

Retiming for Folding (3/6)



$$
\boxed{r(U) - r(V) \leq \left\lfloor \frac{D_F(U \xrightarrow{e} V)}{N} \right\rfloor}
$$

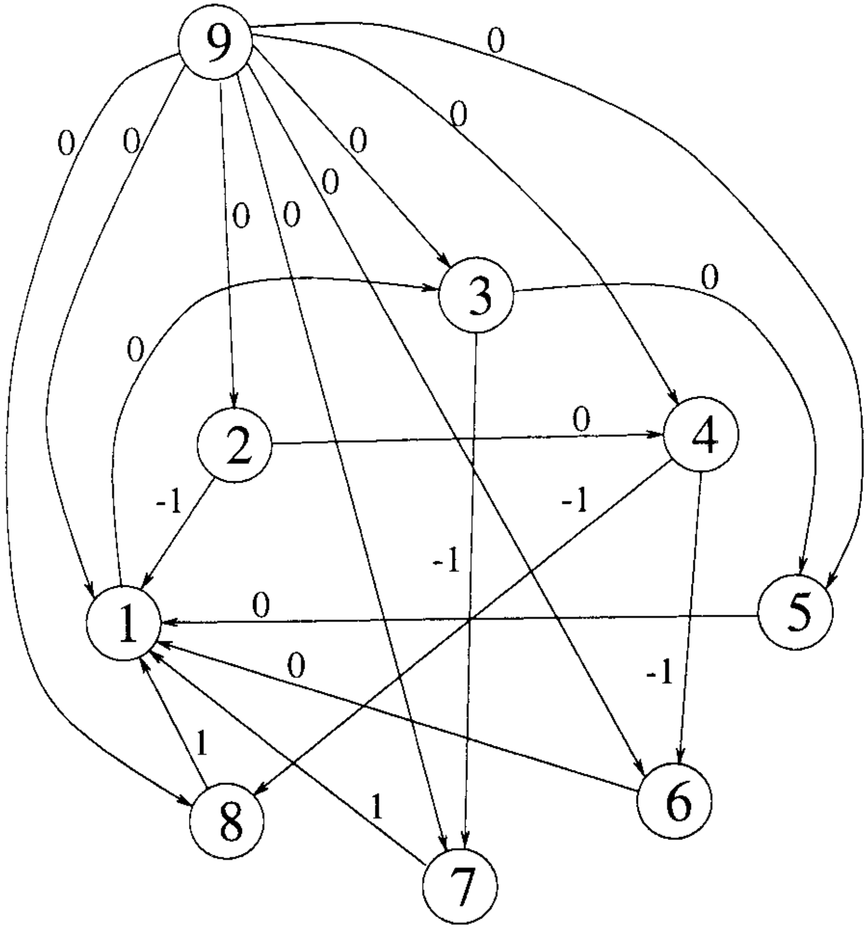
Edge	Folding Equation	Retiming for Folding Constraint
$1 \rightarrow 2$	$D_F(1 \rightarrow 2) = -3$	$r(1) - r(2) \leq -1$ ✓
$1 \rightarrow 5$	$D_F(1 \rightarrow 5) = 0$	$r(1) - r(5) \leq 0$
$1 \rightarrow 6$	$D_F(1 \rightarrow 6) = 2$	$r(1) - r(6) \leq 0$
$1 \rightarrow 7$	$D_F(1 \rightarrow 7) = 7$	$r(1) - r(7) \leq 1$
$1 \rightarrow 8$	$D_F(1 \rightarrow 8) = 5$	$r(1) - r(8) \leq 1$
$3 \rightarrow 1$	$D_F(3 \rightarrow 1) = 0$	$r(3) - r(1) \leq 0$
$4 \rightarrow 2$	$D_F(4 \rightarrow 2) = 0$	$r(4) - r(2) \leq 0$
$5 \rightarrow 3$	$D_F(5 \rightarrow 3) = 0$	$r(5) - r(3) \leq 0$
$6 \rightarrow 4$	$D_F(6 \rightarrow 4) = -4$	$r(6) - r(4) \leq -1$ ✓
$7 \rightarrow 3$	$D_F(7 \rightarrow 3) = -3$	$r(7) - r(3) \leq -1$ ✓
$8 \rightarrow 4$	$D_F(8 \rightarrow 4) = -3$	$r(8) - r(4) \leq -1$ ✓

< 0 : too less delay elements

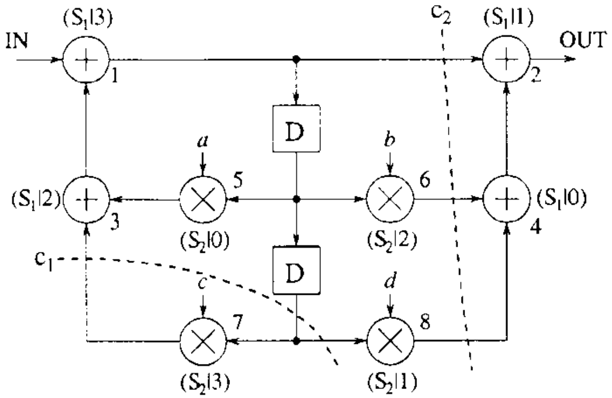
solution: add registers on those paths

Retiming for Folding (4/6)

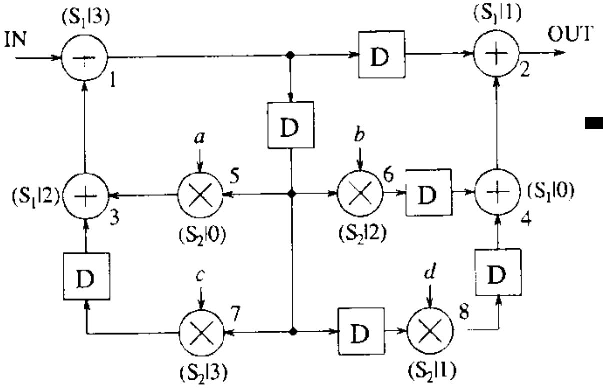
- Constraint graph
  - □ r(1)=-1
  - □ r(2)=0
  - □ r(3)=-1
  - □ r(4)=0
  - □ r(5)=-1
  - □ r(6)=-1
  - □ r(7)=-2
  - □ r(8)=-1



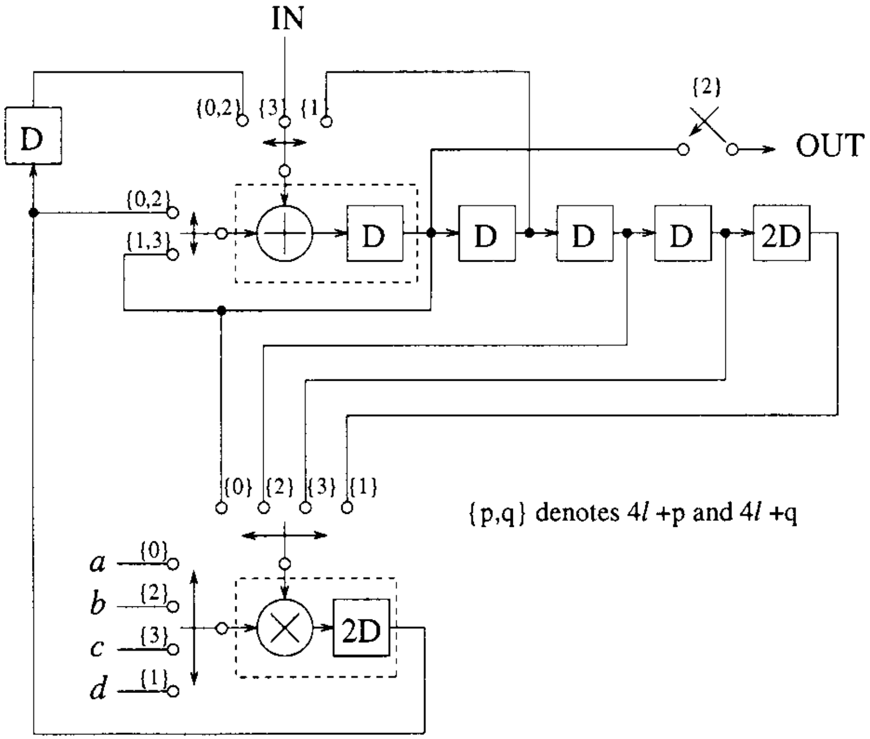
Retiming for Folding (5/6)



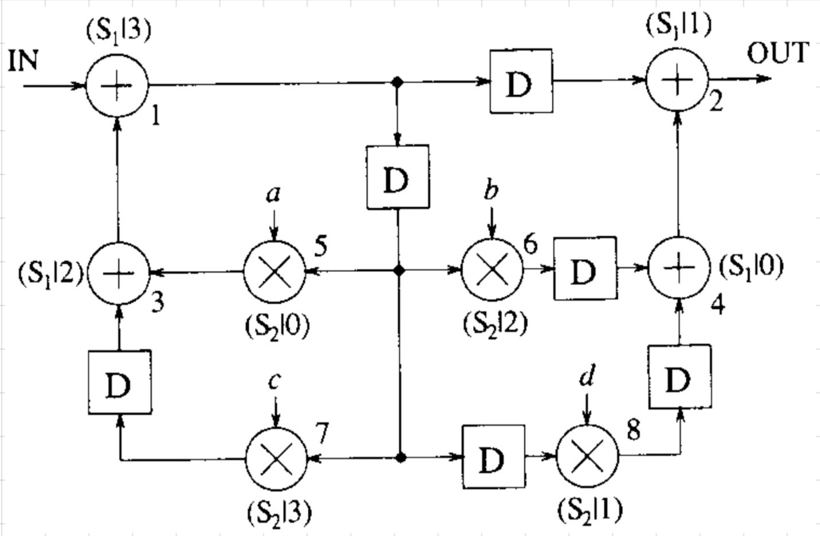
Retiming



Folding



{p,q} denotes 4*l* +p and 4*l* +q



~~$D_F(1 \to 2) = 4(1) - 1 + 1 - 3 = 1$~~

~~$D_F(1 \to 5) = 4(1) - 1 + 0 - 3 = 0$~~

~~$D_F(1 \to 6) = 4(1) - 1 + 2 - 3 = 2$~~

~~$D_F(1 \to 7) = 4(1) - 1 + 3 - 3 = 3$~~

~~$D_F(1 \to 8) = 4(2) - 1 + 1 - 3 = 5$~~

~~$D_F(3 \to 1) = 4(0) - 1 + 3 - 2 = 0$~~

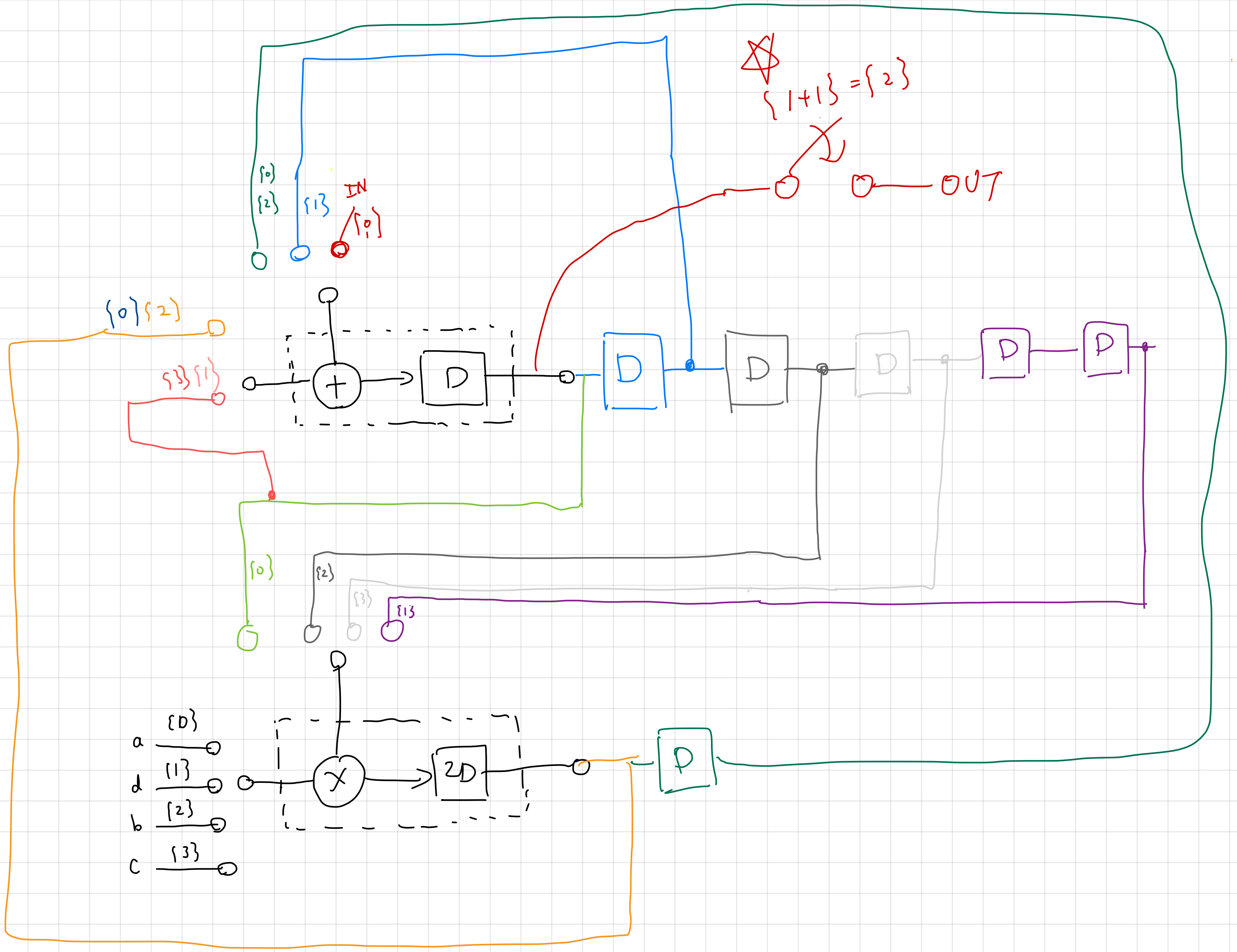
~~$D_F(4 \to 2) = 4(0) - 1 + 1 - 0 = 0$~~

~~$D_F(5 \to 3) = 4(0) - 2 + 2 - 0 = 0$~~

~~$D_F(6 \to 4) = 4(1) - 2 + 0 - 2 = 0$~~

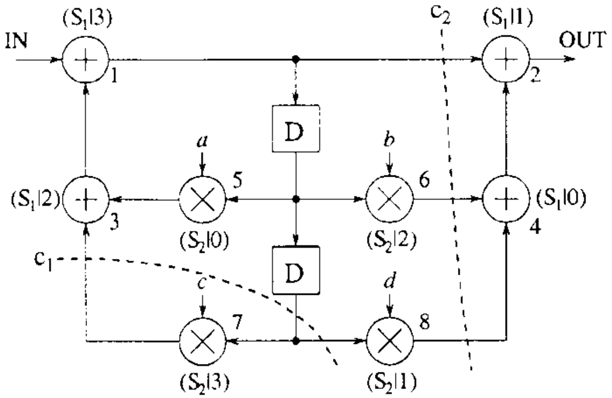
~~$D_F(7 \to 3) = 4(1) - 2 + 2 - 3 = 1$~~

$D_F(8 \to 4) = 4(1) - 2 + 0 - 1 = 1.$



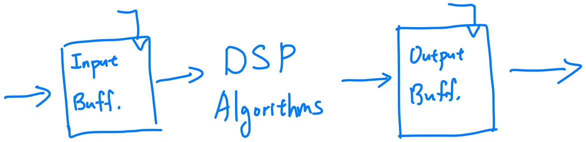
Retiming for Folding (6/6)

- Another point of view
  - □ Apply cutset retiming at c_1 and c_2 to add/subtract w delays
  - □ ➔add/subtract Nw on D_F
  - □ To make D_F>=0



Edge	Folding Equation	Retiming for Folding Constraint
$1 \rightarrow 2$	$D_F(1 \rightarrow 2) = -3$	$r(1) - r(2) \leq -1$
$1 \rightarrow 5$	$D_F(1 \rightarrow 5) = 0$	$r(1) - r(5) \leq 0$
$1 \rightarrow 6$	$D_F(1 \rightarrow 6) = 2$	$r(1) - r(6) \leq 0$
$1 \rightarrow 7$	$D_F(1 \rightarrow 7) = 7$	$r(1) - r(7) \leq 1$
$1 \rightarrow 8$	$D_F(1 \rightarrow 8) = 5$	$r(1) - r(8) \leq 1$
$3 \rightarrow 1$	$D_F(3 \rightarrow 1) = 0$	$r(3) - r(1) \leq 0$
$4 \rightarrow 2$	$D_F(4 \rightarrow 2) = 0$	$r(4) - r(2) \leq 0$
$5 \rightarrow 3$	$D_F(5 \rightarrow 3) = 0$	$r(5) - r(3) \leq 0$
$6 \rightarrow 4$	$D_F(6 \rightarrow 4) = -4$	$r(6) - r(4) \leq -1$
$7 \rightarrow 3$	$D_F(7 \rightarrow 3) = -3$	$r(7) - r(3) \leq -1$
$8 \rightarrow 4$	$D_F(8 \rightarrow 4) = -3$	$r(8) - r(4) \leq -1$

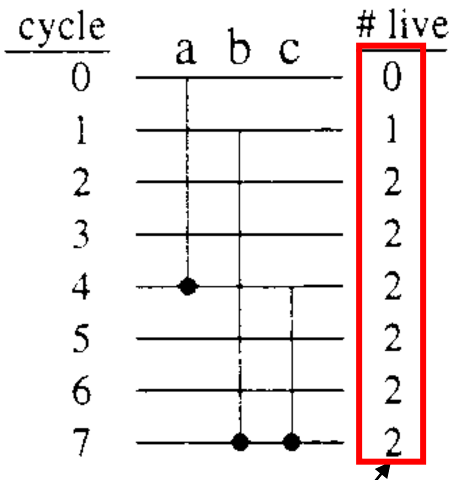
Register Minimization Techniques (1/8)



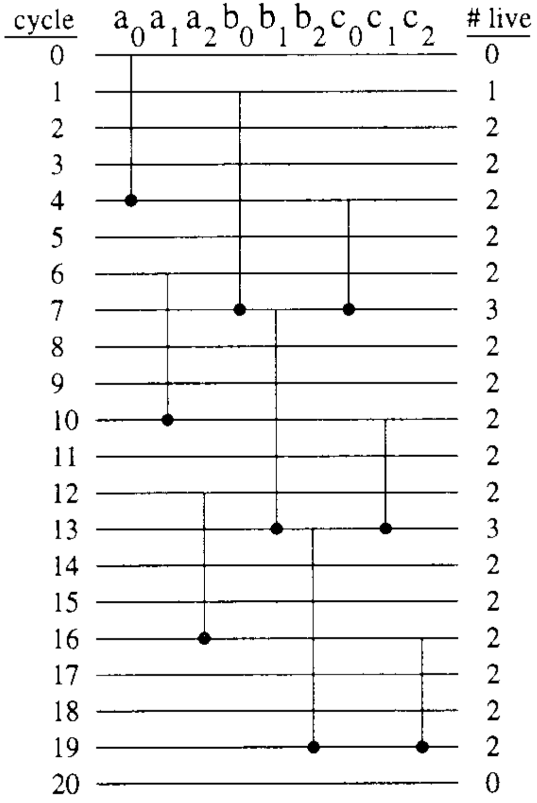
- Lifetime analysis
  - A procedure used to compute the minimum number of registers required to implement a DSP algorithm in hardware
  - Ex:
    - a lives during time unit {1, 2, 3, 4}
    - b lives during time unit {2, 3, 4, 5, 6, 7}
    - c lives during time unit {5, 6, 7}

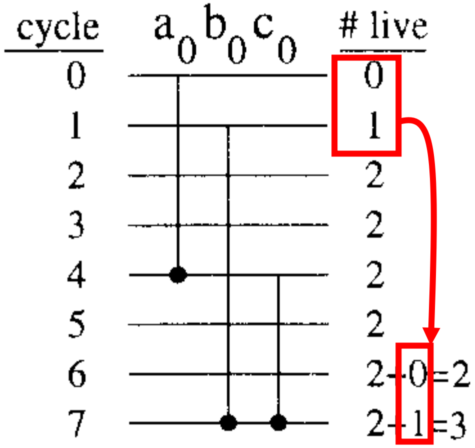
Register Minimization Techniques (2/8)

- Lifetime analysis—linear lifetime chart



Minimum number of required registers

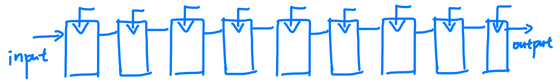




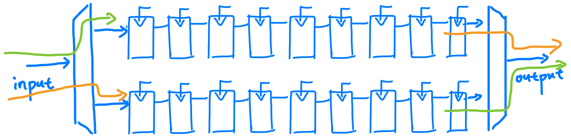
3 iterations with period N=6

Register Minimization Techniques (3/8)

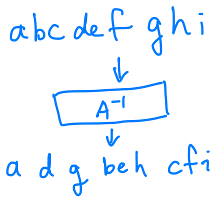
A Naïve Way (Timing may not be correct): Shift Registers



Double Buffer (Pin-Pong buffer)



- Lifetime analysis—lifetime table
- Ex: transpose matrix



$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

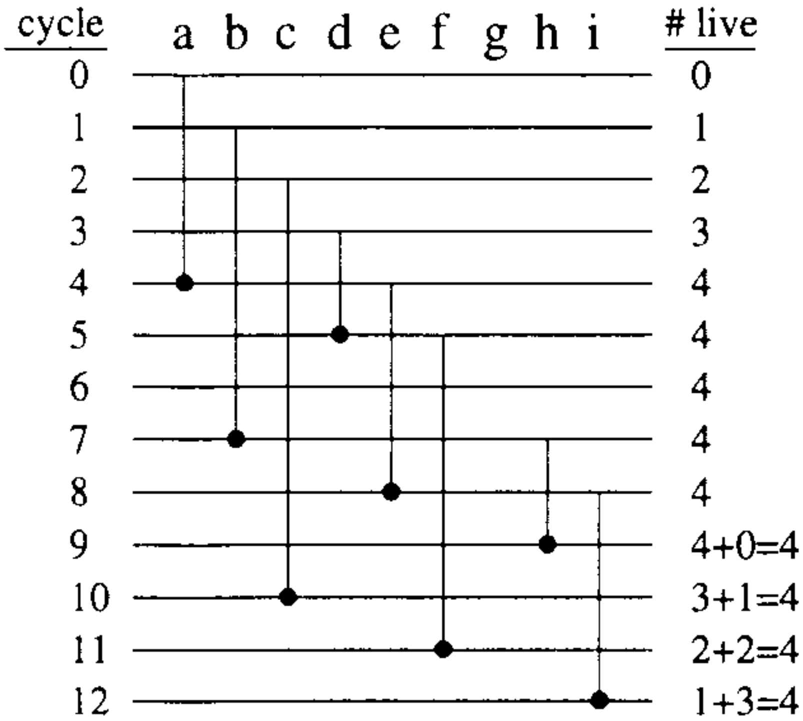
Input in Raster Scan Order

$T_{output}=T_{zlout}$+latency

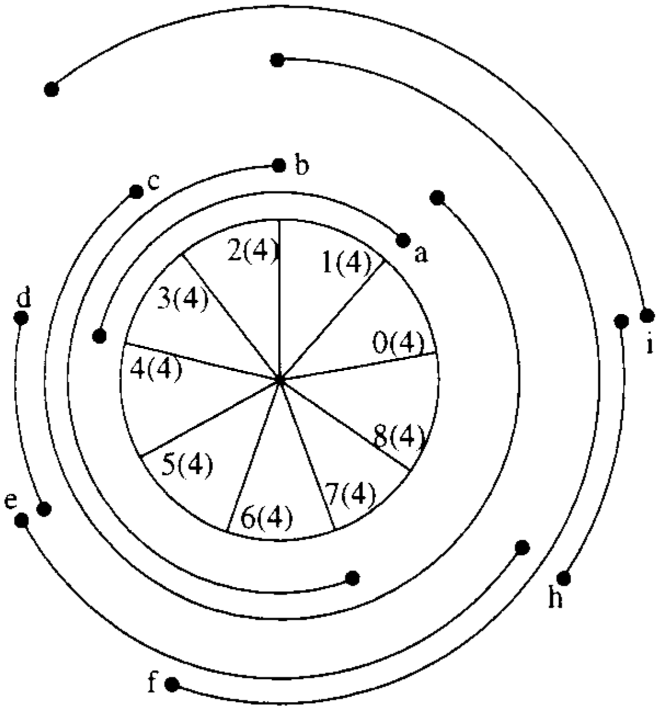
Sample	T_{input}	T_{zlout}	T_{diff}	T_{output}	Life Period $(T_{input} \to T_{output})$
a	0	0	0	4	$0 \to 4$
b	1	3	2	7	$1 \to 7$
c	2	6	4	10	$2 \to 10$
d	3	1	$\boxed{-2}$	5	$3 \to 5$
e	4	4	0	8	$4 \to 8$
f	5	7	2	11	$5 \to 11$
g	6	2	$\boxed{-4}$	6	$6 \to 6$
h	7	5	$\boxed{-2}$	9	$7 \to 9$
i	8	8	0	12	$8 \to 12$

Illegal!→add latency 4

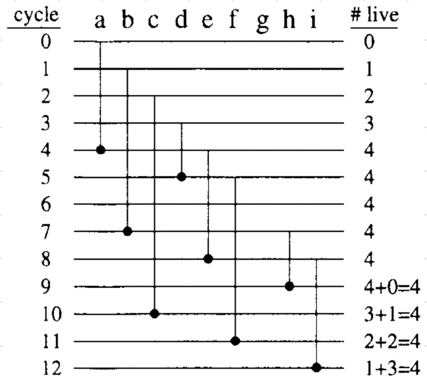
Register Minimization Techniques (4/8)



Linear lifetime chart



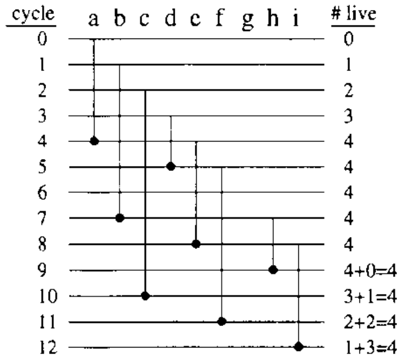
Circular lifetime chart



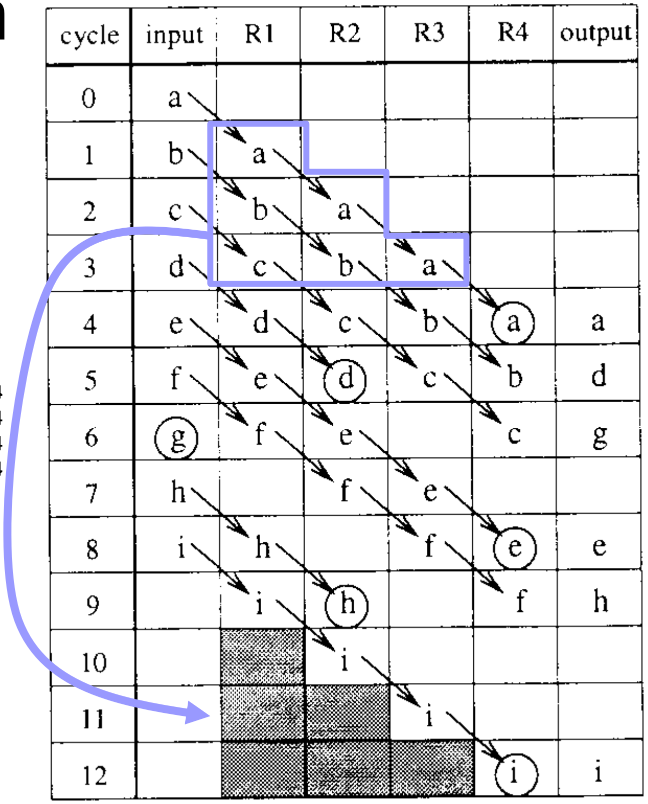
cycle	i/p	R1	R2	R3	R4	o/p
0	a					
1	b	a				
2	c	b	a			
3	d	c	b	a		
4	e	d	c	b	a	a
5	f	e	d	c	b	d
6	g	f	e	b	c	g
7	h	c	f	e	b	b
8	i	h	c	f	e	e
9	X	i	h	c	f	h
10	X	X	i	f	c	c
11	X	X	X	i	f	f
12	X	X	X	X	i	i

Register Minimization Techniques (5/8)

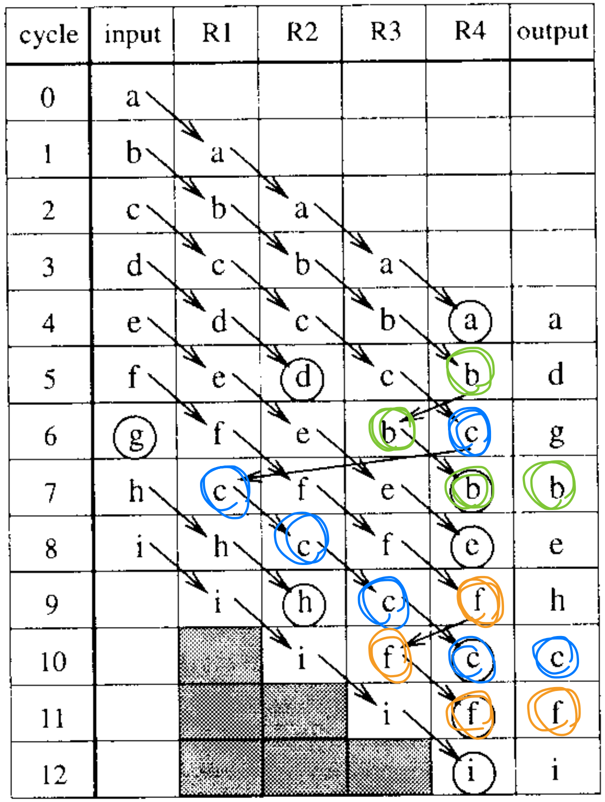
- Data allocation using forward-backward register allocation



N=9
Hashing

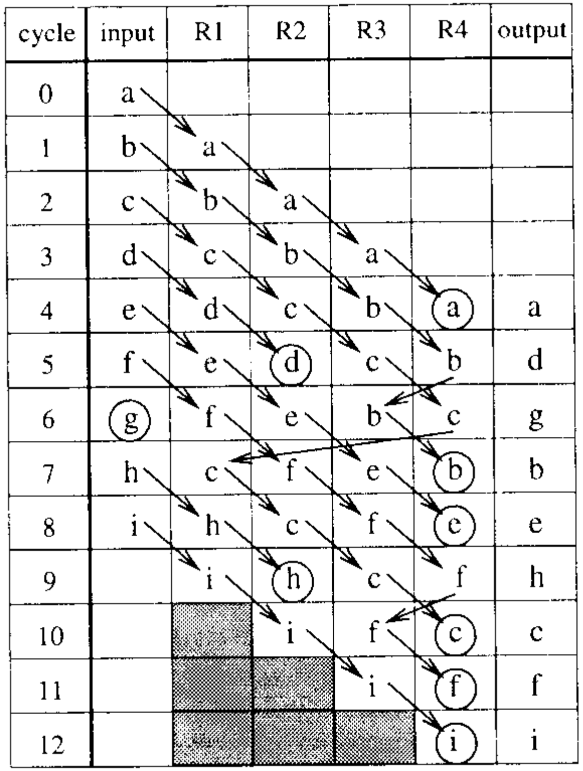


cycle	input	R1	R2	R3	R4	output
0	a					
1	b	a				
2	c	b	a			
3	d	c	b	a		
4	e	d	c	b	a	a
5	f	e	d	c	b	d
6	g	f	e		c	g
7	h		f	e		
8	i	h		f	e	e
9		i	h		f	h
10		▨	i			
11		▨	▨	i		
12		▨	▨	▨	i	i

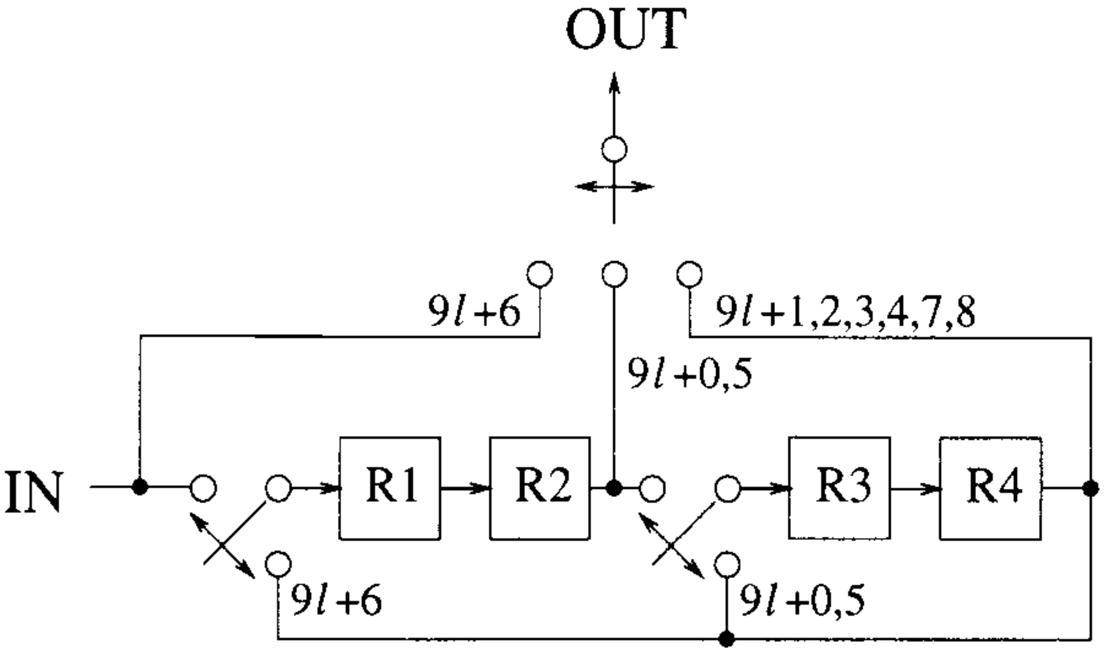


cycle	input	R1	R2	R3	R4	output
0	a					
1	b	a				
2	c	b	a			
3	d	c	b	a		
4	e	d	c	b	a	a
5	f	e	d	c	b	d
6	g	f	e	b	c	g
7	h	c	f	e	b	b
8	i	h	c	f	e	e
9		i	h	c	f	h
10		▨	i	f	c	c
11		▨	▨	i	f	f
12		▨	▨	▨	i	i

Register Minimization Techniques (6/8)

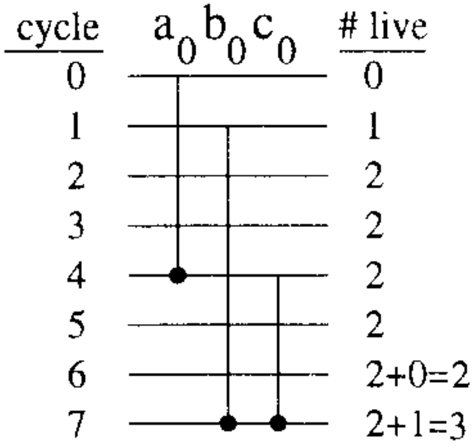


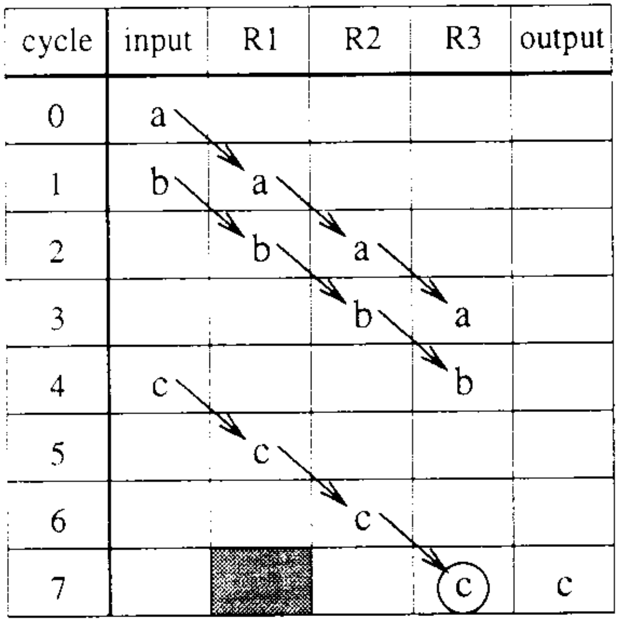
cycle	input	R1	R2	R3	R4	output
0	a					
1	b	a				
2	c	b	a			
3	d	c	b	a		
4	e	d	c	b	a	a
5	f	e	d	c	b	d
6	g	f	e	b	c	g
7	h	c	f	e	b	b
8	i	h	c	f	e	e
9		i	h	c	f	h
10		▨	i	f	c	c
11		▨	▨	i	f	f
12		▨	▨	▨	i	i



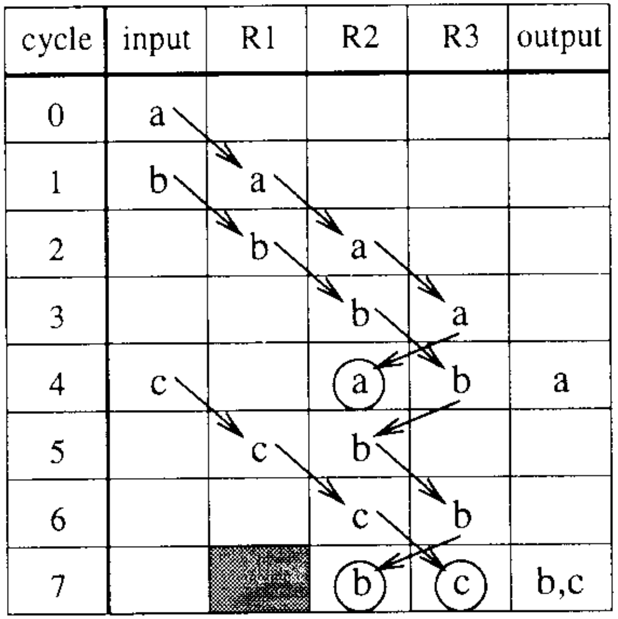
9l+9 = 9l+0

Register Minimization Techniques (7/8)



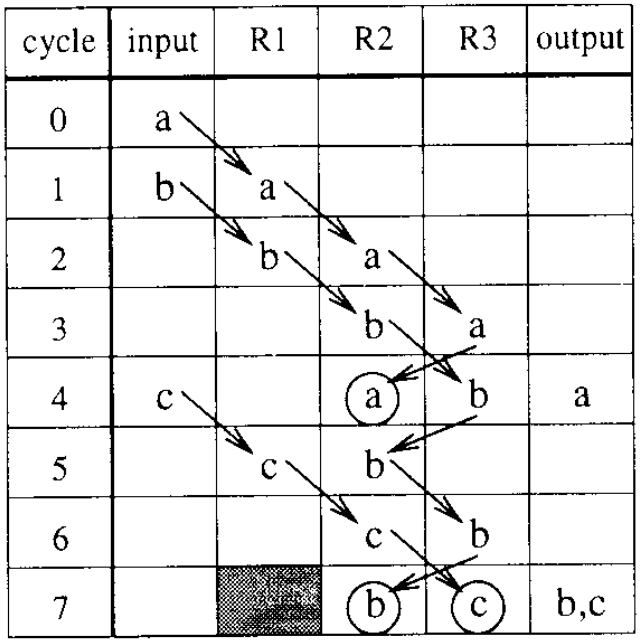


cycle	input	R1	R2	R3	output
0	a				
1	b	a			
2		b	a		
3			b	a	
4	c			b	
5		c			
6			c		
7		▨		c	c

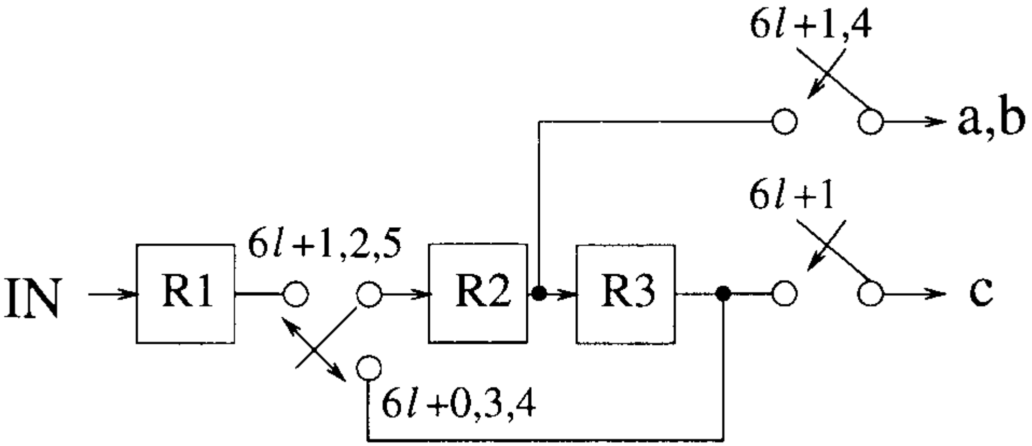


cycle	input	R1	R2	R3	output
0	a				
1	b	a			
2		b	a		
3			b	a	
4	c		a	b	a
5		c	b		
6			c	b	
7		▨	b	c	b,c

Register Minimization Techniques (8/8)



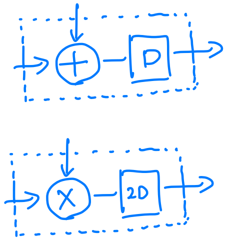
cycle	input	R1	R2	R3	output
0	a				
1	b	a			
2		b	a		
3			b	a	
4	c		a	b	a
5		c	b		
6			c	b	
7		▨	b	c	b,c

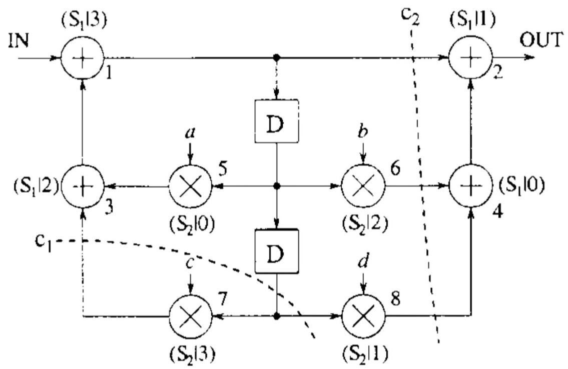


Register Minimization in Folded Architecture (1/4)

- Perform retiming for folding
- Write the folding equations
- Use the folding equations to construct a lifetime table
- Draw the lifetime chart and determine the required number of registers
- Perform forward-backward register allocation
- Draw the folded architecture that uses the minimum number of registers

Register Minimization in Folded Architecture (2/4)





- Lifetime table
  - □ T_{input} of node U is u+P_U
  - □ T_{output} of the node U is $u + P_U + \max_V\{D_F(U \to V)\}$

$$
\begin{aligned}
D_F(1 \to 2) &= 4(1) - 1 + 1 - 3 = 1 \\
D_F(1 \to 5) &= 4(1) - 1 + 0 - 3 = 0 \\
D_F(1 \to 6) &= 4(1) - 1 + 2 - 3 = 2 \\
D_F(1 \to 7) &= 4(1) - 1 + 3 - 3 = 3 \\
D_F(1 \to 8) &= 4(2) - 1 + 1 - 3 = 5 \\
D_F(3 \to 1) &= 4(0) - 1 + 3 - 2 = 0 \\
D_F(4 \to 2) &= 4(0) - 1 + 1 - 0 = 0 \\
D_F(5 \to 3) &= 4(0) - 2 + 2 - 0 = 0 \\
D_F(6 \to 4) &= 4(1) - 2 + 0 - 2 = 0 \\
D_F(7 \to 3) &= 4(1) - 2 + 2 - 3 = 1 \\
D_F(8 \to 4) &= 4(1) - 2 + 0 - 1 = 1.
\end{aligned}
$$

node	$T_{input} \to T_{output}$
1	$4 \to 9$
2	—
3	$3 \to 3$
4	$1 \to 1$
5	$2 \to 2$
6	$4 \to 4$
7	$5 \to 6$
8	$3 \to 4$

(S1|3) + ⊕-[D]- =3+1=4

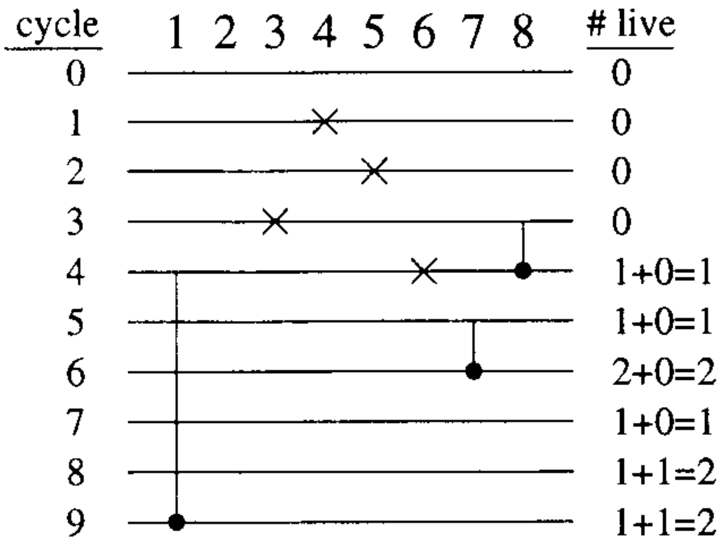
max=5

(S2|3) + ⊕-[2D]- = 3+2=5

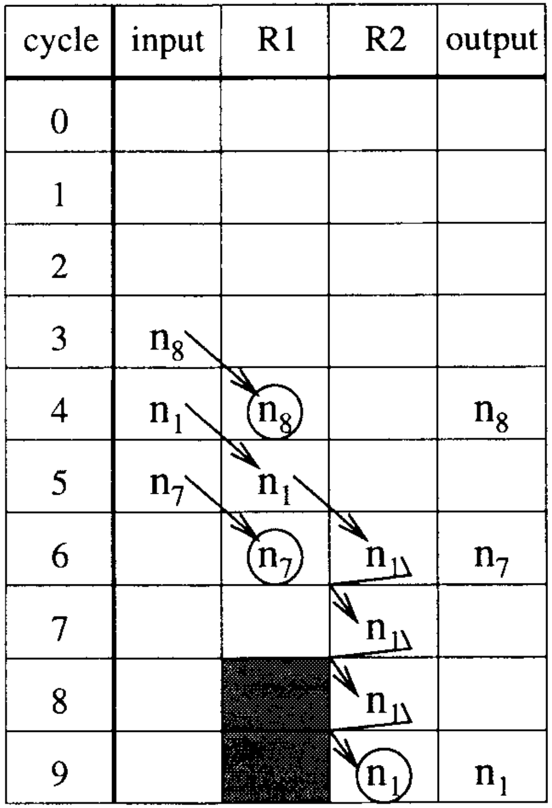
max=1

max=1

Register Minimization in Folded Architecture (3/4)



cycle	# live
0	0
1	0
2	0
3	0
4	1+0=1
5	1+0=1
6	2+0=2
7	1+0=1
8	1+1=2
9	1+1=2

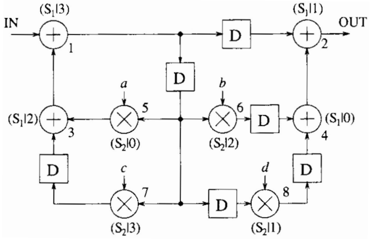


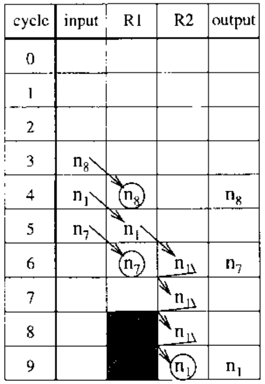
cycle	input	R1	R2	output
0				
1				
2				
3	n_8			
4	n_1	n_8		n_8
5	n_7	n_1		
6		n_7	n_1	n_7
7			n_1	
8		▨	n_1	
9		▨	n_1	n_1

Register Minimization in Folded Architecture (4/4)

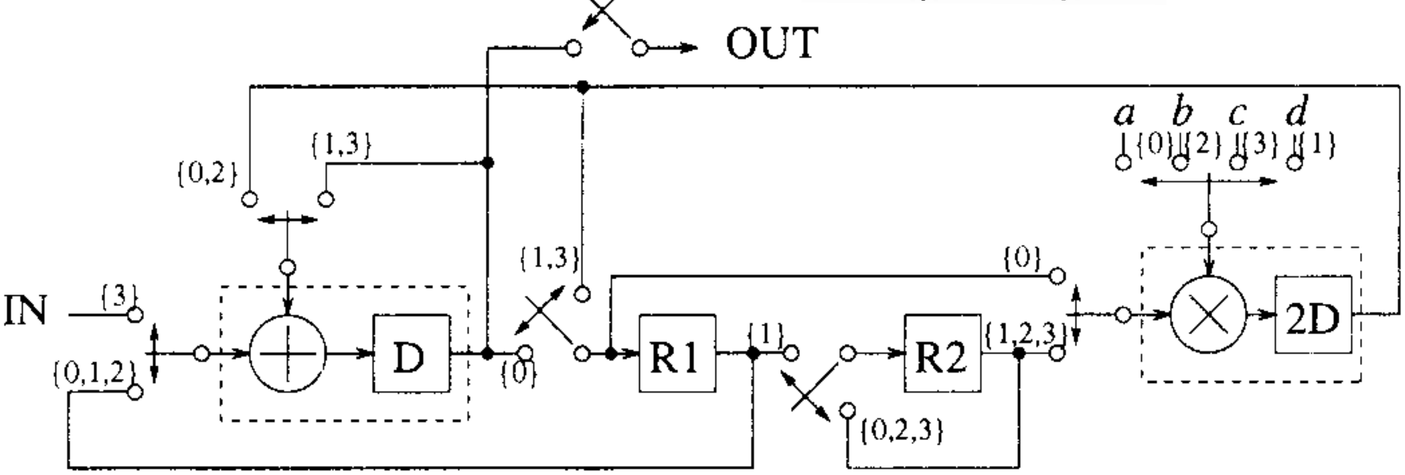
- Number of registers: 6→2

$$
\begin{aligned}
D_F(1 \to 2) &= 4(1) - 1 + 1 - 3 = 1 \\
D_F(1 \to 5) &= 4(1) - 1 + 0 - 3 = 0 \\
D_F(1 \to 6) &= 4(1) - 1 + 2 - 3 = 2 \\
D_F(1 \to 7) &= 4(1) - 1 + 3 - 3 = 3 \\
D_F(1 \to 8) &= 4(2) - 1 + 1 - 3 = 5 \\
D_F(3 \to 1) &= 4(0) - 1 + 3 - 2 = 0 \\
D_F(4 \to 2) &= 4(0) - 1 + 1 - 0 = 0 \\
D_F(5 \to 3) &= 4(0) - 2 + 2 - 0 = 0 \\
D_F(6 \to 4) &= 4(1) - 2 + 0 - 2 = 0 \\
D_F(7 \to 3) &= 4(1) - 2 + 2 - 3 = 1 \\
D_F(8 \to 4) &= 4(1) - 2 + 0 - 1 = 1.
\end{aligned}
$$

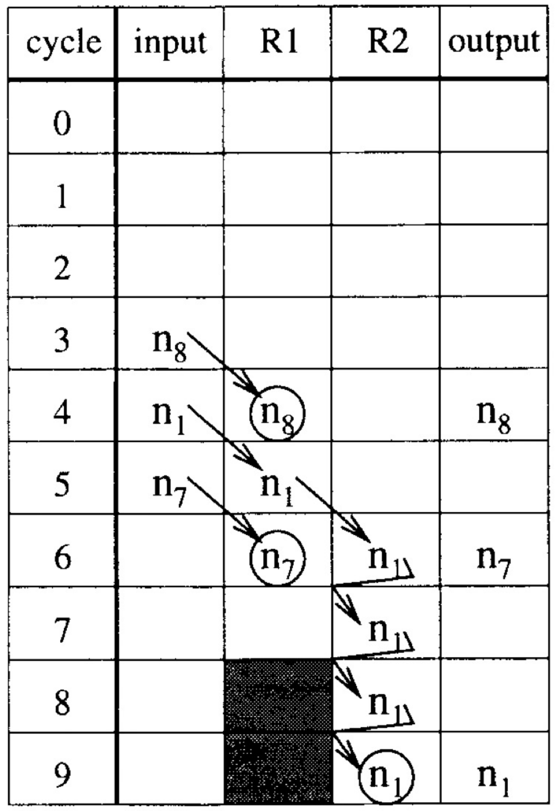




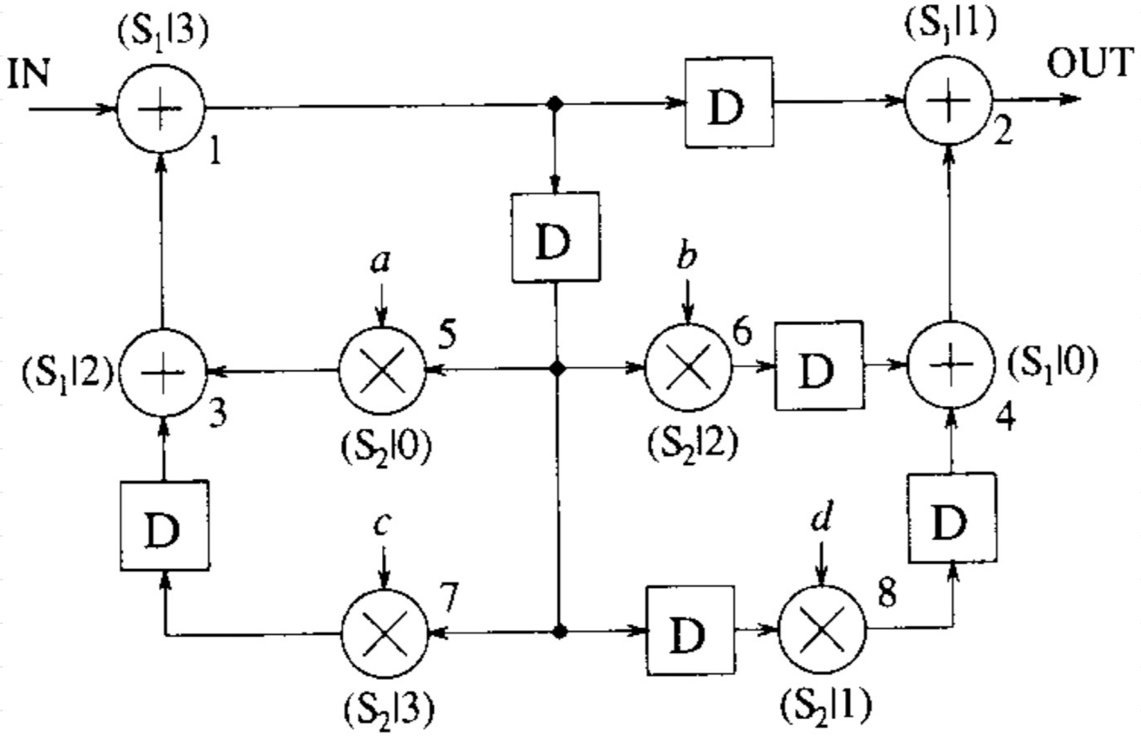
cycle	input	R1	R2	output
0				
1				
2				
3	n_8			
4	n_1	n_8		n_8
5	n_7	n_1		
6		n_7	n_1	n_7
7			n_1	
8			n_1	
9			n_1	n_1



{p,q} denotes 4*l* +p and 4*l* +q



cycle	input	R1	R2	output
0				
1				
2				
3	n_8			
4	n_1	n_8		n_8
5	n_7	n_1		
6		n_7	n_1	n_7
7			n_1	
8			n_1	
9			n_1	n_1



~~$D_F(1 \to 2) = 4(1) - 1 + 1 - 3 = 1$~~

~~$D_F(1 \to 5) = 4(1) - 1 + 0 - 3 = 0$~~

~~$D_F(1 \to 6) = 4(1) - 1 + 2 - 3 = 2$~~

~~$D_F(1 \to 7) = 4(1) - 1 + 3 - 3 = 3$~~

~~$D_F(1 \to 8) = 4(2) - 1 + 1 - 3 = 5$~~

~~$D_F(3 \to 1) = 4(0) - 1 + 3 - 2 = 0$~~

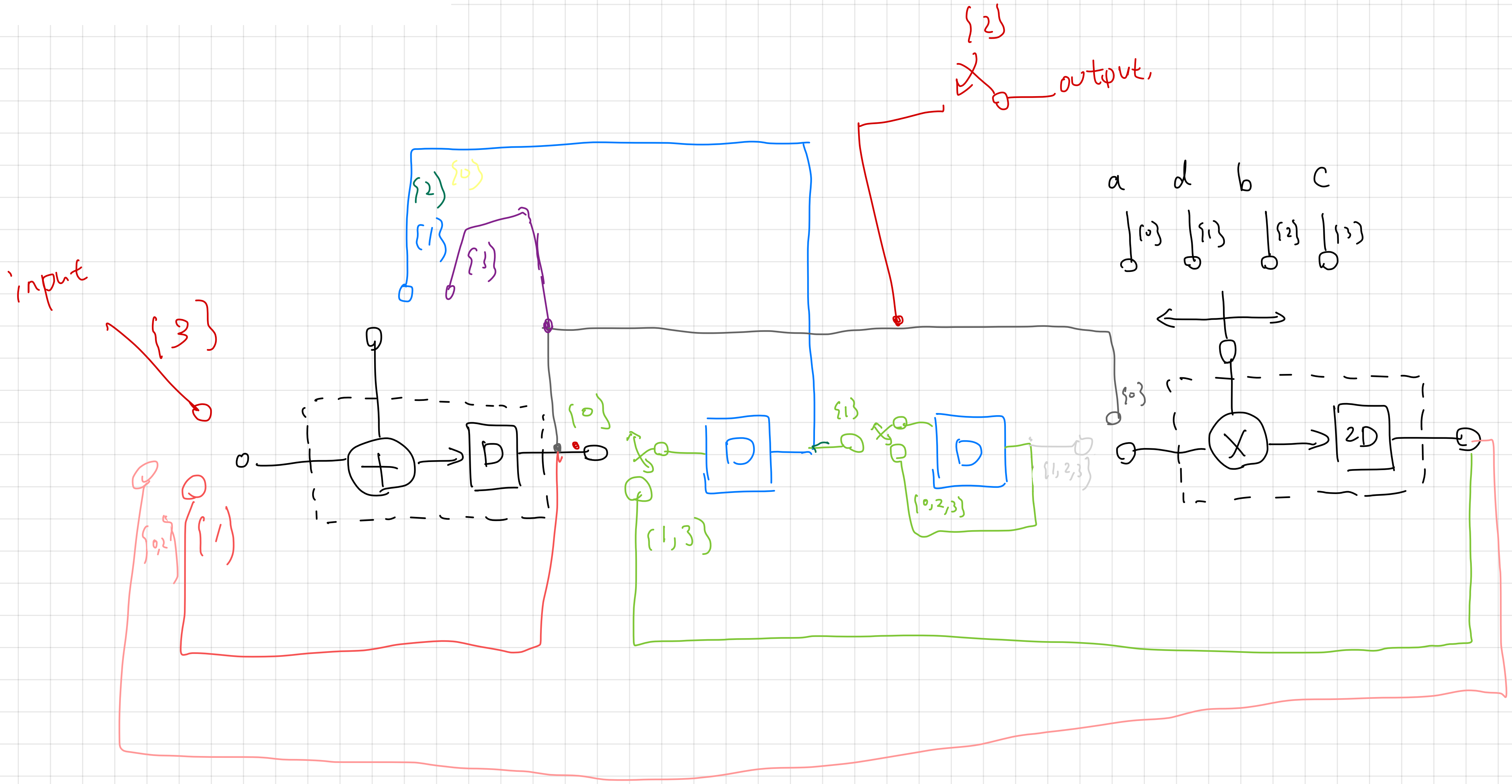
~~$D_F(4 \to 2) = 4(0) - 1 + 1 - 0 = 0$~~

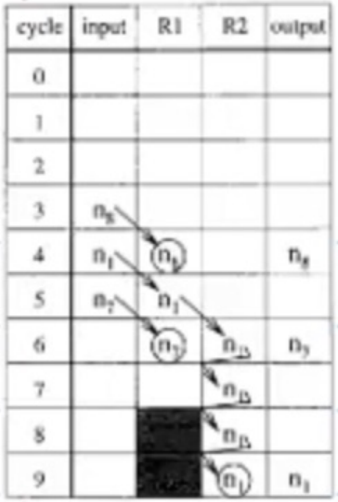
~~$D_F(5 \to 3) = 4(0) - 2 + 2 - 0 = 0$~~

~~$D_F(6 \to 4) = 4(1) - 2 + 0 - 2 = 0$~~

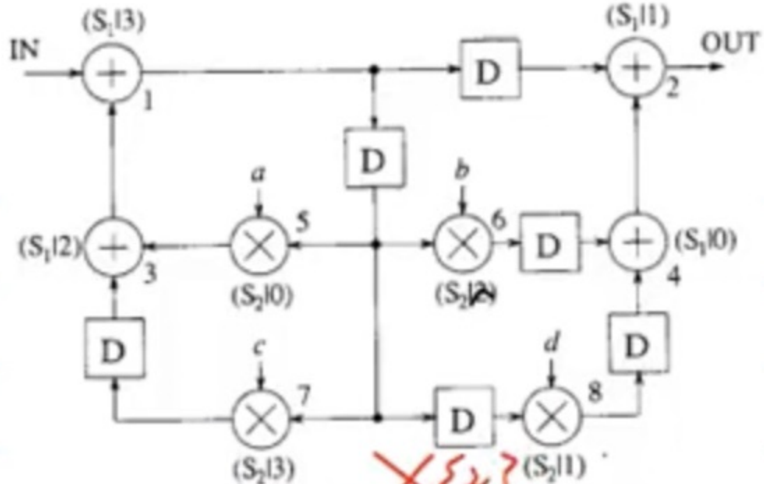
~~$D_F(7 \to 3) = 4(1) - 2 + 2 - 3 = 1$~~

~~$D_F(8 \to 4) = 4(1) - 2 + 0 - 1 = 1.$~~

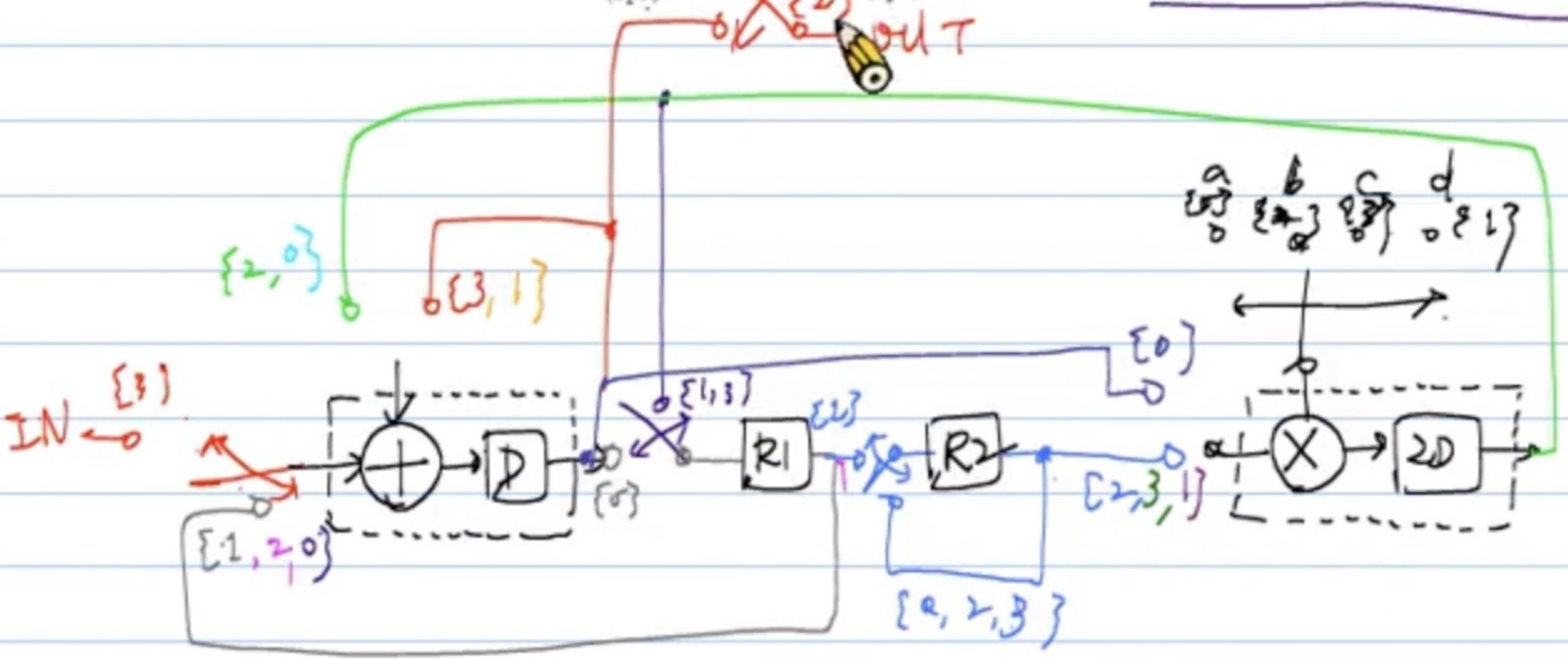




cycle	input	R1	R2	output
0				
1				
2				
3	n_8			
4	n_1	n_8		n_8
5	n_7	n_1		
6		n_7	n_{13}	n_7
7			n_{13}	
8		■	n_{13}	
9		■	n_1	n_1



$$D_F(1 \to 2) = 4(1) - 1 + 1 - 3 = 1$$
$$D_F(1 \to 5) = 4(1) - 1 + 0 - 3 = 0$$
$$D_F(1 \to 6) = 4(1) - 1 + 2 - 3 = 2$$
$$D_F(1 \to 7) = 4(1) - 1 + 3 - 3 = 3$$
$$D_F(1 \to 8) = 4(2) - 1 + 1 - 3 = 5$$
$$D_F(3 \to 1) = 4(0) - 1 + 3 - 2 = 0$$
$$D_F(4 \to 2) = 4(0) - 1 + 1 - 0 = 0$$
$$D_F(5 \to 3) = 4(0) - 2 + 2 - 0 = 0$$
$$D_F(6 \to 4) = 4(1) - 2 + 0 - 2 = 0$$
$$D_F(7 \to 3) = 4(1) - 2 + 2 - 3 = 1$$
$$D_F(8 \to 4) = 4(1) - 2 + 0 - 1 = 1.$$


{2}
OUT
{2, 0}
{3, 1}
a b c d
{3} {2} {3} {1}
{0}
IN
{3}
{1, 3}
{2}
D
R1
R2
{0}
2D
{2, 3, 1}
{1, 2, 0}
{0, 2, 3}