The Abdus Salam
International Centre for Theoretical Physics

United Nations
Educational, Scientific
and Cultural Organization

International Atomic
Energy Agency

310/1780-6

ICTP-INFN Advanced Tranining Course on
FPGA and VHDL for Hardware Simulation and Synthesis
27 November - 22 December 2006

DIGITAL DESIGN 6

Pirouz BAZARGAN SABET
Lip 6
University Pierre et Marie Curie (VI)
Department ASIM
4, place Jussieu
75252 Paris Cedex 05
FRANCE

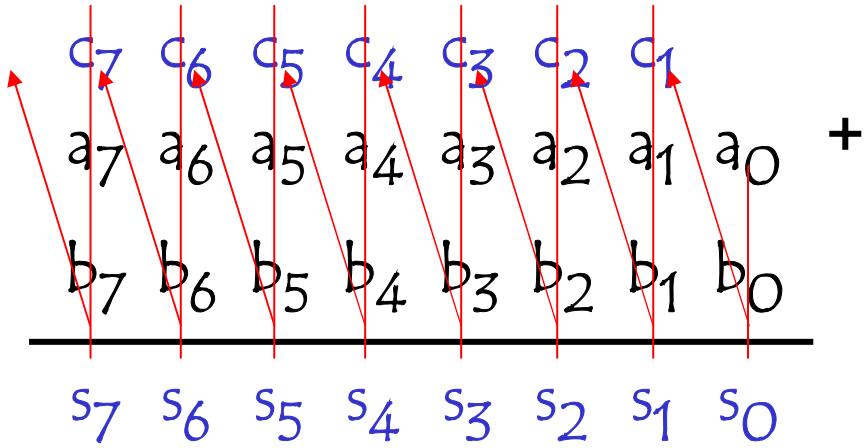
Outline

- Digital CMOS Design
- Arithmetic Operators
  - Adders
  - Comparators
  - Shifters

Adders

Adding two natural numbers

Let consider two natural numbers a and b
coded on 8 bits using Natural Binary Code

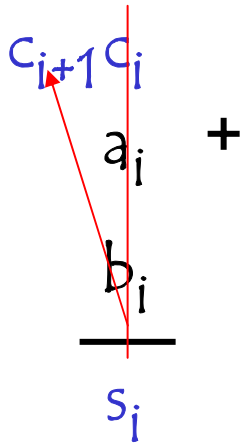


Adders

Adding two natural numbers

At each stage, I need to sum 3 single bit numbers a_i b_i c_i

The carry out of the stage i is the input carry of the next stage



s_i and c_{i+1} are Boolean functions of a_i b_i c_i

Adders

Adding two natural numbers

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

$a_i\ b_i$

c_i

s_i

$s_i = a_i \oplus b_i \oplus c_i$

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

$a_i\ b_i$

c_i

c_{i+1}

$c_{i+1} = a_i.b_i + a_i.c_i + b_i.c_i$

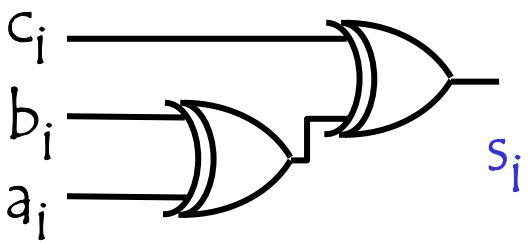
Adders

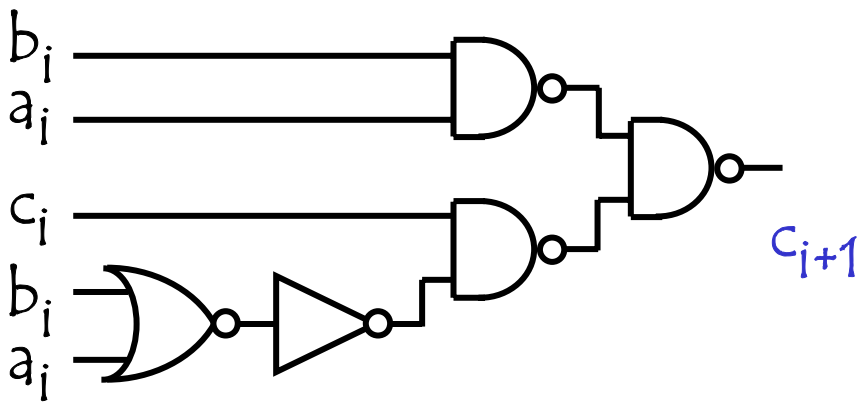
Adding two natural numbers

$s_i = a_i \oplus b_i \oplus c_i$

$c_{i+1} = a_i.b_i + a_i.c_i + b_i.c_i$

$c_{i+1} = a_i.b_i + (a_i+b_i) . c_i$





Adders

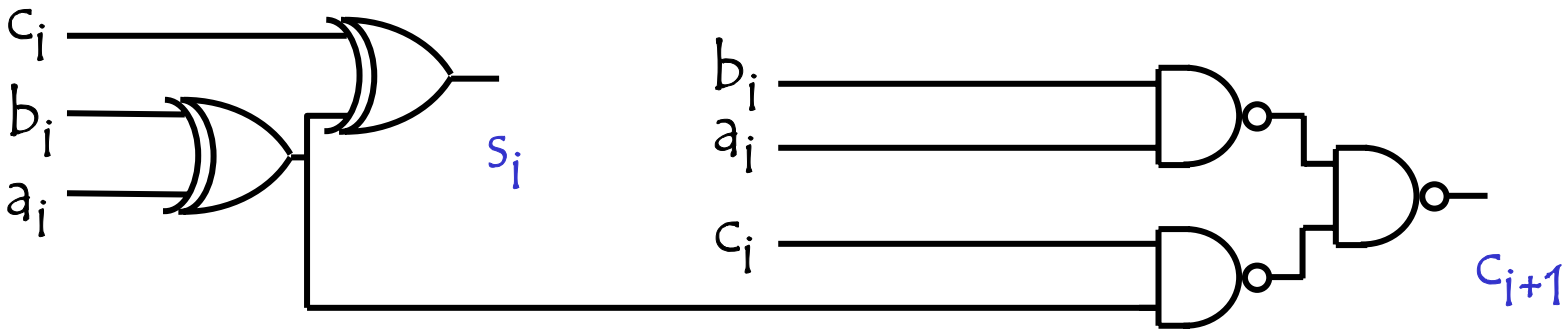
Adding two natural numbers

$s_i = a_i \oplus b_i \oplus c_i$

$c_{i+1} = a_i.b_i + a_i.c_i + b_i.c_i$

$c_{i+1} = a_i.b_i + (a_i+b_i) . c_i$

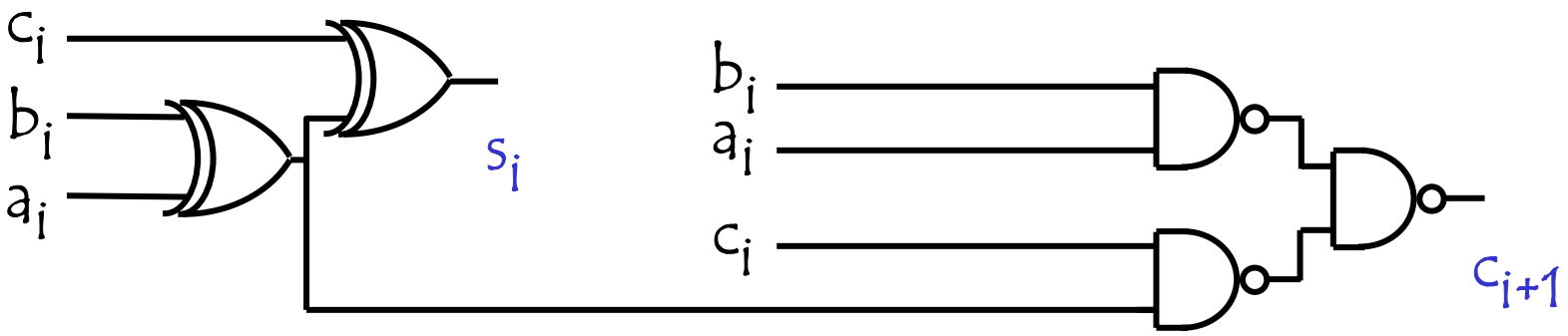
$c_{i+1} = a_i.b_i + (a_i \oplus b_i) . c_i$



Adders

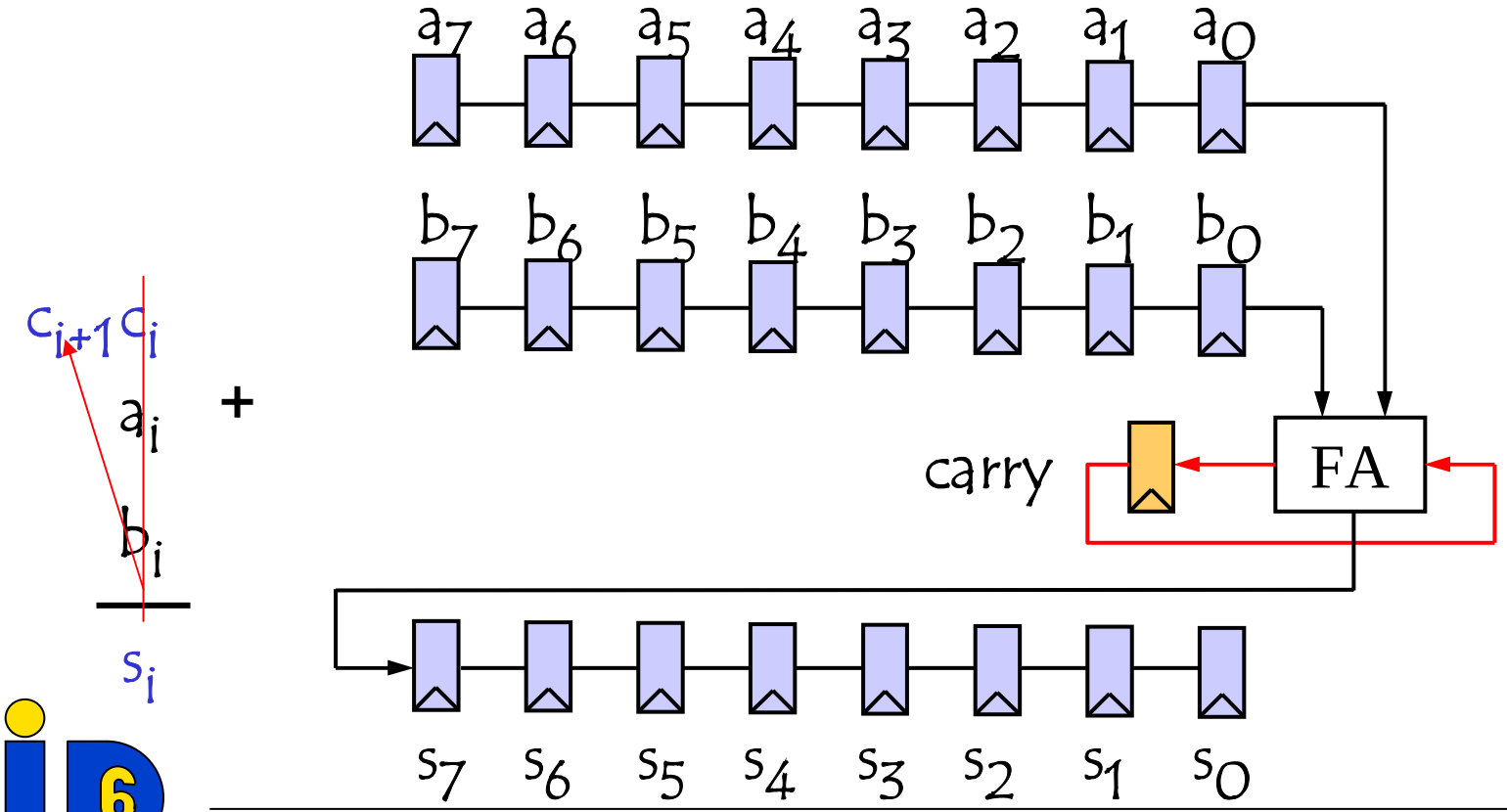
Adding two natural numbers

The circuit generating s_i and c_{i+1} is called a Full Adder (FA)



Adders

Adding two natural numbers



Adders

Adding two natural numbers

Sequential Adder

Area $\propto$ n Delay $\propto$ n cycles

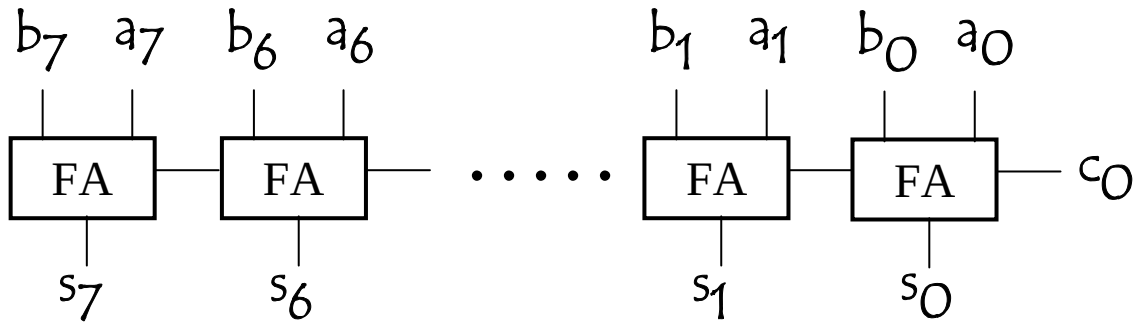
Timing should be improved

Adders

Adding two natural numbers

At each stage, I need to sum 3 single bit numbers a_i b_i c_i

The carry out of the stage i is the input carry of the next stage



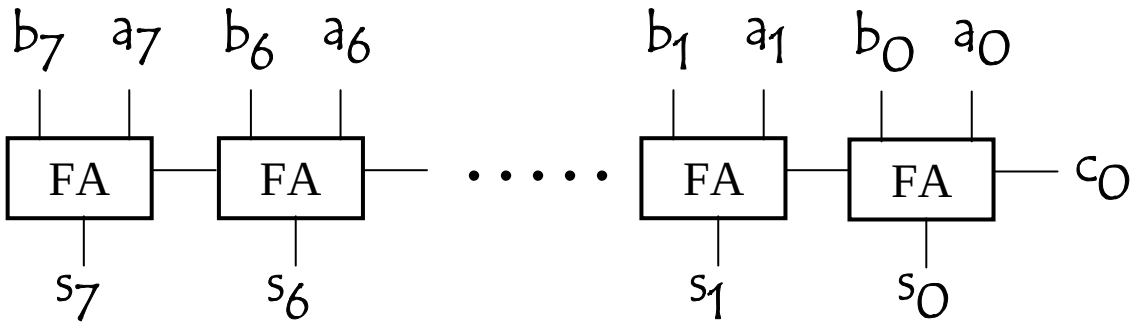
Carry Propagate Adder (CPA)

Adders

Adding two natural numbers

Carry Propagate Adder (CPA)

Area $\propto$ n Delay $\propto$ n

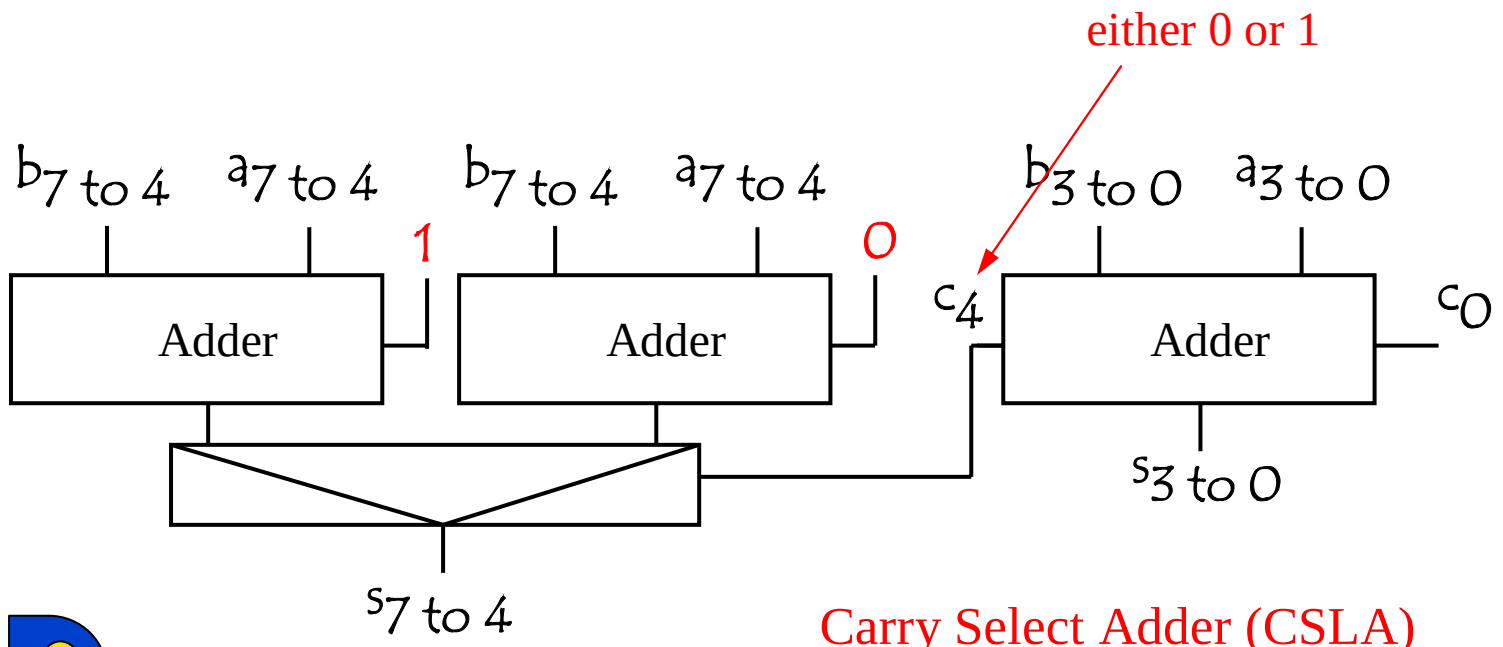


Timing should be improved

Adders

Adding two natural numbers

Acceleration technics



Carry Select Adder (CSLA)

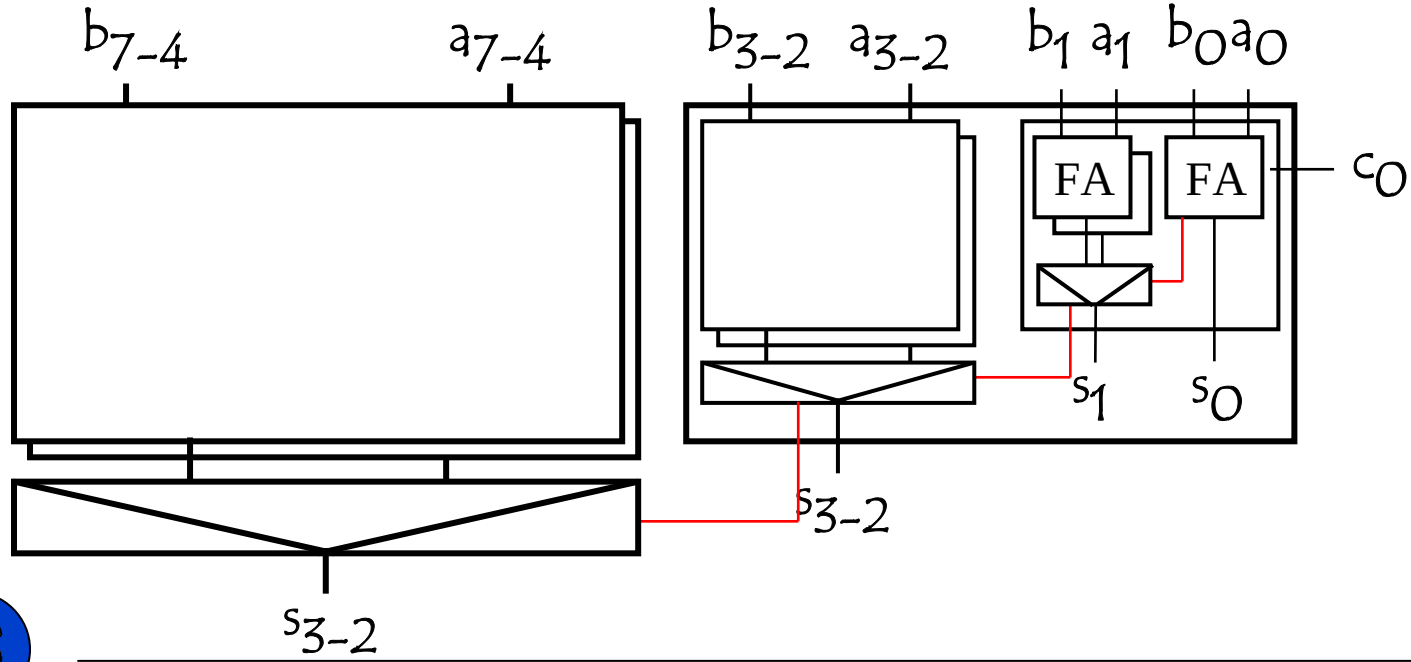
Adders

Adding two natural numbers

Carry Select Adder (CSLA)

Area $\propto n^{log(3)} = n^{1.585}$

Delay $\propto log\ (n)$



Adders

Adding two natural numbers

Acceleration technics

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

$a_i\ b_i$

c_i

c_{i+1}

carry out depends on carry in

carry out does not depend on carry in

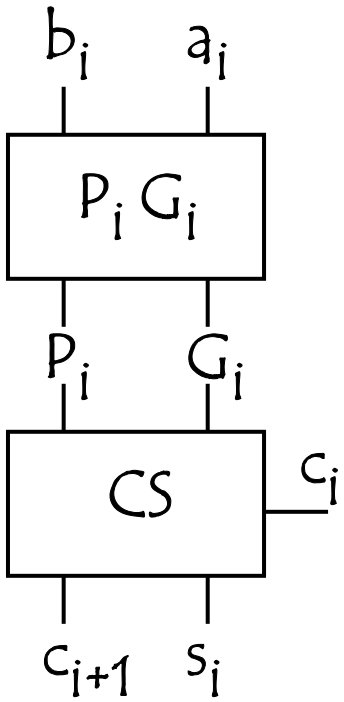
00	01	11	10
absorbtion	propagation	generation	propagation

$a_i\ b_i$

Adders

Adding two natural numbers

Acceleration technics



$$G_i = a_i\, b_i$$

$$P_i = a_i \oplus b_i$$

$$c_{i+1} = G_i + P_i\, c_i$$

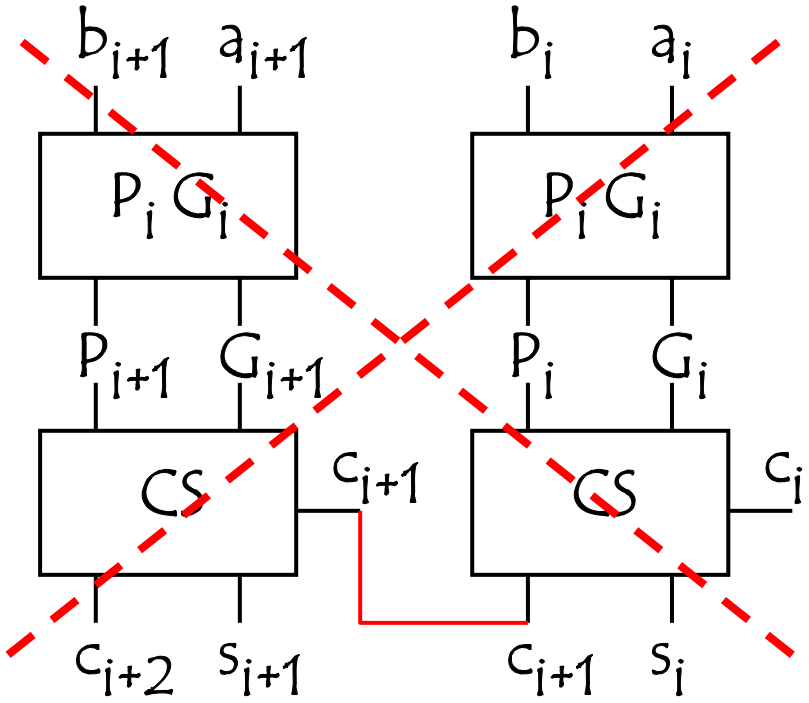
$$s_i = P_i \oplus c_i$$

a_i b_i	00	01	11	10
	absorbtion	propagation	generation	propagation

Adders

Adding two natural numbers

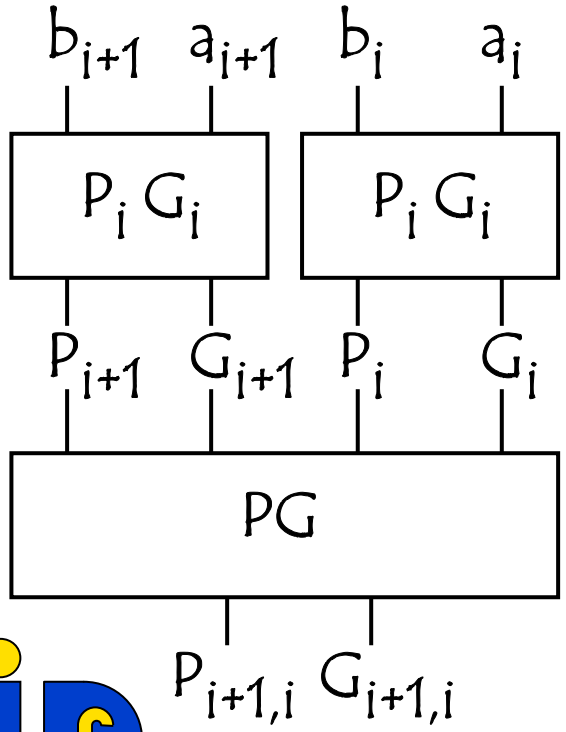
Acceleration technics



Adders

Adding two natural numbers

Acceleration technics



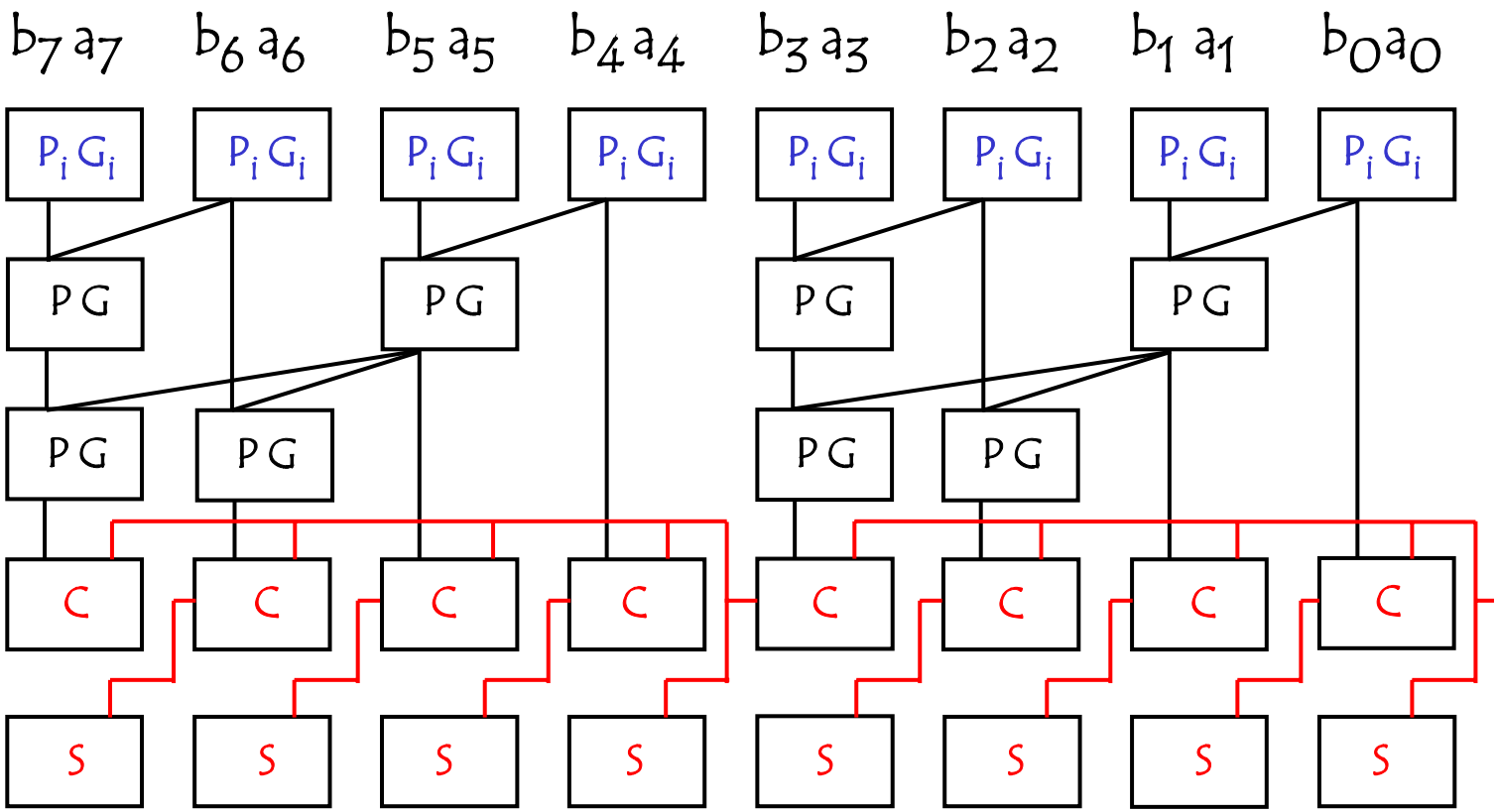
$$G_{i+1} = a_{i+1}\, b_{i+1} \qquad G_i = a_i\, b_i$$

$$P_{i+1} = a_{i+1} \oplus b_{i+1} \qquad P_i = a_i \oplus b_i$$

$$G_{i+1,\,i} = G_{i+1} + G_i . P_{i+1}$$

$$P_{i+1,\,i} = P_i \cdot P_{i+1}$$

Adders



Adders

Adding two natural numbers (summary)

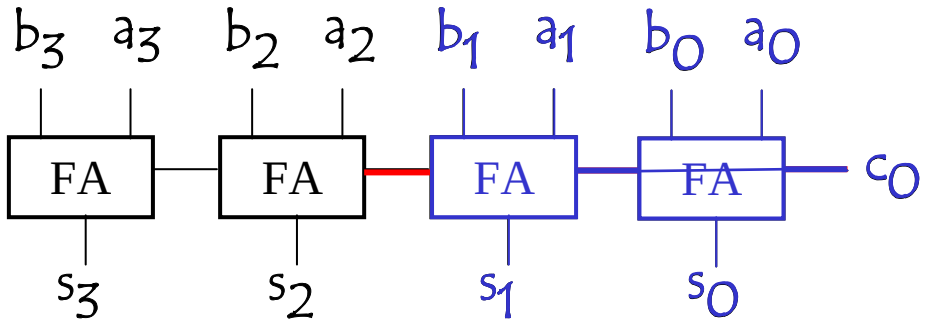
	Area	Delay
Carry Propagate (CPA)	n	n
Carry Select (CSLA)	n *log*(3)	*log* (n)
Carry Lookahead (CLA)	n *log* (n)	*log* (n)
Magic Adder	n	Cste

More improvements ?

Adders

Adding two natural numbers

Carry Propagate Adder (CPA)

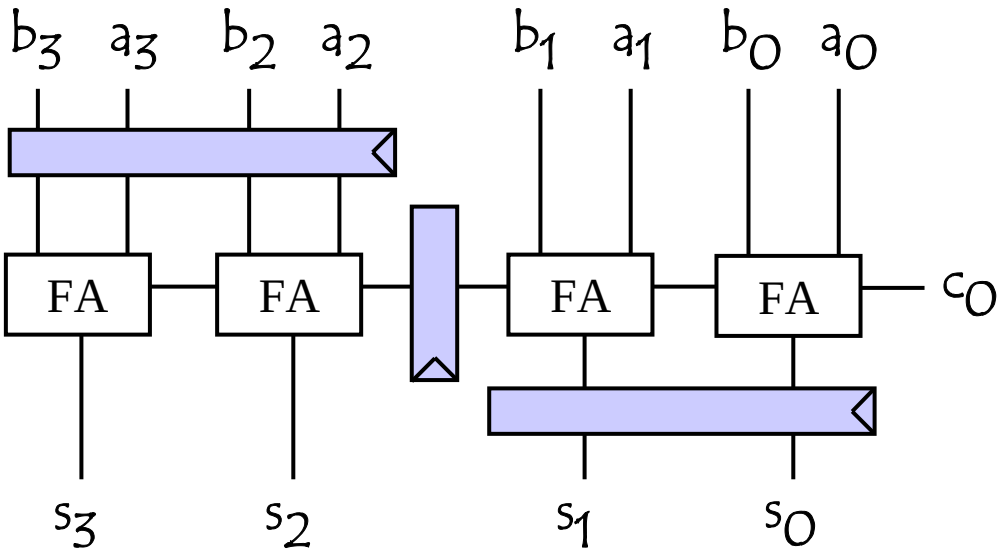


Adders

Adding two natural numbers

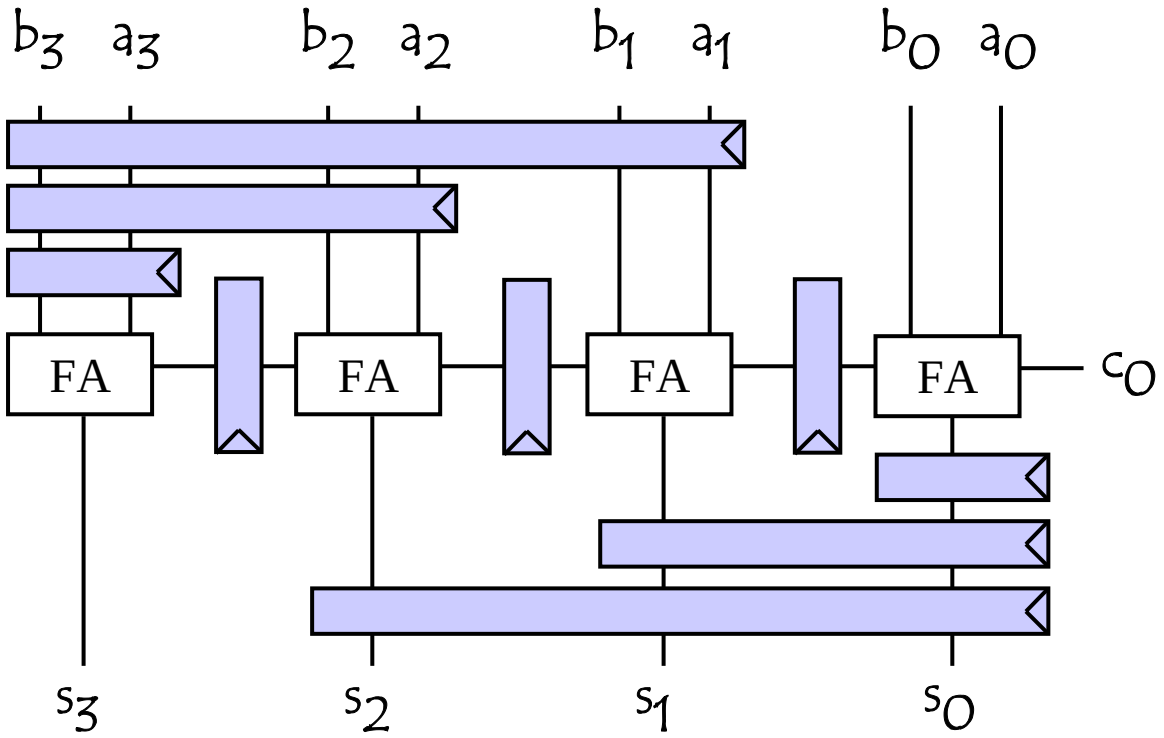
Carry Propagate Adder (CPA)

pipelining



Adders

Adding two natural numbers



Adders

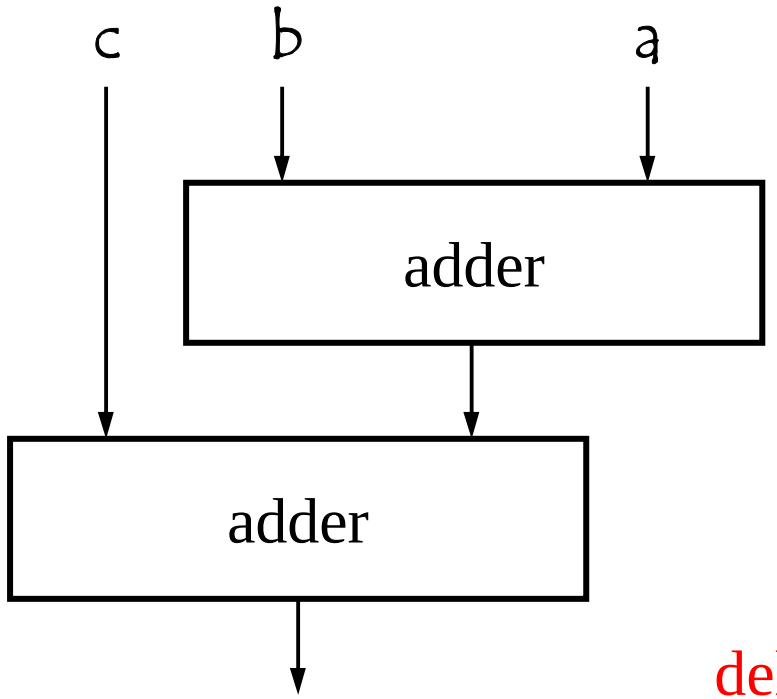
Adding two natural numbers (summary)

	Area	Delay
Carry Propagate (CPA)	n	n
Carry Select (CSLA)	$n^{log(3)}$	*log* (n)
Carry Lookahead (CLA)	n *log* (n)	*log* (n)
	n^2	Cste (1 cycle)
Magic Adder	n	Cste

When there is no door to escape break the wall

Adders

Adding three natural numbers



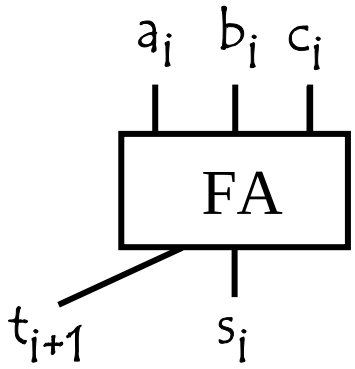
delay

Adders

Adding three natural numbers

$s_i = a_i \oplus b_i \oplus c_i$ $\quad$ $c_{i+1} = a_i.b_i + a_i.c_i + b_i.c_i$

the expressions are symetrical in regard of a, b and c

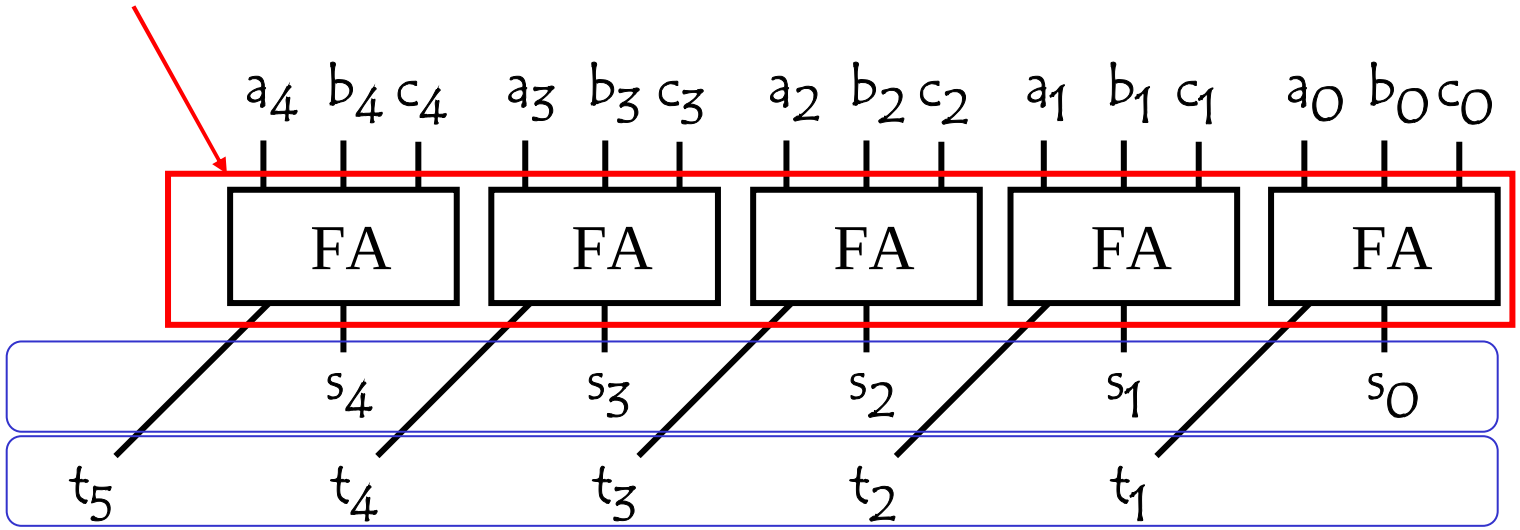


A full adder creates 2 numbers from 3

Adders

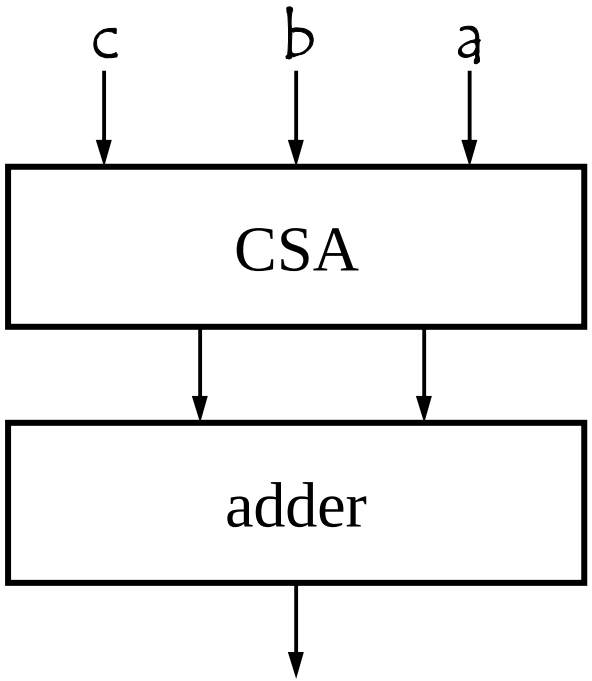
Adding three natural numbers

Carry Save Adder (CSA)



Adders

Adding three natural numbers



Delay = cste

Area $\propto$ n

Adders

Adding two natural numbers

Changing the representation of numbers

Given a natural number a : a is coded using 2n bits

a = a0 + a1 Redondante Binary Code

Example : the number 5 can be coded on 4 bits as

0000 + 0101
0001 + 0100
0100 + 0011

Adders

Adding two natural numbers

Changing the representation of numbers

$$a = a0 + a1$$

$$b = b0 + b1$$

Adding a and b in Redondante Binary Code is finding c

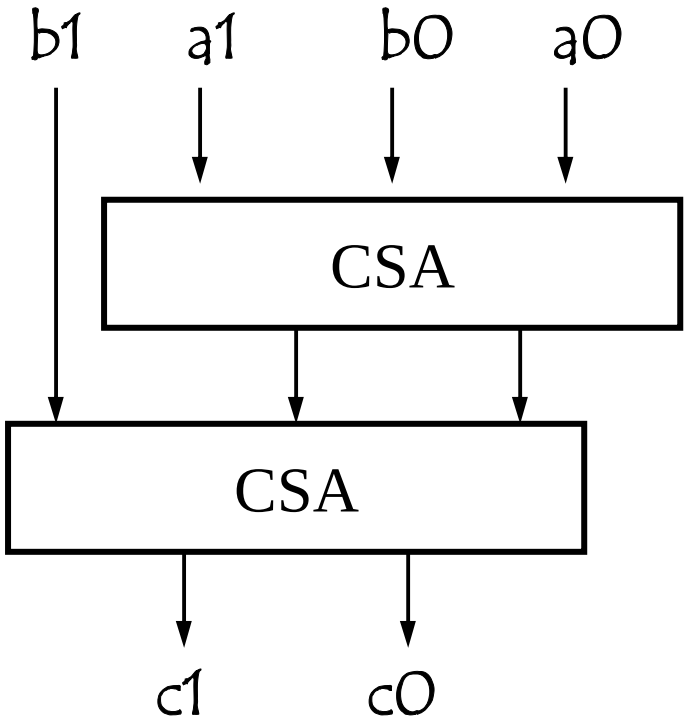
$$c = c0 + c1 \text{ such as}$$

$$c0 + c1 = a0 + a1 + b0 + b1$$

Adding 4 numbers to generate 2

Adders

Adding two natural numbers



Delay = cste

Area $\propto$ n

Outline

- Digital CMOS Design
- Arithmetic Operators
  - Adders
  - Comparators
  - Shifters

Comparators

Comparing a natural number to a constante

Let consider a natural number a coded on 8 bits using Natural Binary Code

$a_7\ a_6\ a_5\ a_4\ a_3\ a_2\ a_1\ a_0$

$= ?$

$0\ 0\ 0\ 0\ 0\ 0\ 0\ 0$

0 / 1

Comparators

Comparing a natural number

Boolean function

Null = 1 if

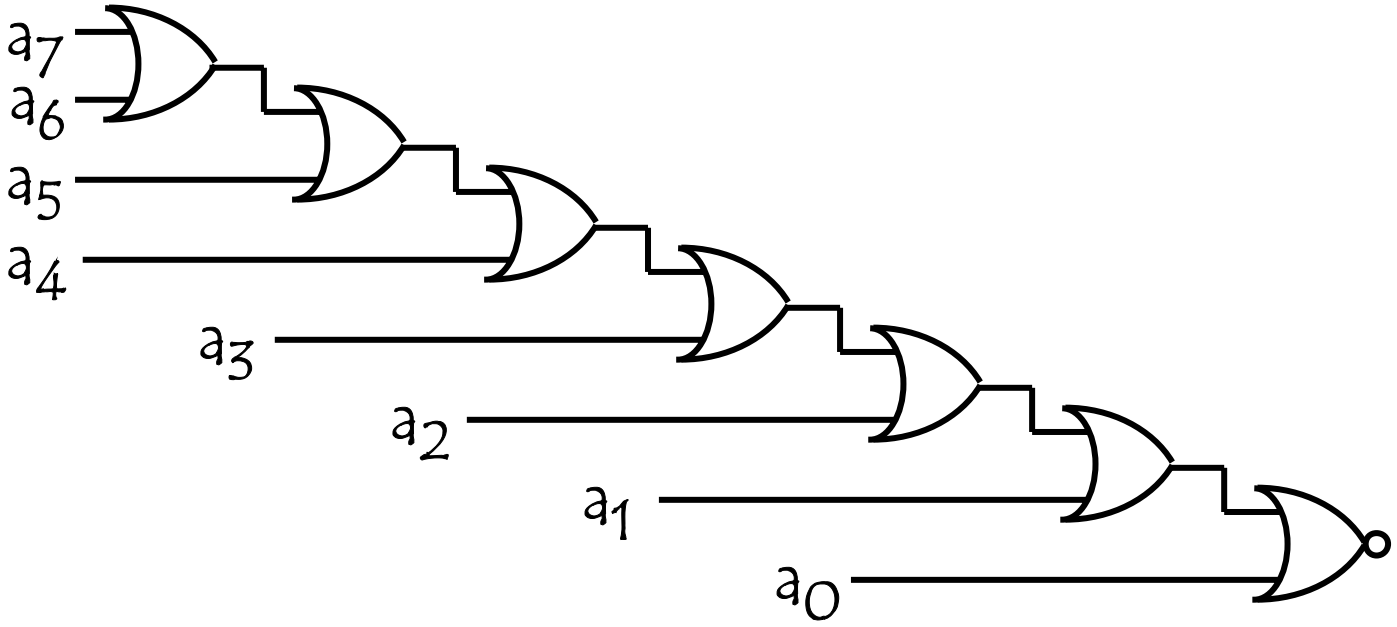
$$\overline{a_7} \cdot \overline{a_6} \cdot \overline{a_5} \cdot \overline{a_4} \cdot \overline{a_3} \cdot \overline{a_2} \cdot \overline{a_1} \cdot \overline{a_0} = 1$$

$$\text{Null} = a_7 + a_6 + a_5 + a_4 + a_3 + a_2 + a_1 + a_0$$

Comparators

Comparing a natural number

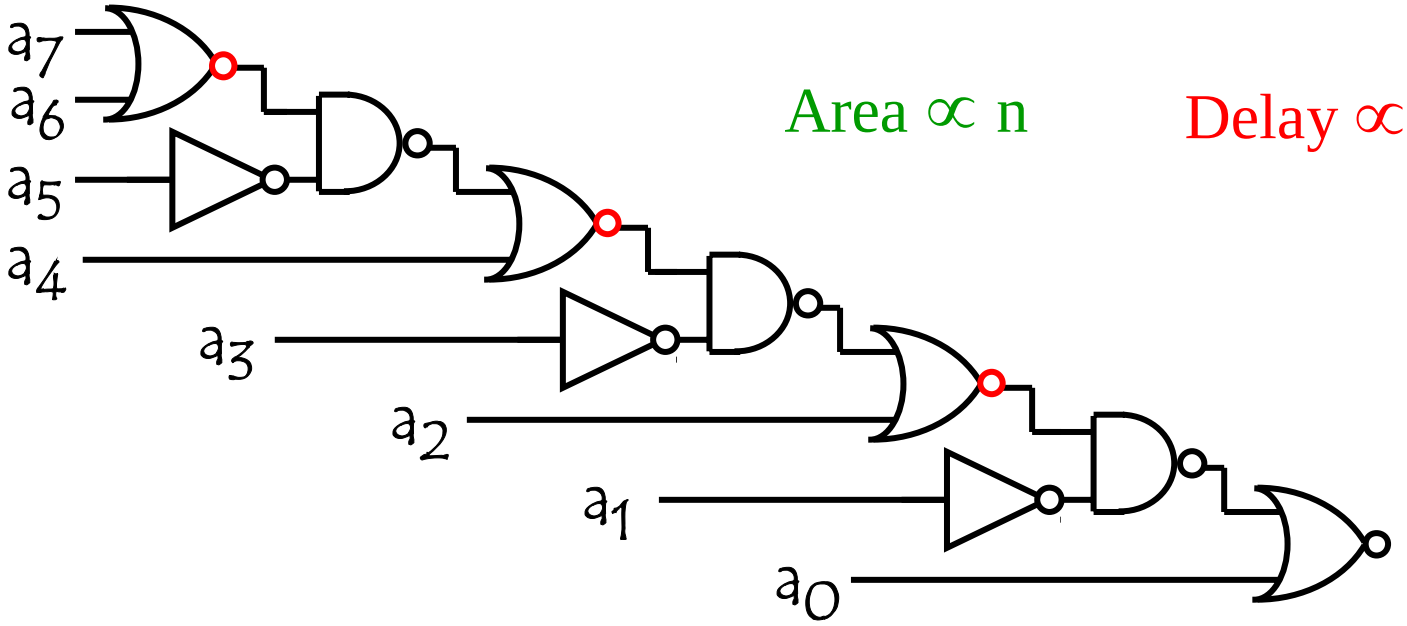
Implementation



Comparators

Comparing a natural number

Implementation



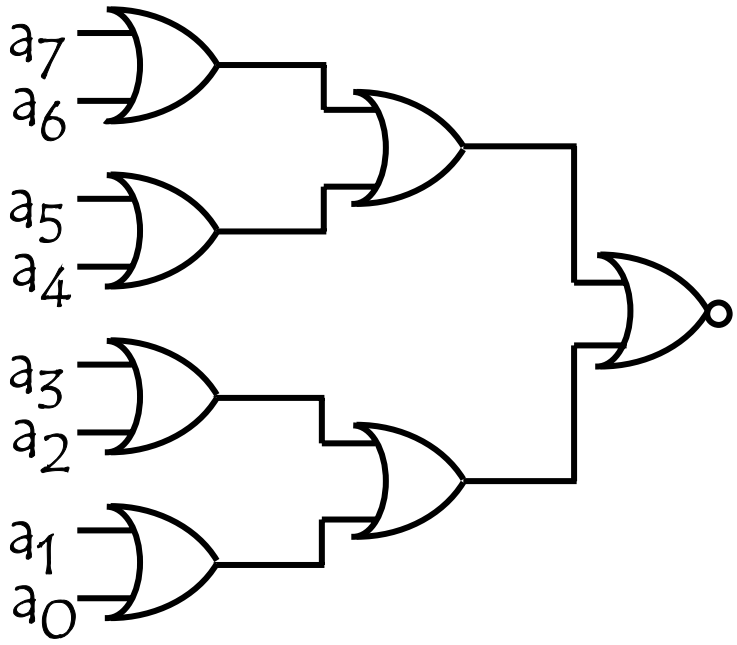
Area $\propto$ n

Delay $\propto$ n

Comparators

Comparing a natural number

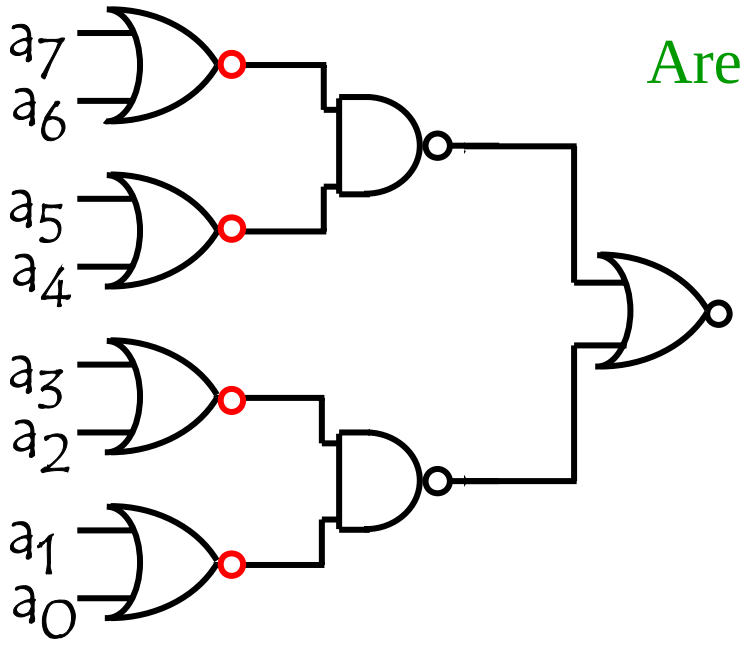
Implementation improvement



Comparators

Comparing a natural number

Implementation improvement



Area ∝ n Delay ∝ *log*(n)

Comparators

Comparing two natural numbers

Let consider two natural numbers a and b
coded on 8 bits using Natural Binary Code

$$a_7\ a_6\ a_5\ a_4\ a_3\ a_2\ a_1\ a_0$$
$$= ?$$
$$b_7\ b_6\ b_5\ b_4\ b_3\ b_2\ b_1\ b_0$$

$$\downarrow$$

$$0 / 1$$

Comparators

Comparing a natural number

Boolean function

Equal = 1 if

$$\overline{(a_7 \oplus b_7)} \cdot \ \ldots \ \cdot \overline{(a_0 \oplus b_0)} = 1$$

$$\text{Equal} = \overline{(a_7 \oplus b_7) + \ \ldots \ + (a_0 \oplus b_0)}$$

Comparators

Comparing a natural number

Implementation