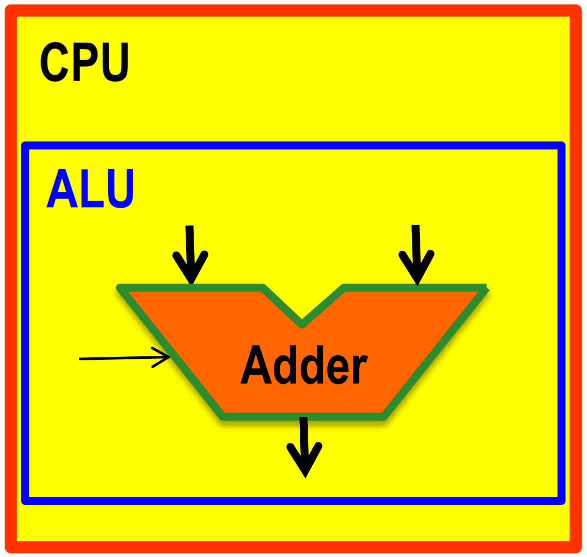
Binary Adders

SoC & CPU

- A system-on-a-chip (SoC) contains a CPU (Central Processing Unit), GPU (Graphics Processing Unit), memory, USB controller, power management circuits, wireless radios (WiFi, 3G, 4G LTE, 5G …), …
- A CPU contains: ALU (Arithmetic Logic Unit), CU (Control Unit), …

Binary Adder

- The binary Adder is the main component of the Arithmetic Logic Unit (ALU)
- Adders are also used to calculate addresses, table indices, increment and decrement operators…



Problem: 1-bit binary Adder

- Design a binary logic circuit to ADD 2-binary digits: (a, b) … (This is 1-bit Adder).

1-bit Addition

$$\begin{array}{r} a \\ +\ b \\ \hline \end{array}$$

1-bit Addition

$$\begin{array}{r} 1 \\ +\ 0 \\ \hline \end{array}$$

1-bit Adder: Truth table

- How many inputs = ?
- How many outputs = ?

1-bit Adder: Truth table

- How many inputs = 2
- How many outputs = 2

a	b	c	s
0	0		
0	1		
1	0		
1	1		

1-bit Adder: Truth table

a	b	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

1-bit Adder: Carry logic equation

a	b	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Therefore,

$$C = a\,b$$

1-bit Adder: Sum logic equation

a	b	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Therefore,

$$C = a\,b$$

$$S = \overline{a}\,b + a\,\overline{b}$$

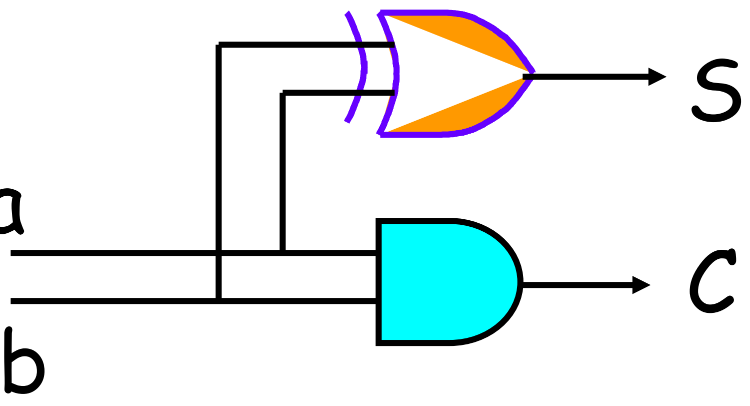
$$= a \oplus b$$

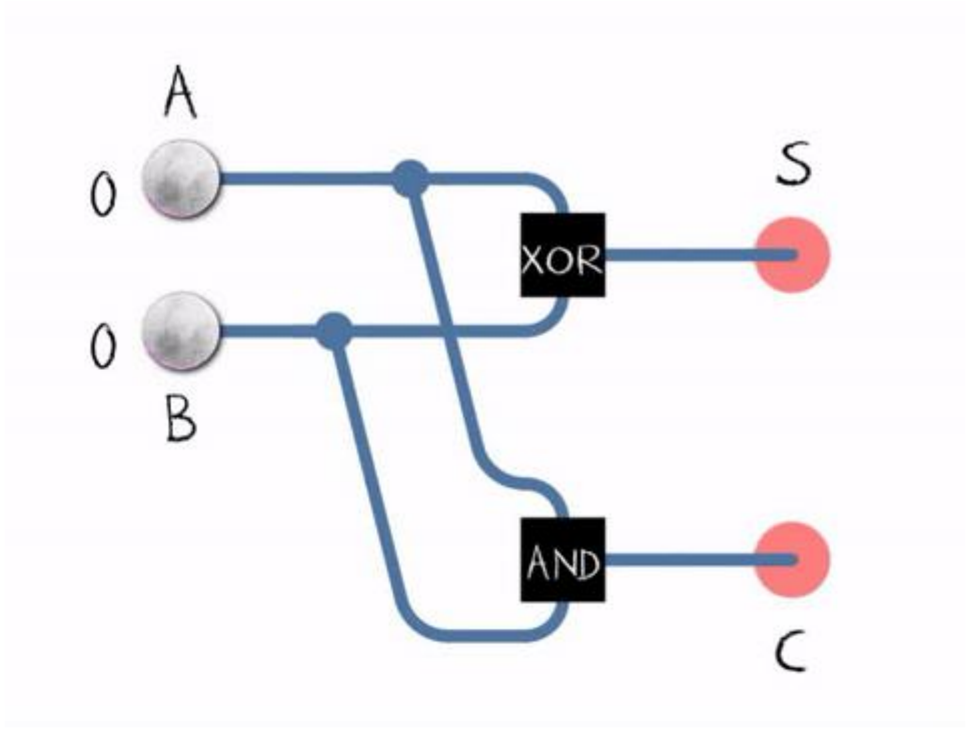
1-bit Adder: Logic circuit

$$C = a\,b$$

$$S = \overline{a}\,b + a\,\overline{b}$$

$$= a \oplus b$$




A
0
0
B
XOR
AND
S
C

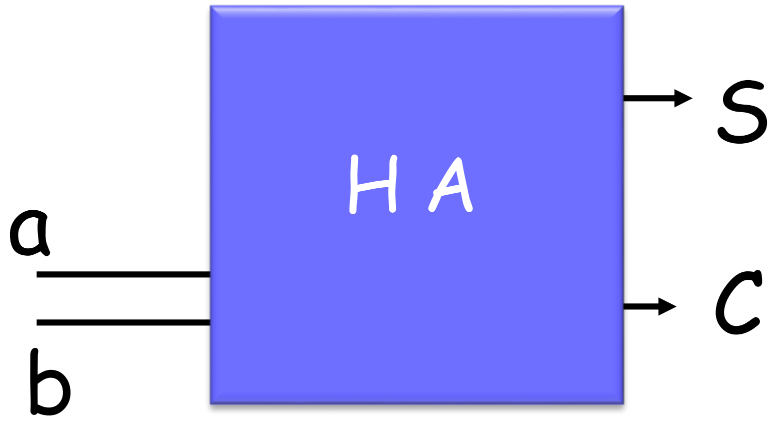
1-bit Adder: Graphical Symbol

$$C = a\,b$$

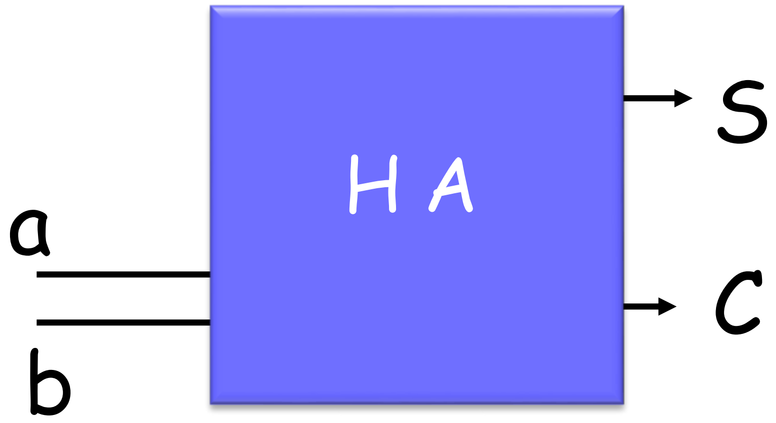
$$S = \overline{a}\,b + a\,\overline{b}$$

$$= a \oplus b$$

HA = Half-Adder



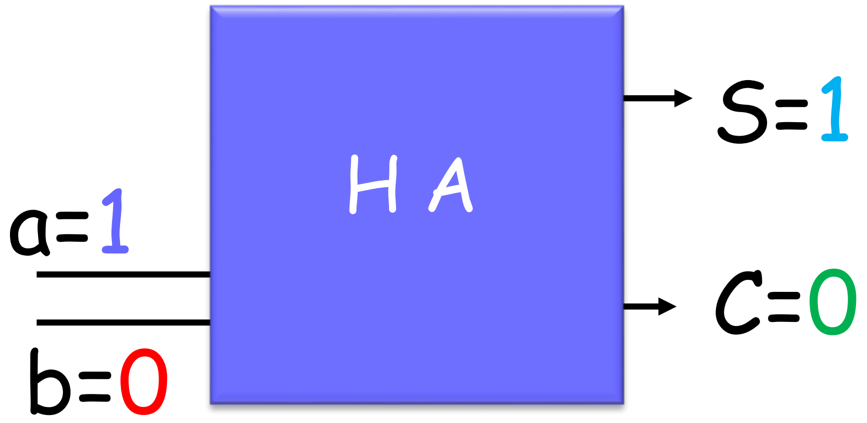
Example-1



Example-1

$$\begin{array}{r} 1 \\ +\ 0 \\ \hline \end{array}$$

0 is the carry and 1 is the sum



Example-2

$$\begin{array}{r} 11 \\ +\ 01 \\ \hline ?? \end{array}$$

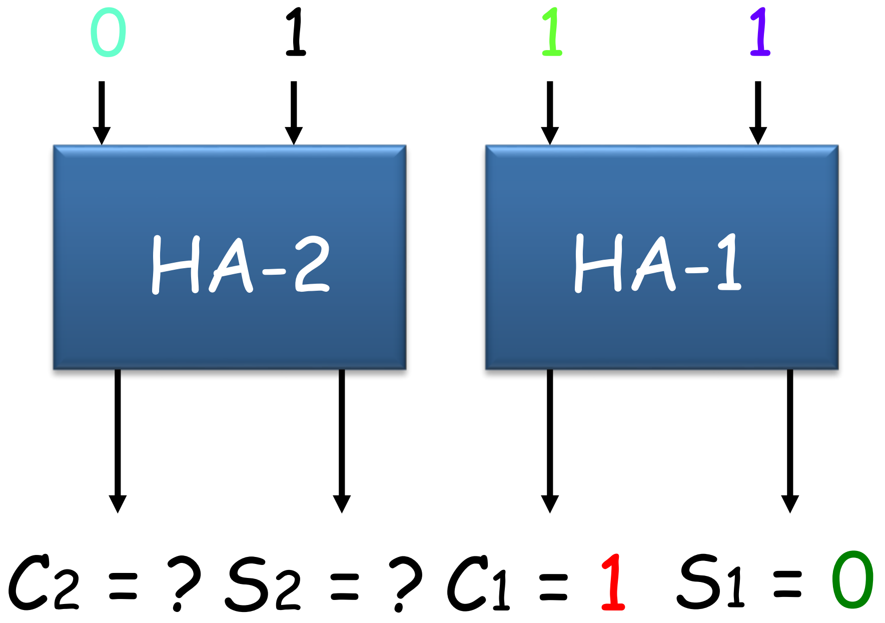
Example-2: Addition

$$\begin{array}{r} {\color{red}1} \\ 11 \\ +\ 01 \\ \hline 100 \end{array}$$

What about the logic circuit ?

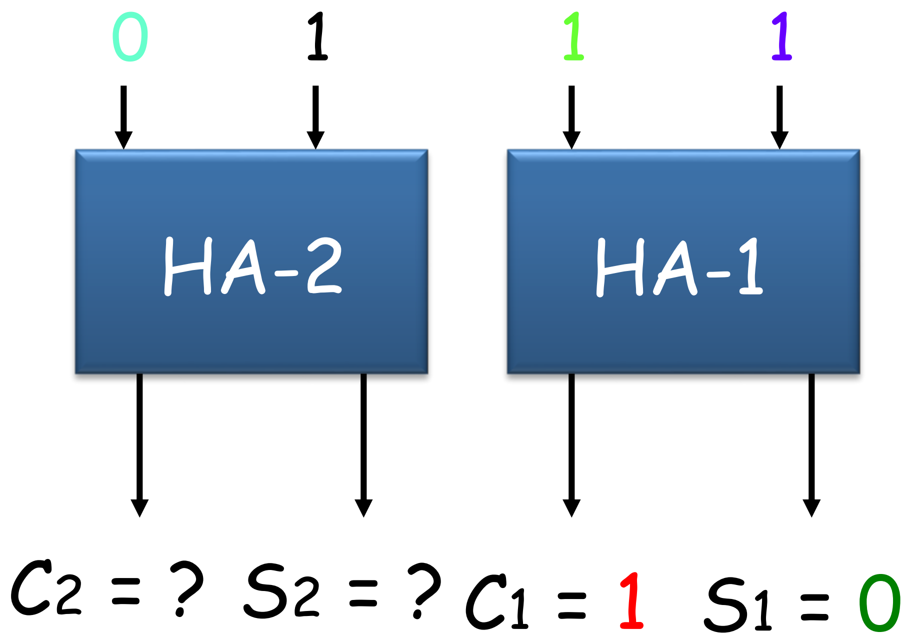
Example-2: Logic circuit

$$\begin{array}{r} 1 \\ 11 \\ +\ 01 \\ \hline 100 \end{array}$$



Example-2: Logic circuit

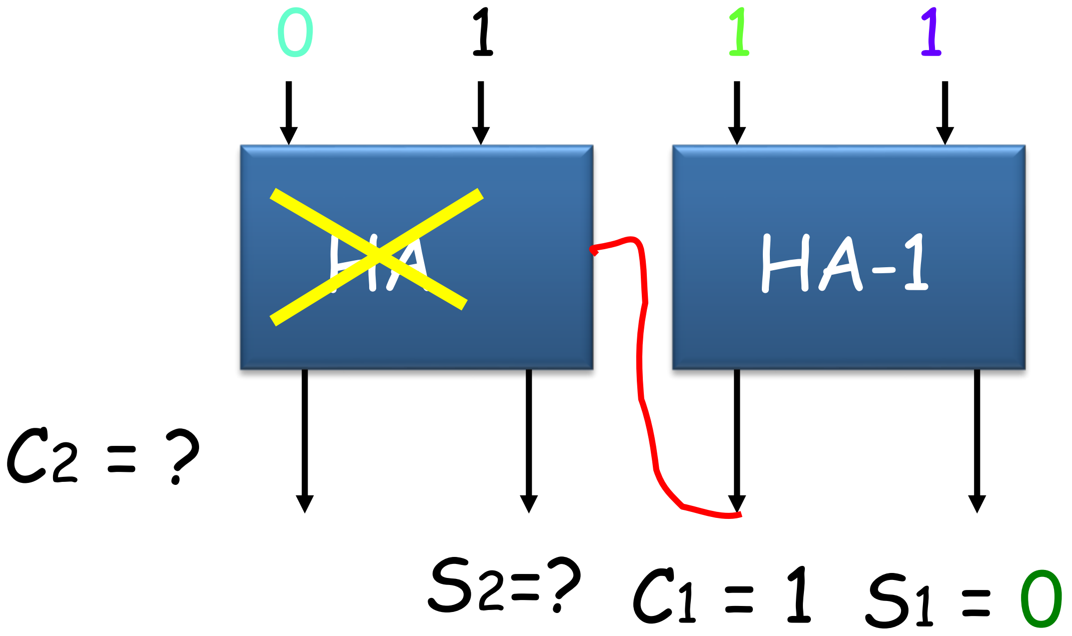
```
  1
 11
+01
---
100
```



HA-2 should have 3 inputs in order to add the carry 1 (of the HA-1) + 1 + 0

Example-2: Logic circuit

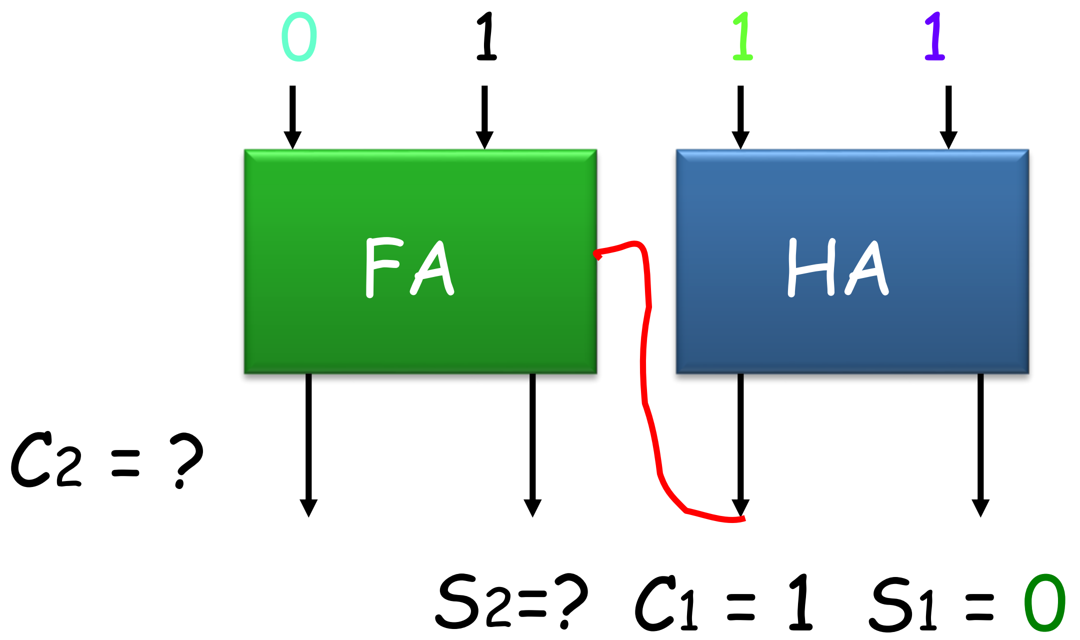
$$
\begin{array}{r}
1 \\
11 \\
+\ 01 \\
\hline
100
\end{array}
$$



Therefore we need logic adders with three inputs = ***?***

Full-Adder ?

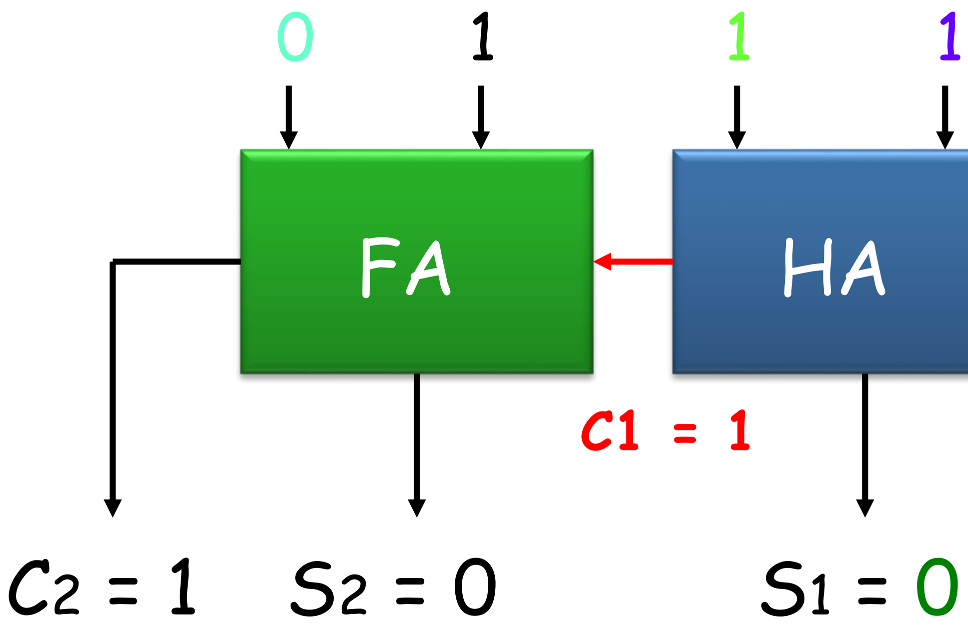
$$\begin{array}{r} 1 \\ 11 \\ +\ 01 \\ \hline 100 \end{array}$$



Therefore we need logic adders with three inputs = ***Full Adders***

Result

```
  1
  11
+ 01
-----
 100
```

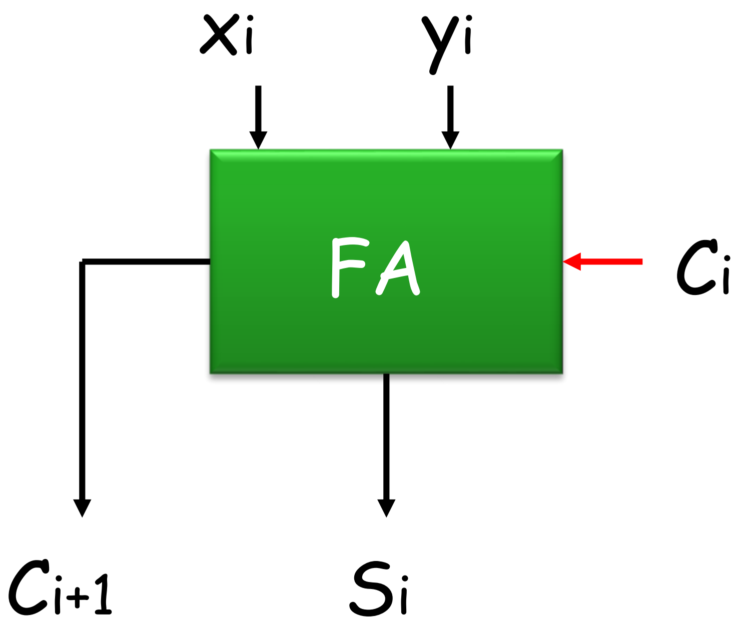


Therefore we need logic adders with three inputs = ***Full Adders***

Design a Full Adder

3 inputs and 2 outputs

Full-Adder



Full-Adder: Truth table

C_i	x_i	y_i	C_{i+1}	S_i
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Full-Adder: Truth table

C_i	x_i	y_i	C_{i+1}	S_i
0	0	0	0	0
0	0	1	0	1
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Full-Adder: Truth table

C_i	x_i	y_i	C_{i+1}	S_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Full-Adder: Logic equations

C_i	x_i	y_i	C_{i+1}	S_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$C_{i+1} = \overline{C_i}\, x_i\, y_i + C_i\, \overline{x_i}\, y_i + C_i\, x_i\, \overline{y_i} + C_i\, x_i\, y_i$$

$$S_i = \overline{C_i}\, \overline{x_i}\, y_i + \overline{C_i}\, x_i\, \overline{y_i} + C_i\, \overline{x_i}\, \overline{y_i} + C_i\, x_i\, y_i$$

Full-Adder: Simplification: C_{i+1}

Carry

Full-Adder: Simplification … C_{i+1}

$$C_{i+1} = \overline{C_i}\, x_i\, y_i + C_i\, \overline{x_i}\, y_i + C_i\, x_i\, \overline{y_i} + C_i\, x_i\, y_i$$

Full-Adder: Simplification … C_{i+1}

$$C_{i+1} = \overline{C_i}\, x_i\, y_i + C_i\, \overline{x_i}\, y_i + C_i\, x_i\, \overline{y_i} + C_i\, x_i\, y_i$$

$$= C_i(\overline{x_i}\, y_i + x_i\, \overline{y_i}) + x_i\, y_i\, (\overline{C_i} + C_i)$$

Full-Adder: Simplification … C_{i+1}

$$C_{i+1} = \overline{C_i}\, x_i\, y_i + C_i\, \overline{x_i}\, y_i + C_i\, x_i\, \overline{y_i} + C_i\, x_i\, y_i$$

$$= C_i(\overline{x_i}\, y_i + x_i\, \overline{y_i}) + x_i\, y_i\, (\overline{C_i} + C_i)$$

$$\boxed{C_{i+1} = C_i(x_i \oplus y_i) + x_i\, y_i}$$

Full-Adder: Simplification: S_i

Sum

Simplify: S_i

$$S_i = \overline{C_i}\,\overline{x_i}\,y_i + \overline{C_i}\,x_i\,\overline{y_i} + C_i\,\overline{x_i}\,\overline{y_i} + C_i\,x_i\,y_i$$

Simplify: S_i

$$S_i = \underbrace{\overline{C_i}\,\overline{x_i}\,y_i + \overline{C_i}\,x_i\,\overline{y_i}}_{} + \underbrace{C_i\,\overline{x_i}\,\overline{y_i} + C_i\,x_i\,y_i}_{}$$

Simplify: S_i

$$S_i = \underbrace{\overline{C_i}\,\overline{x_i}\,y_i + \overline{C_i}\,x_i\,\overline{y_i}} + \underbrace{C_i\,\overline{x_i}\,\overline{y_i} + C_i\,x_i\,y_i}$$

$$S_i = \overline{C_i}(x_i \oplus y_i) \qquad + \quad C_i(\overline{x_i}\,\overline{y_i} + x_i\,y_i)$$

Simplify: S_i

$$S_i = \underbrace{\overline{C_i}\,\overline{x_i}\,y_i + \overline{C_i}\,x_i\,\overline{y_i}} + \underbrace{C_i\,\overline{x_i}\,\overline{y_i} + C_i\,x_i\,y_i}$$

$$S_i = \overline{C_i}(x_i \oplus y_i) \qquad + \quad C_i(\overline{x_i}\,\overline{y_i} + x_i\,y_i)$$

$$S_i = \overline{C_i}(\underbrace{x_i \oplus y_i}) \qquad + \quad C_i(\underbrace{\overline{x_i \oplus y_i}})$$

Done…

$$S_i = \underbrace{\overline{C_i}\,\overline{x_i}\,y_i + \overline{C_i}\,x_i\,\overline{y_i}}_{} + \underbrace{C_i\,\overline{x_i}\,\overline{y_i} + C_i\,x_i\,y_i}_{} = C_i \oplus x_i \oplus y_i$$

$$S_i = \overline{C_i}(x_i \oplus y_i) \qquad + \quad C_i(\overline{x_i}\,\overline{y_i} + x_i\,y_i)$$

$$S_i = \overline{C_i}\underbrace{(x_i \oplus y_i)}_{} \qquad + \quad C_i\underbrace{\overline{(x_i \oplus y_i)}}_{}$$

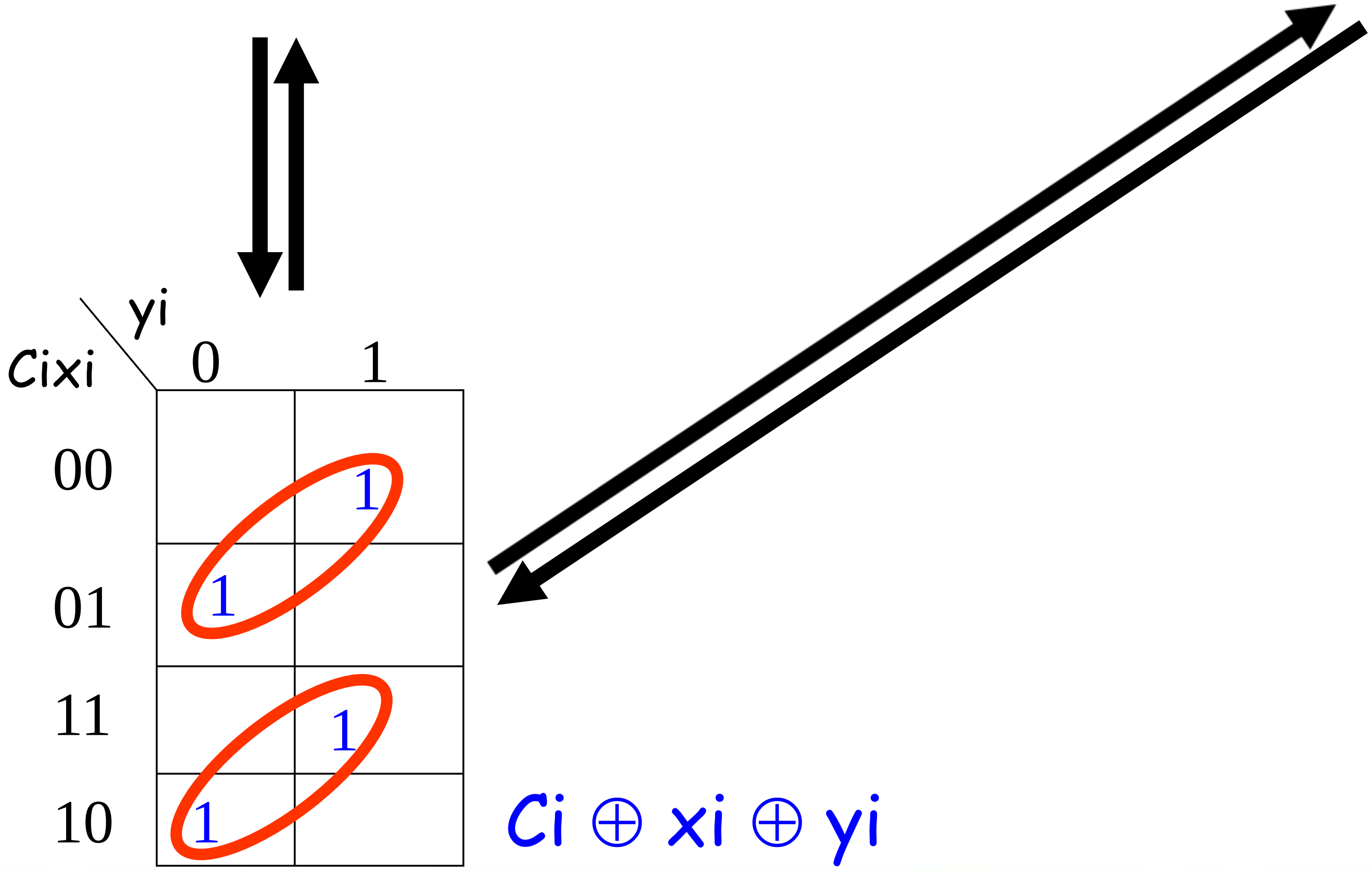
K-maps and XOR gates …

$$S_i = \underbrace{\overline{C_i}\,\overline{x_i}\,y_i + \overline{C_i}\,x_i\,\overline{y_i}} + \underbrace{C_i\,\overline{x_i}\,\overline{y_i} + C_i\,x_i\,y_i}$$

What is the K-map of the above logical expression?

All equivalent … "diagonal" looping …

$$S_i = \overline{C_i}\,\overline{x_i}\,y_i + \overline{C_i}\,x_i\,\overline{y_i} + C_i\,\overline{x_i}\,\overline{y_i} + C_i\,x_i\,y_i = C_i \oplus x_i \oplus y_i$$



$C_i \oplus x_i \oplus y_i$

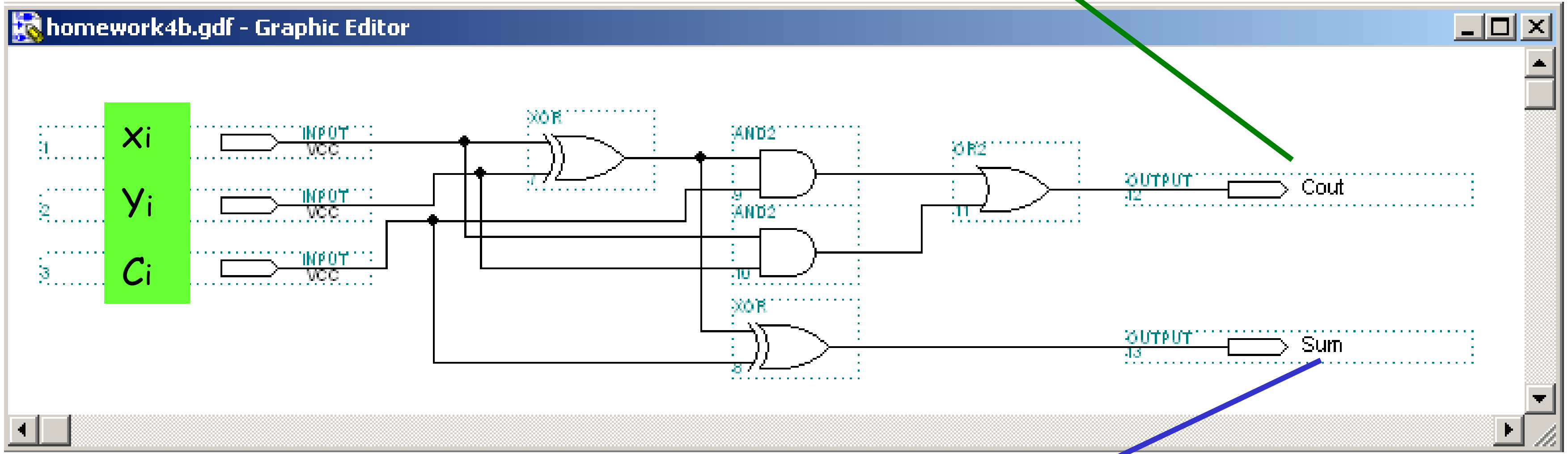
Finally … Full-adder equations

$$C_{i+1} = C_i(x_i \oplus y_i) + x_i y_i$$

$$S_i = C_i \oplus x_i \oplus y_i$$

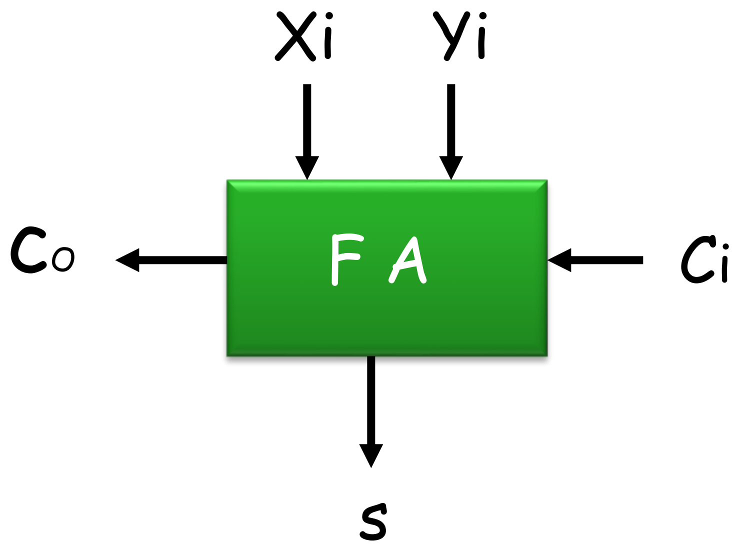
Full-adder … circuit

$C_{i+1} = C_i(x_i \oplus y_i) + x_i y_i$



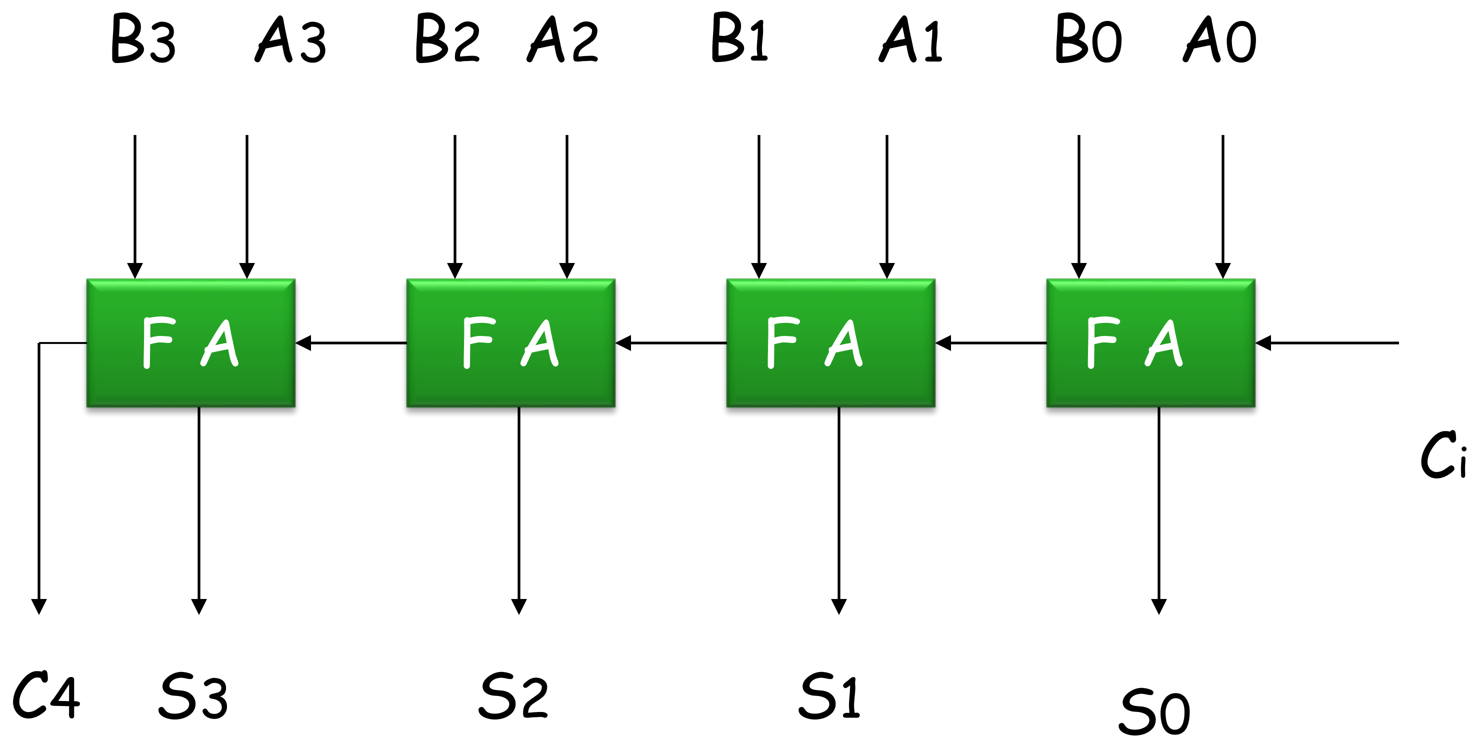
$S_i = C_i \oplus x_i \oplus y_i$

1-bit Full-Adder (3-inputs, 2-outputs)



4-Bit Adder

4-bit Adder

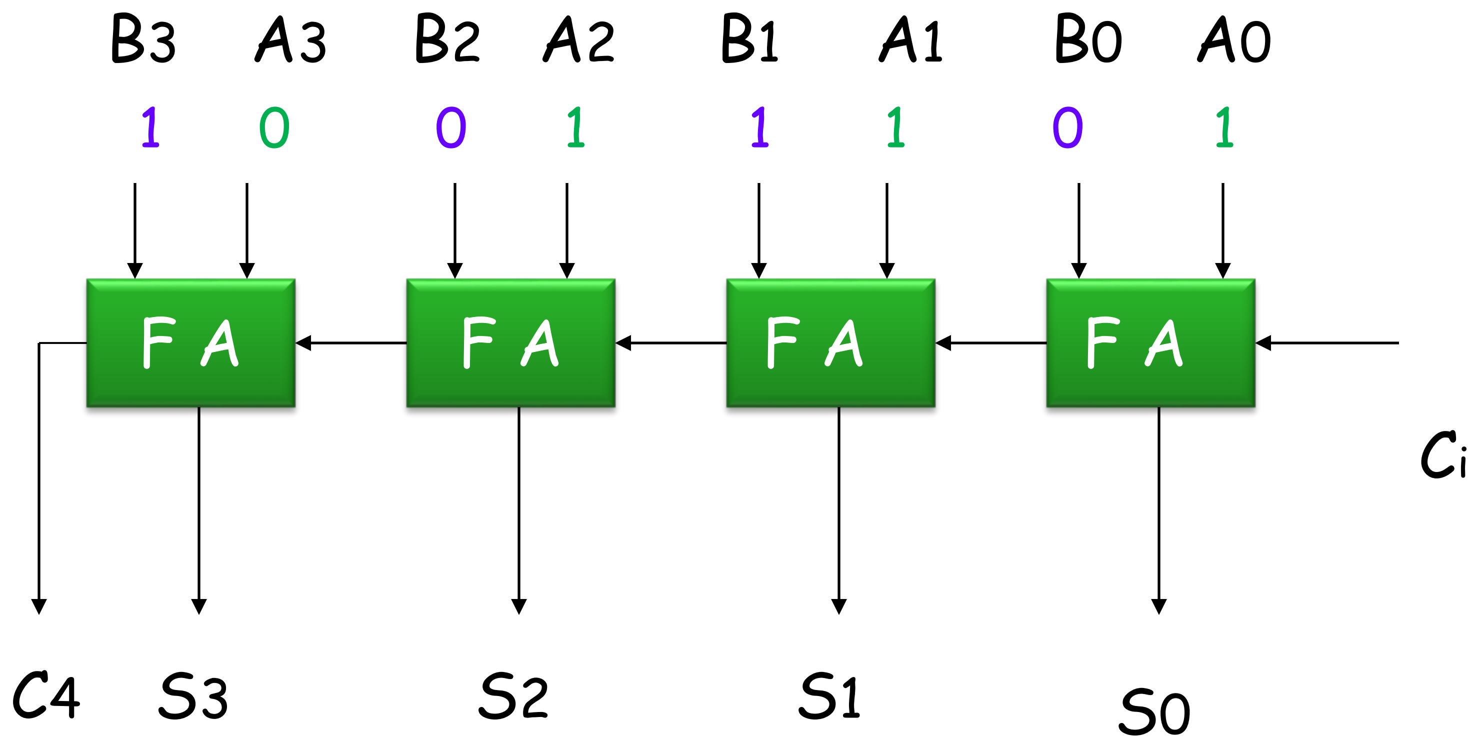


4-Bit Adder Example

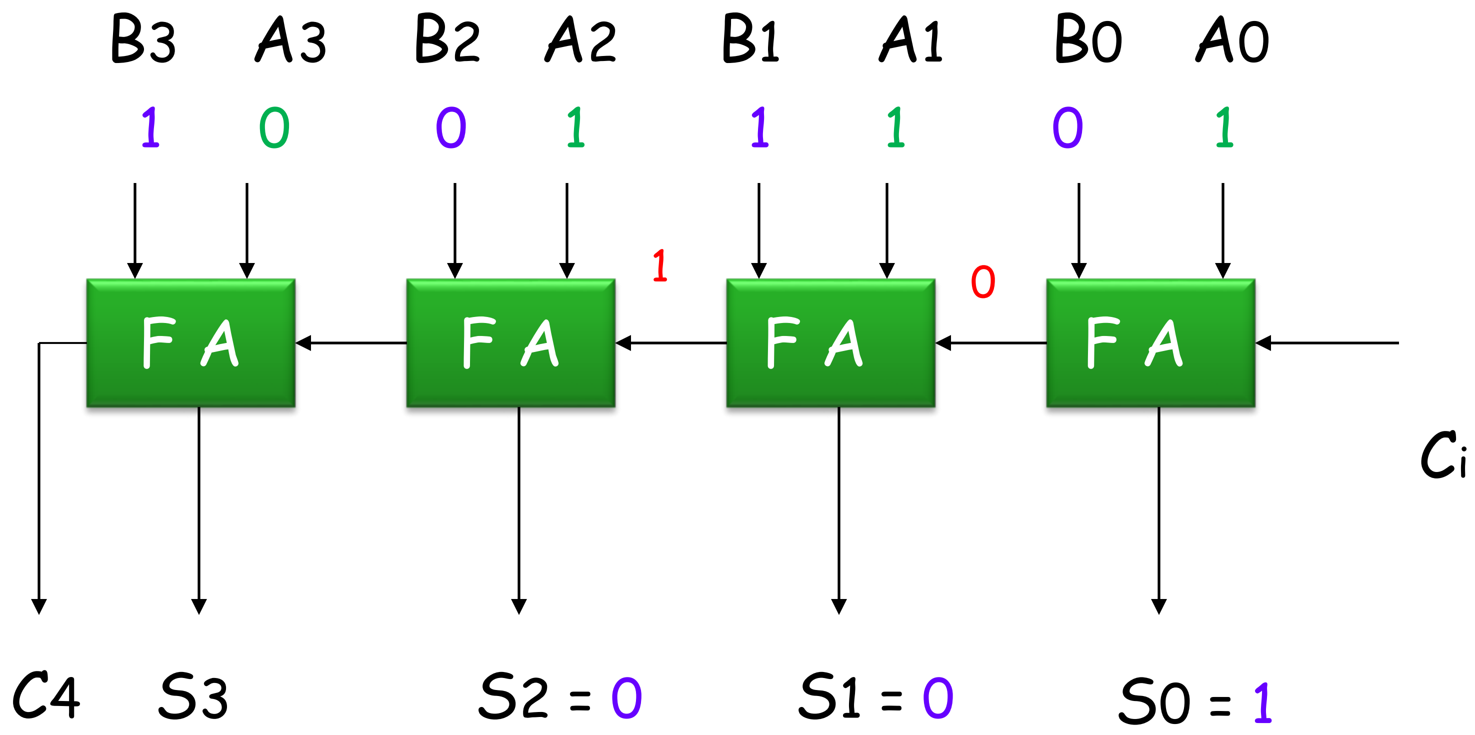
Add: A+B

- 0111 = A
- 1010 = B

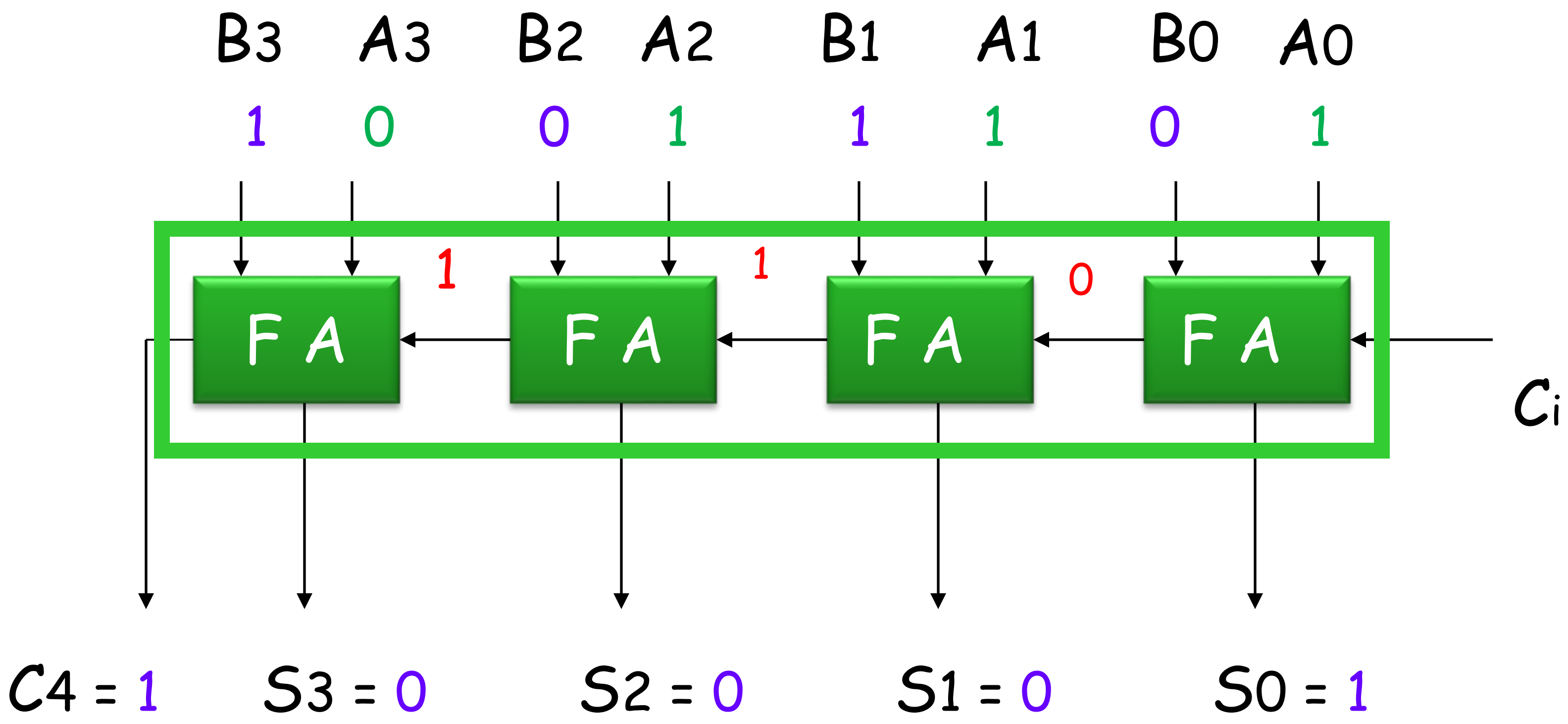
4-bit Adder example



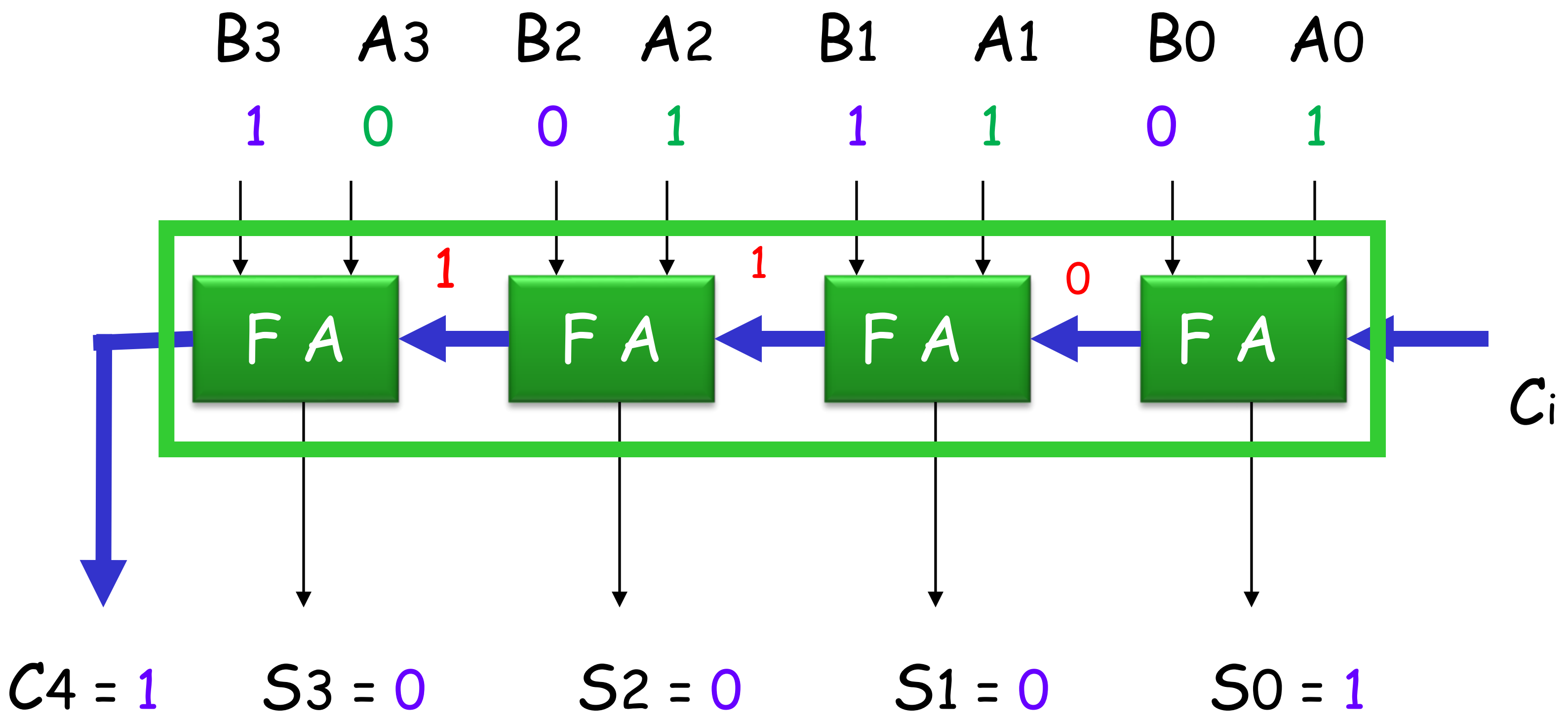
4-bit Adder example



4-bit Adder example

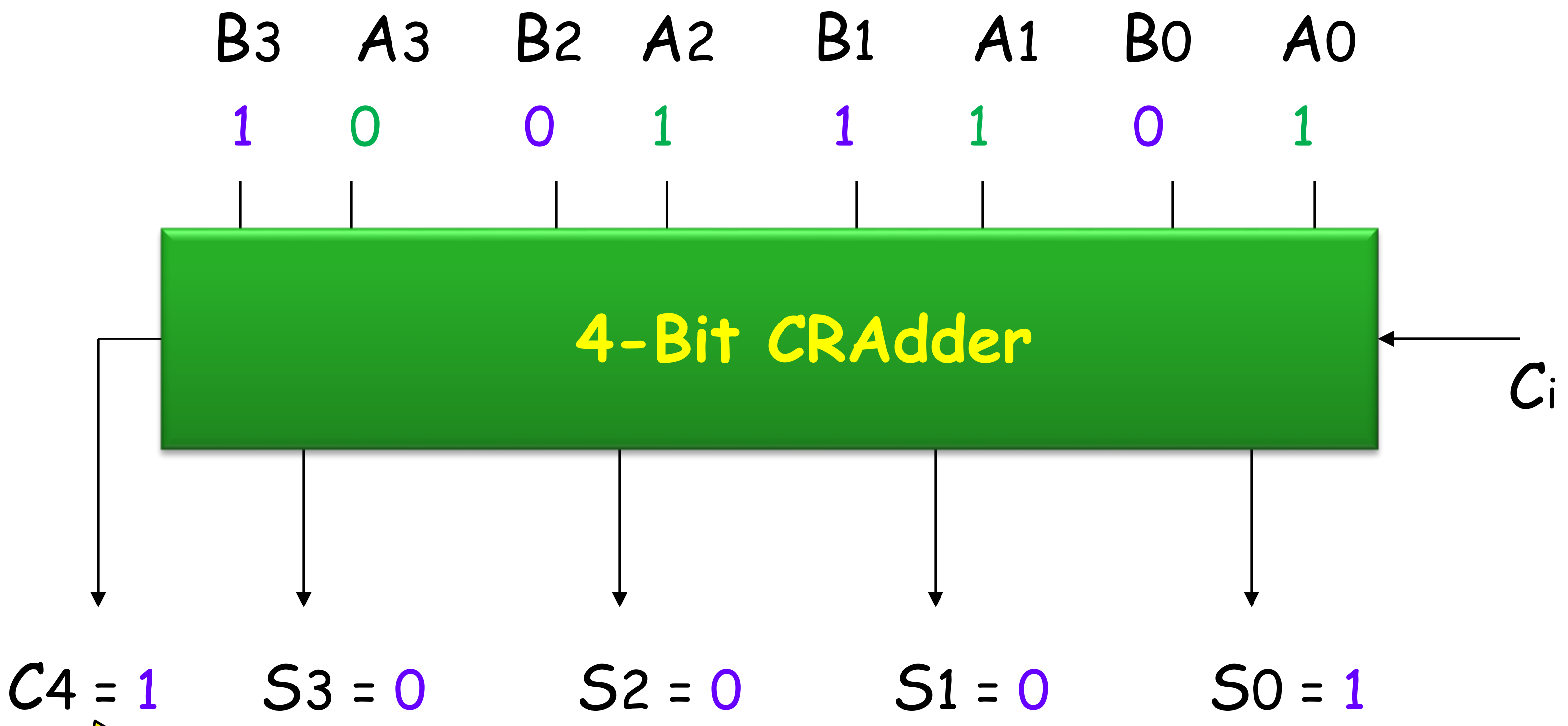


4-bit Carry Ripple Adder (CRA)



The carry "ripples" through the full adders

4-bit CRA: Compact form

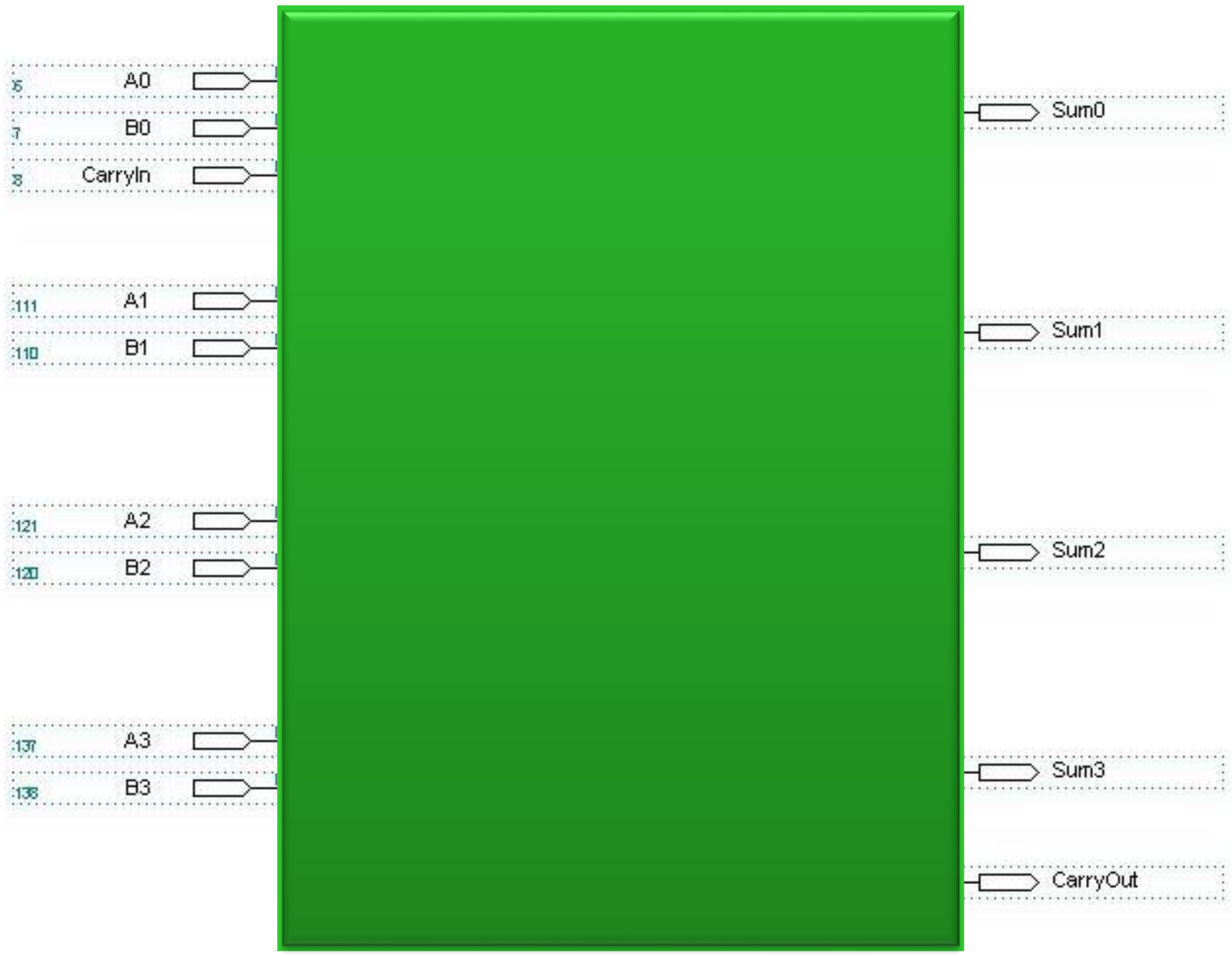


Overflow …the result is
5-bits and the input is 4-bits

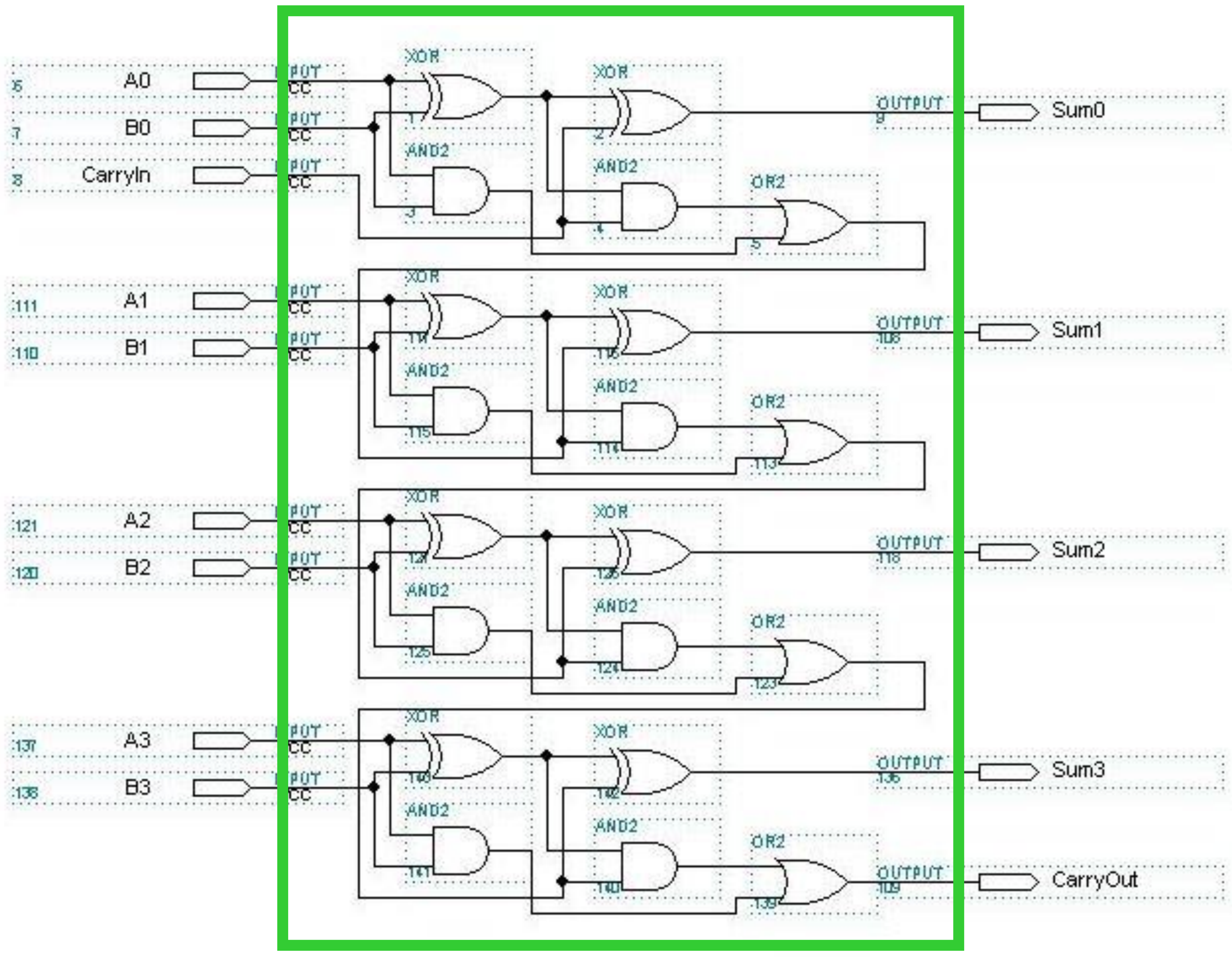
Overflow …

- When the addition result has an extra bit (5-bits) than the inputs (4-bits), this is called overflow
- Overflow occurs in case where the carry-out is one (unsigned numbers addition)
- Overflow is a hardware related “problem”…

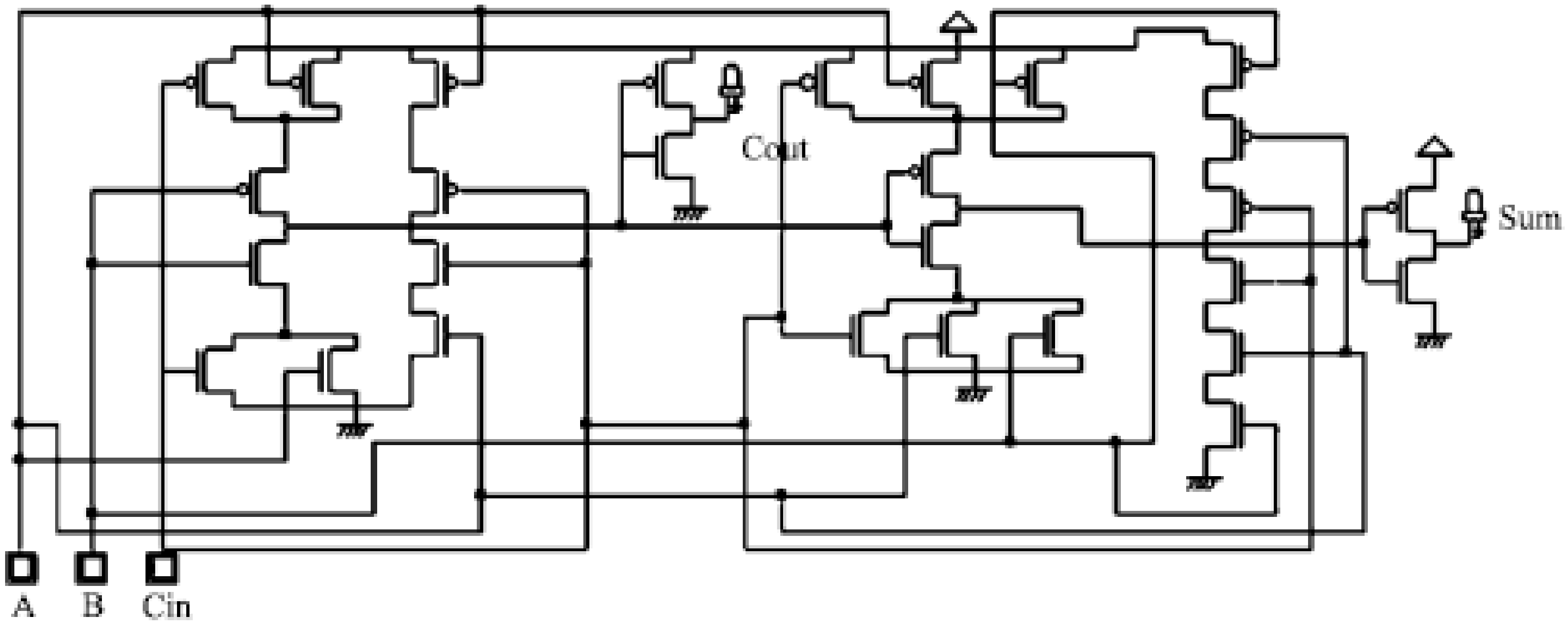
4-bit CRA …



4-bit CRA … circuit



CMOS Carry Ripple Adder, with transistors



$$\mathrm{Sum} = A \oplus B \oplus \mathrm{Cin}$$

$$\mathrm{Cout} = A.B + \mathrm{Cin}.(A \oplus B)$$

High Speed CMOS logic 4-bit Binary Full Adder

CD74HC283