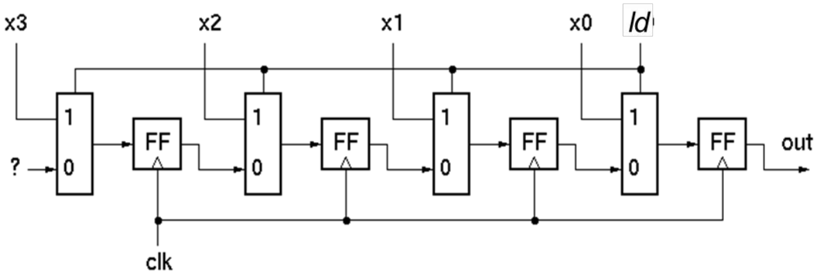
EECS 151 Disc 3

Rahul Kumar (session 1)
Yukio Miyasaka (session 2)

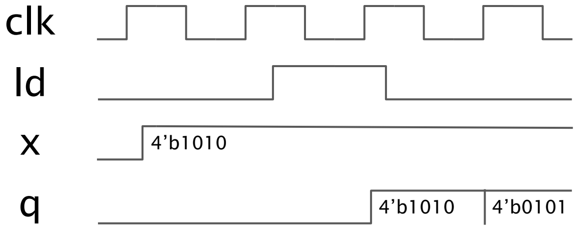
Berkeley
UNIVERSITY OF CALIFORNIA

Contents

Parallel to Serial Converter

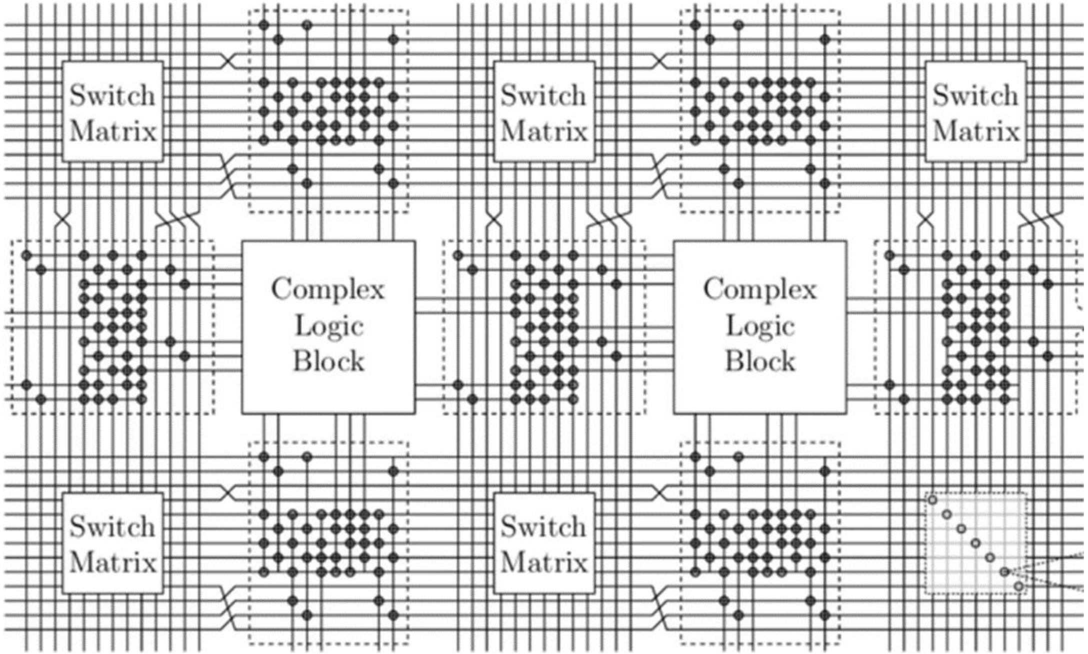


Reads input on a positive clock edge if ld is 1

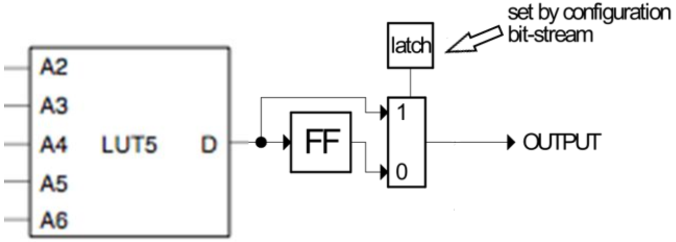



```
module ParToSer (ld, x, out, clk);
  input [3:0] x;
  input ld, clk;
  output out;
  wire [3:0] q, d;

  REGISTER #(.N(4))
      r(.q(q), .d(d), .clk(clk));
  assign d = ld? x: {Q[0], Q[3:1]};
  assign out = q[0];

endmodule
```

FPGA

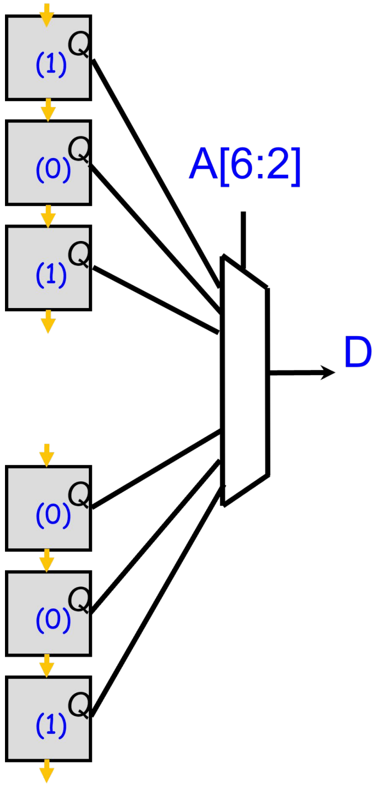

Switch Matrix
Switch Matrix
Switch Matrix
Complex Logic Block
Complex Logic Block
Switch Matrix
Switch Matrix

Logic Block



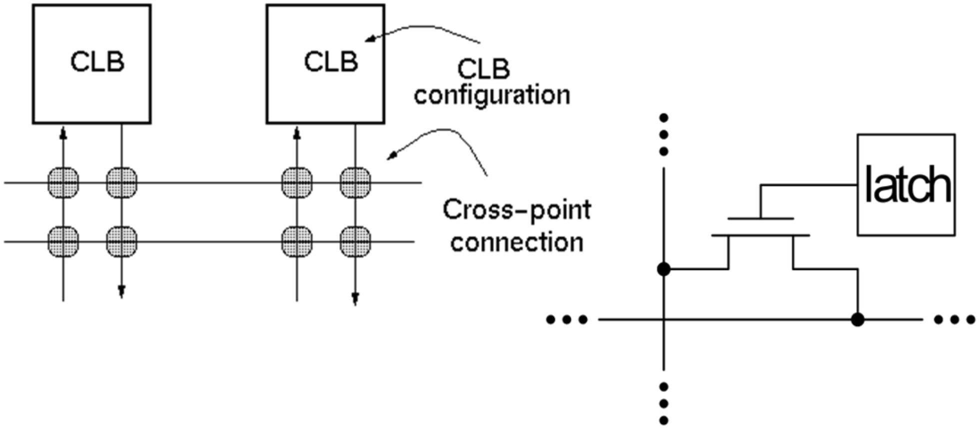
- Contains LUTs and FFs
- FFs are attached at the output of LUTs
- Configuration bits are stored in shift register
- In this case, registers are chained in the ascending order of input

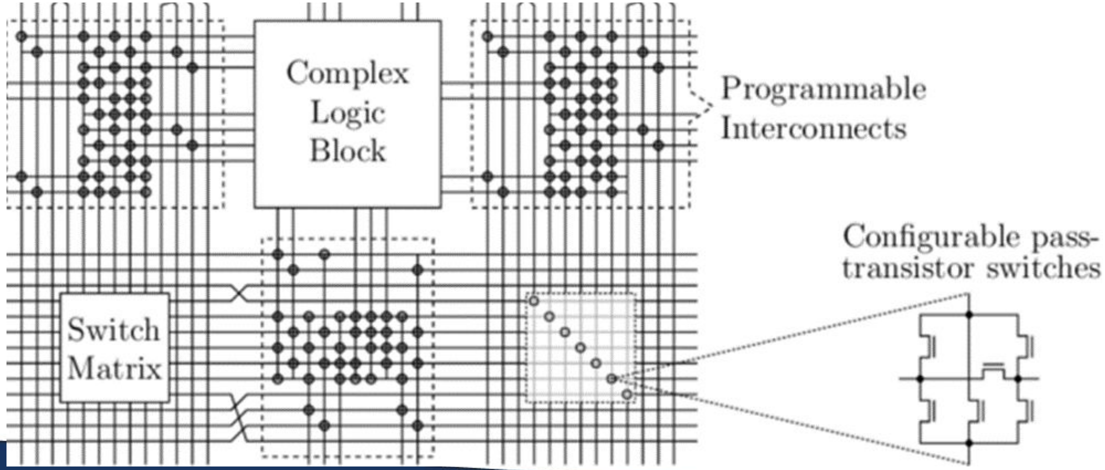
A6	A5	A4	A3	A2	D
0	0	0	0	0	1
0	0	0	0	1	0
0	0	0	1	0	1
		…			
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	1



Routing Switches

- Logic block input/output is connected to wires through a single transistor
- Makes a turn in switch matrix
- No matter if it makes a turn or not, a signal passes through one transistor in switch matrix
- Some FPGAs have *double lines* which skip every other switch matrix

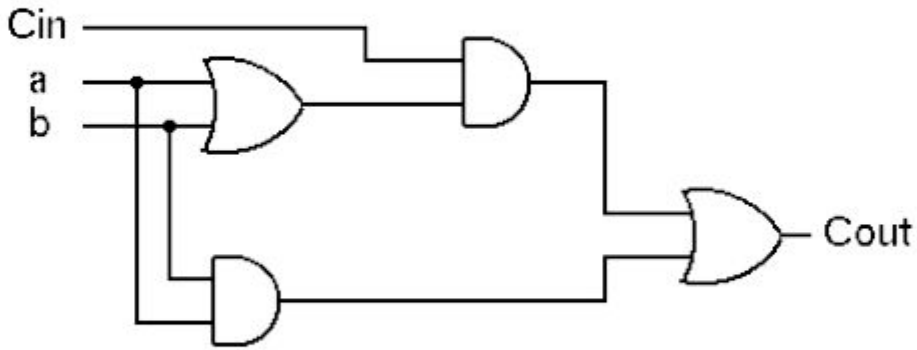




Berkeley
UNIVERSITY OF CALIFORNIA

Critical Path Delay

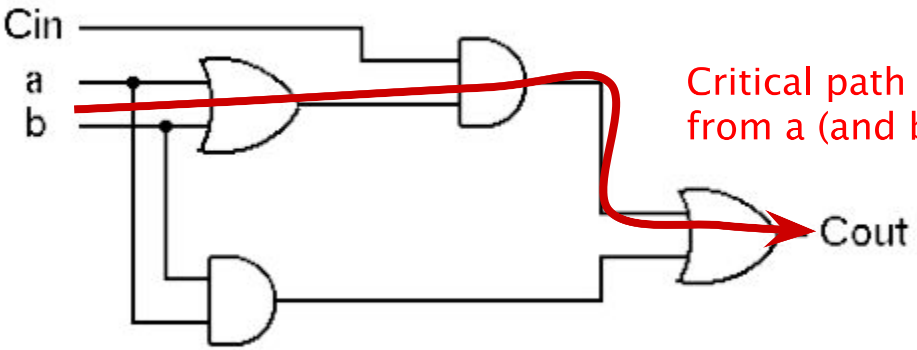
- Assume every gate has same delay for now
- The delay of the slowest path is called *critical path delay*



What is critical path delay in this circuit?

Critical Path Delay

- Assume every gate has same delay for now
- The delay of the slowest path is called *critical path delay*



Critical path delay is 3
from a (and b) to Cout

What is critical path delay in this circuit?

Boolean Algebra

Postulates

- a + 0 = a (a * 1 = a)
- a + a’ = 1 (a * a’ = 0)
- Commutative law ab = ba
- Distributive law a(b + c) = ab + bc, a+(bc) = (a + b)(a + c)

Theorems

- a + a = a (a * a = a)
- a + 1 = 1 (a * 0 = 0)
- (a’)’ = a
- Associative law a(bc) = (ab)c

Practice

$x + xy = ?$

Practice

$x + xy = x * 1 + xy$

Practice

$x + xy = x * 1 + xy = x(1 + y) = x * 1 = x$

Hard to find the first step

Massage it by generating 1 (or 0) randomly

K-map

- Simple way to crate a small SOP
- Fill cells according to the truth table
- Group as many ones as possible
- Groups can overlap
- Extract products
- Sum them up

a	b	c	d	x
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
		…		
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

		cd			
		00	01	11	10
ab	00	0	0	0	0
	01	0	0	0	1
	11	1	0	0	1
	10	1	0	0	1

K-map

Group size must be (2^n, 2^m)

- Simple way to crate a small SOP
- Fill cells according to the truth table
- Group as many ones as possible
- Groups can overlap
- Extract products
- Sum them up

a	b	c	d	x
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
		…		
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

cd

ab	00	01	11	10
00	0	0	0	0
01	0	0	0	1
11	1	0	0	1
10	1	0	0	1

K-map

- Simple way to crate a small SOP
- Fill cells according to the truth table
- Group as many ones as possible
- Groups can overlap
- Extract products
- Sum them up

Group size must be (2^n, 2^m)

a	b	c	d	x
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
		…		
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

cd

ab	00	01	11	10
00	0	0	0	0
01	0	0	0	1
11	1	0	0	1
10	1	0	0	1

bcd’

ad’

= ad’ + bcd’

K-map with Don’t-cares

- Some functions have limited input space
- They can output arbitrary values outside

a	b	c	d	x
0	0	0	0	0
0	0	0	1	-
0	0	1	0	-
		…		
1	1	0	1	1
1	1	1	0	1
1	1	1	1	-

cd

ab	00	01	11	10
00	0	-	-	-
01	0	-	-	-
11	-	1	-	1
10	0	-	-	-

K-map with Don't-cares

cd

ab

	00	01	11	10
00	0	-	-	-
01	0	-	-	-
11	-	1	-	1
10	0	-	-	-

cd

ab

	00	01	11	10
00	0	-	-	-
01	0	-	-	-
11	-	1	-	1
10	0	-	-	-

Which is better?

K-map with Don't-cares

cd

ab	00	01	11	10
00	0	-	-	-
01	0	-	-	-
11	-	1	-	1
10	0	-	-	-

ab

cd

ab	00	01	11	10
00	0	-	-	-
01	0	-	-	-
11	-	1	-	1
10	0	-	-	-

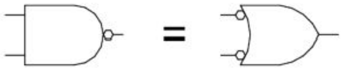
c + d

Which is better? - Depends on implementation.

DeMorgan's Law

(a + b)’ = a’b’

(ab)’ = a’ + b’



“Bubble pushing”: flip the gate, propagate the bubbles.

We can freely generate two consecutive bubbles anywhere.