Queen Mary
University of London
Science and Engineering

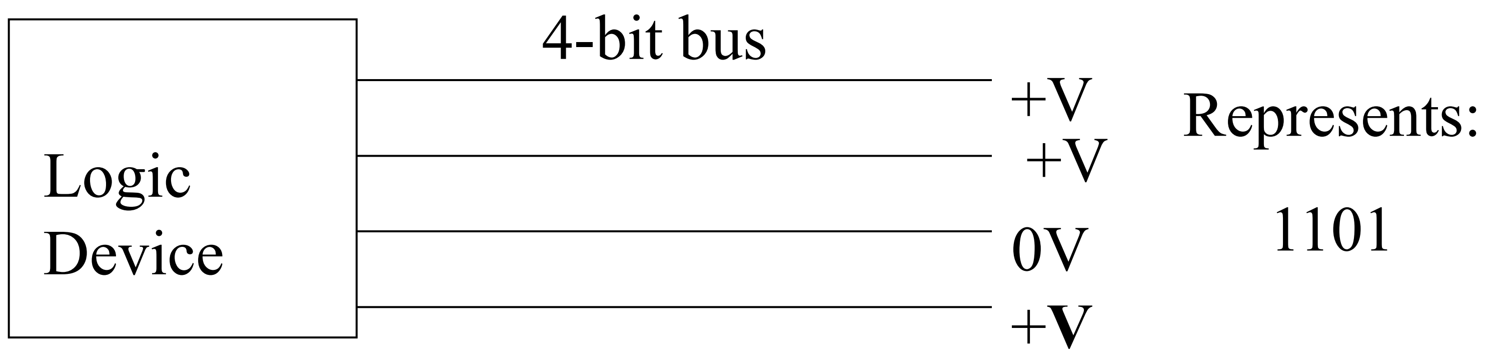
EBU4202: Digital Circuit Design
Digital System Blocks

Dr. Md Hasanuzzaman Sagor (Hasan)
Dr. Chao Shu (Chao)
Dr. Farha Lakhani (Farha)

School of Electronic Engineering and Computer Science,
Queen Mary University of London,
London, United Kingdom.

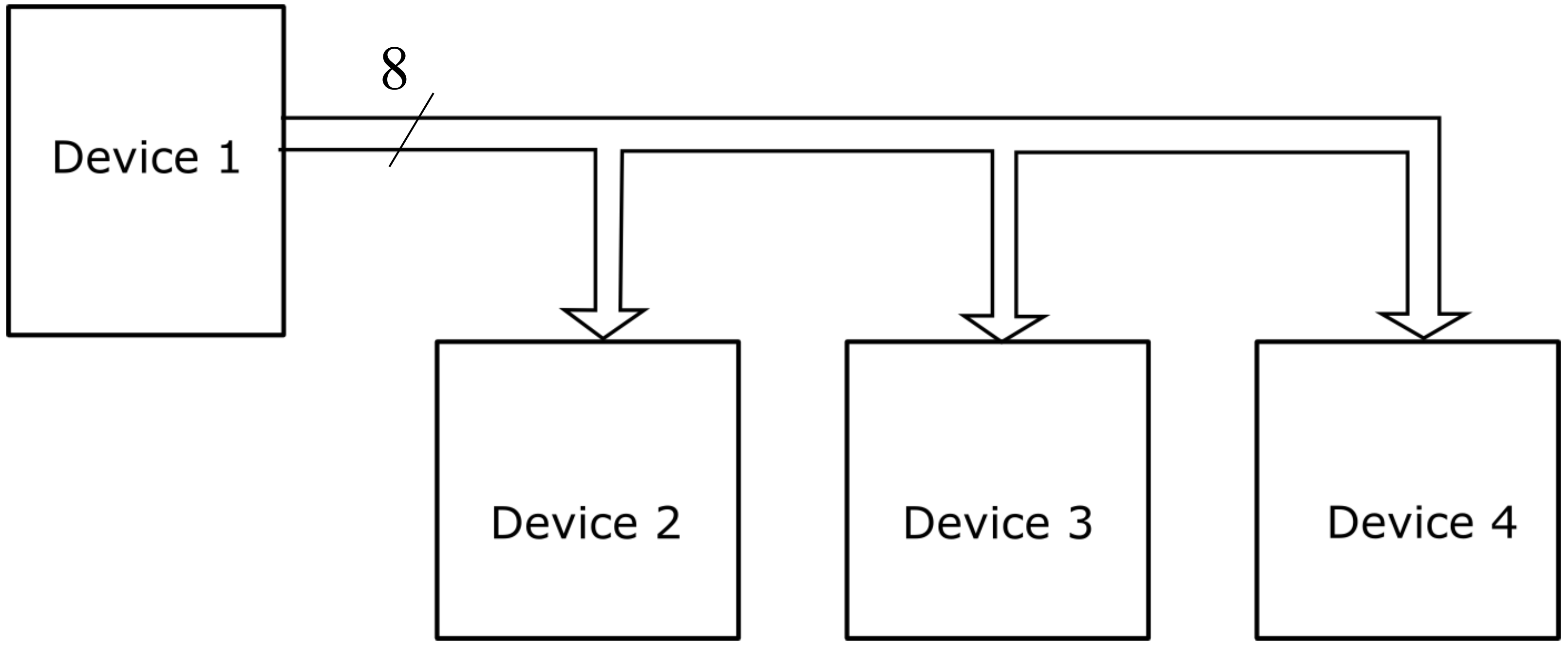
Buses

- Set of two or more electrical conductors representing a binary value



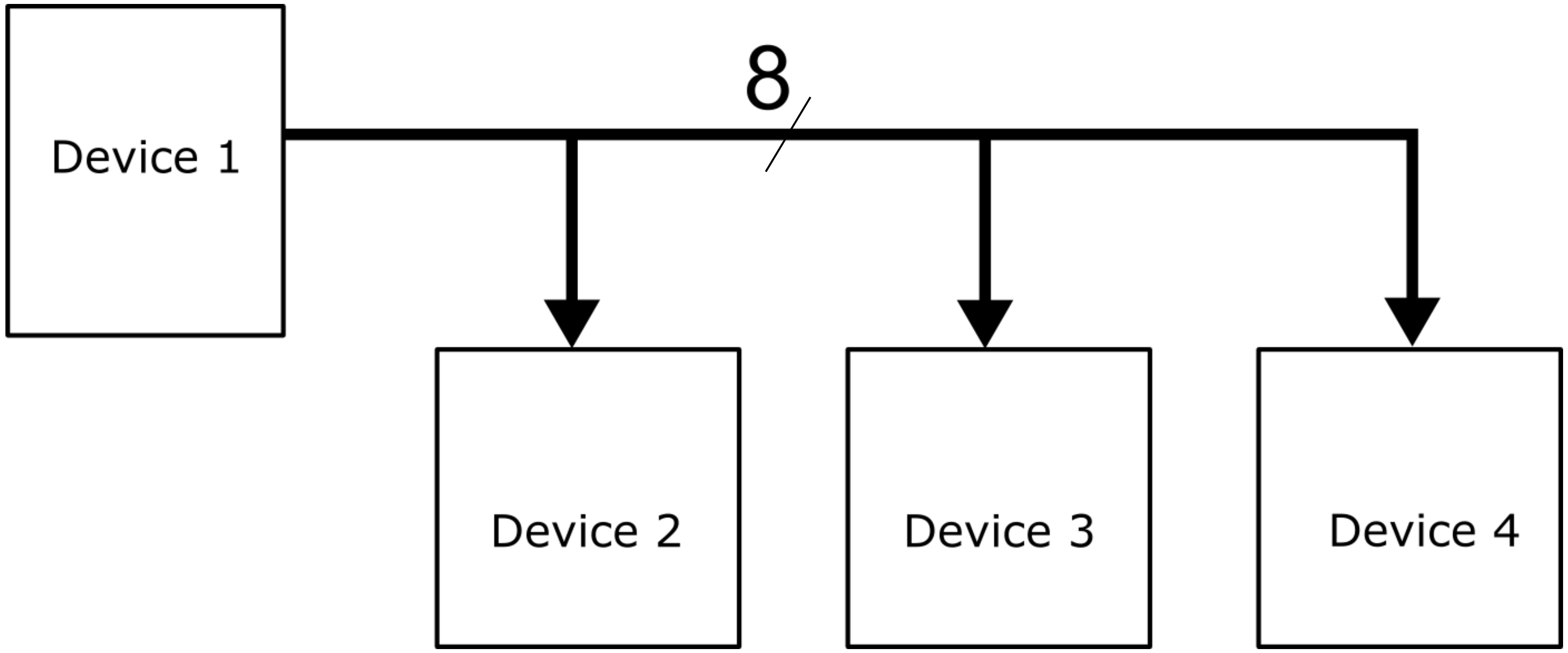
Buses

- Often more than just a one-to-one connection



Buses

- Often more than just a one-to-one connection

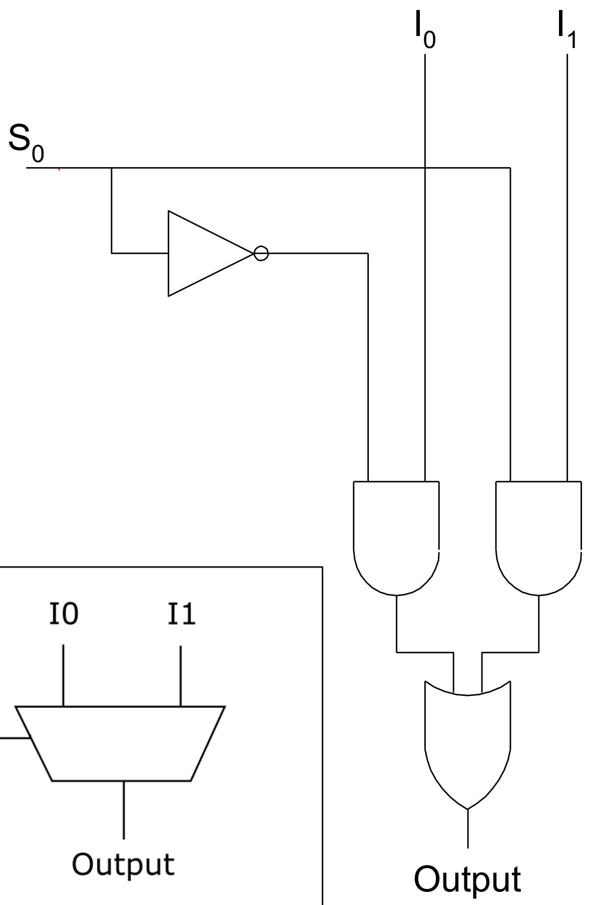


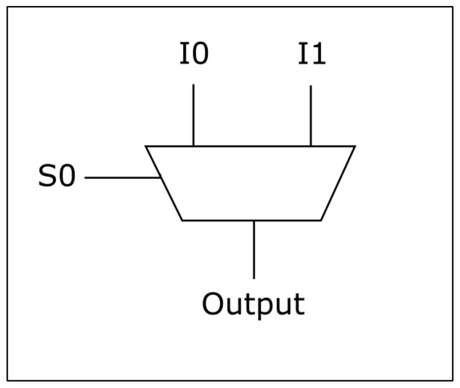
Selectors and Decoders

Selectors or *Multiplexers*

- Often labelled MUX
- 2-Input Selector:

S_0	Output
0	I_0
1	I_1

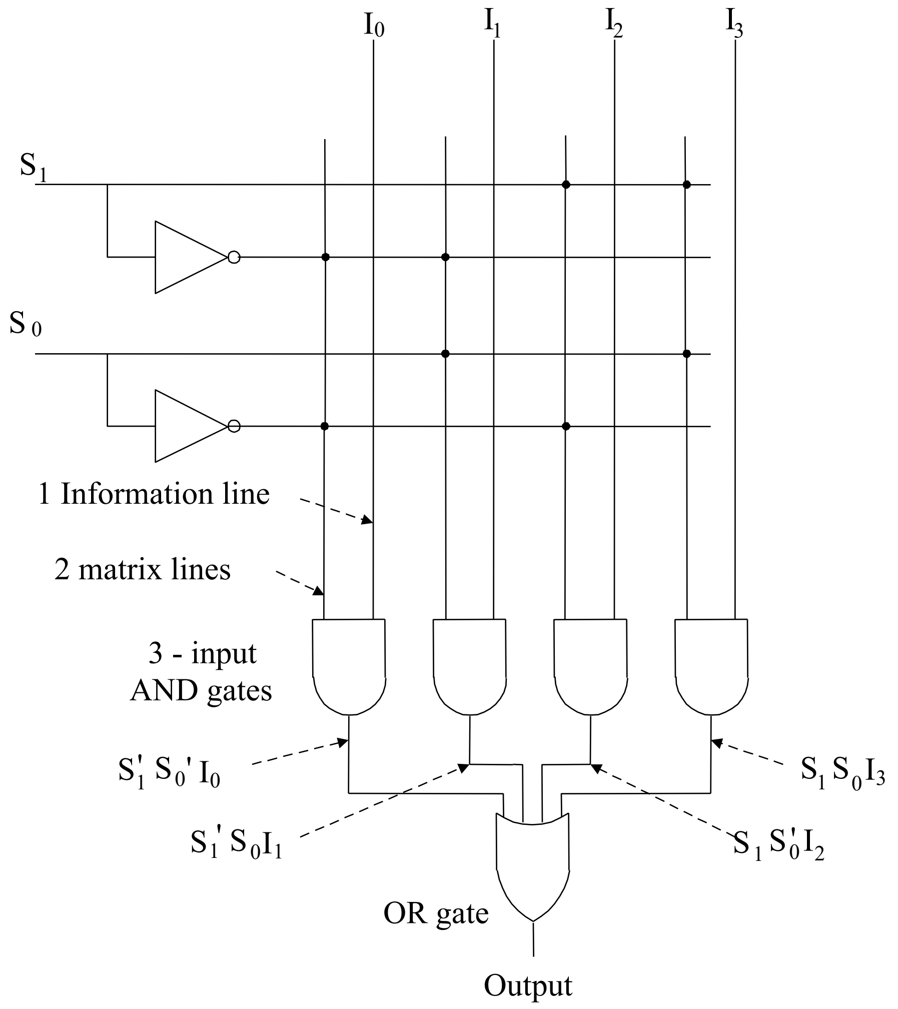




Selectors or *Multiplexers*

4-input Selector

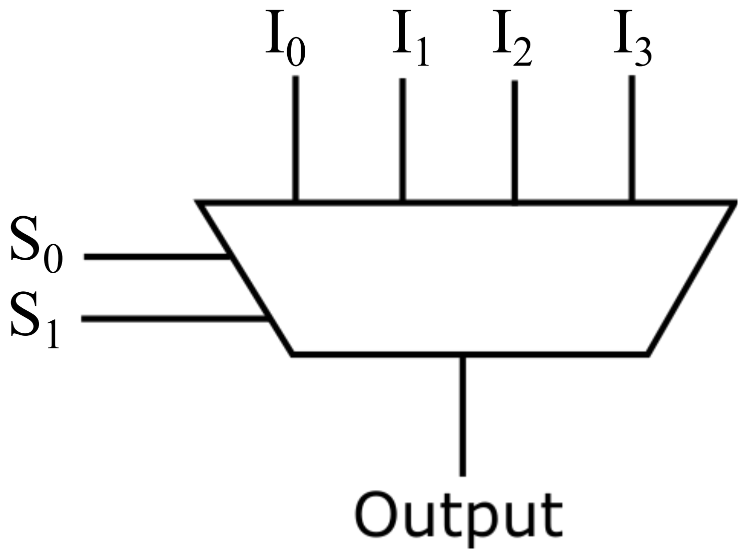
S_1	S_0	Output
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3



Multiplexers or Selectors

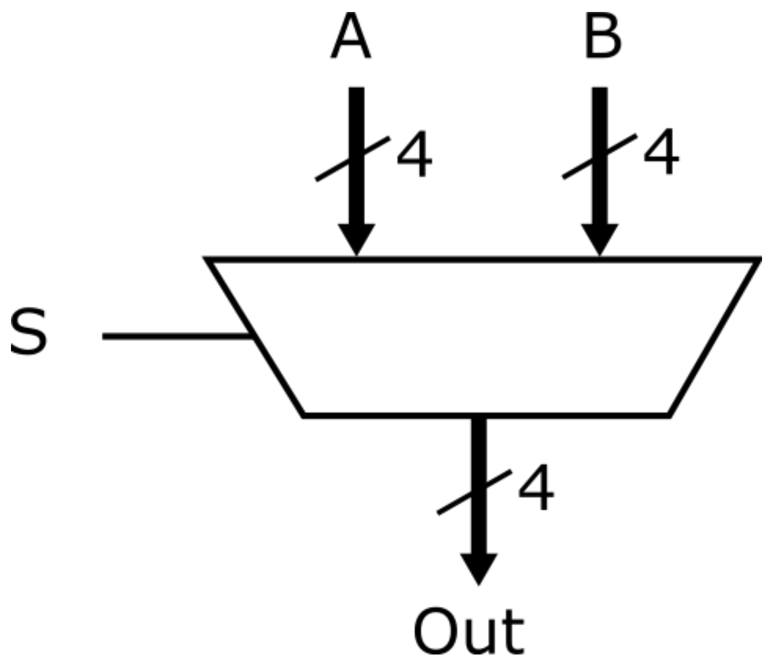
4-input Selector

S_1	S_0	Output
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3



Multiplexers or Selectors

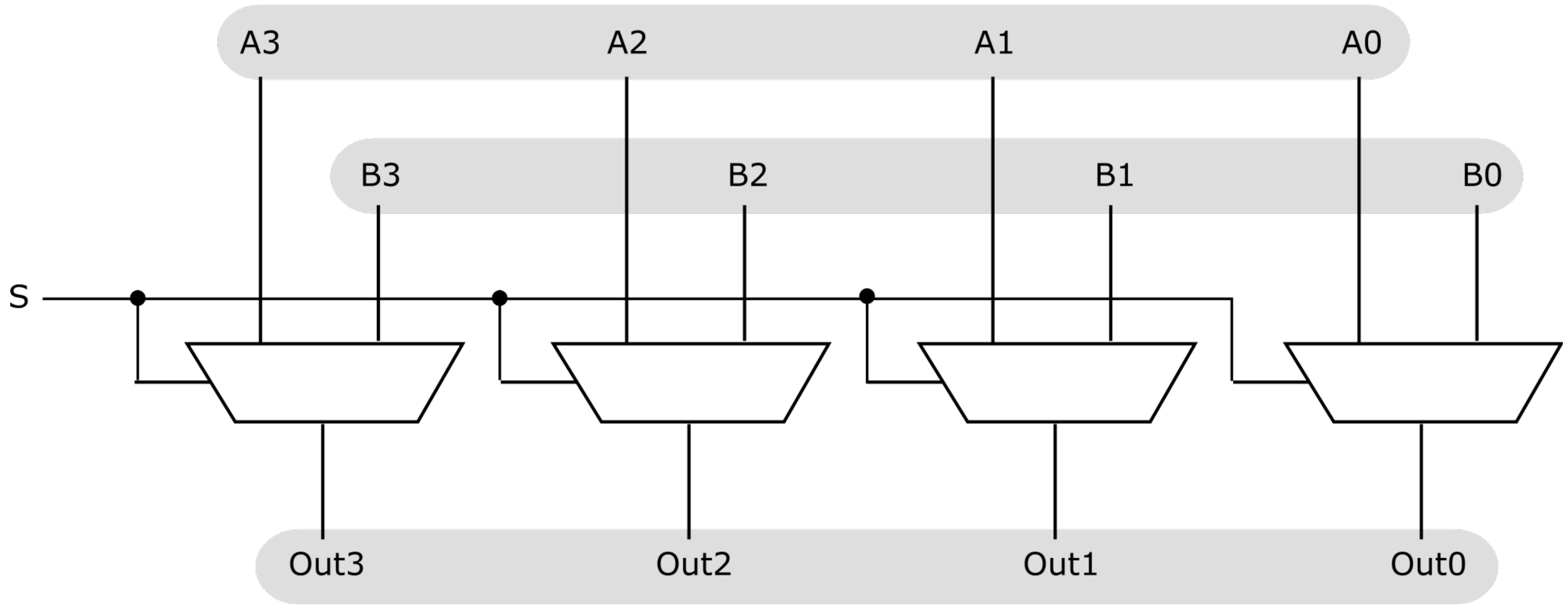
2-Input 4-bit MUX



Queen Mary University of London

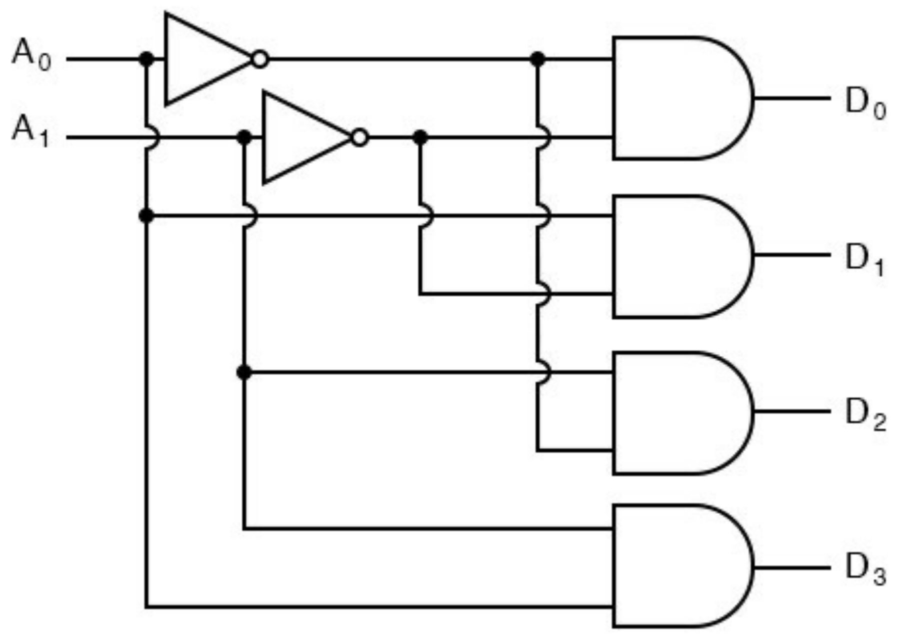
Multiplexers or Selectors

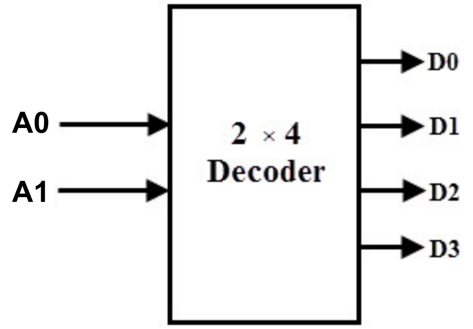
2-Input 4-bit MUX – Internal Implementation



Decoder or *Demultiplexer*

- Converts n number of binary information into 2^n number of output lines.
- Outputs 1 on the wire corresponding to the binary number represented by the inputs.

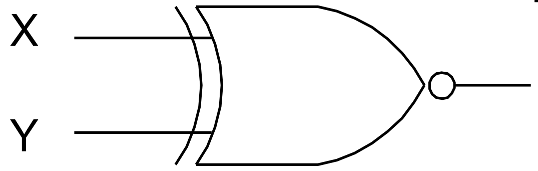




A1	A0	D3	D2	D1	D0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Queen Mary University of London

Magnitude Comparator (XNOR)



Output = (X'Y+XY')'
= XY+X'Y'

X	Y	Output
0	0	1
0	1	0
1	0	0
1	1	1

Basic comparator:

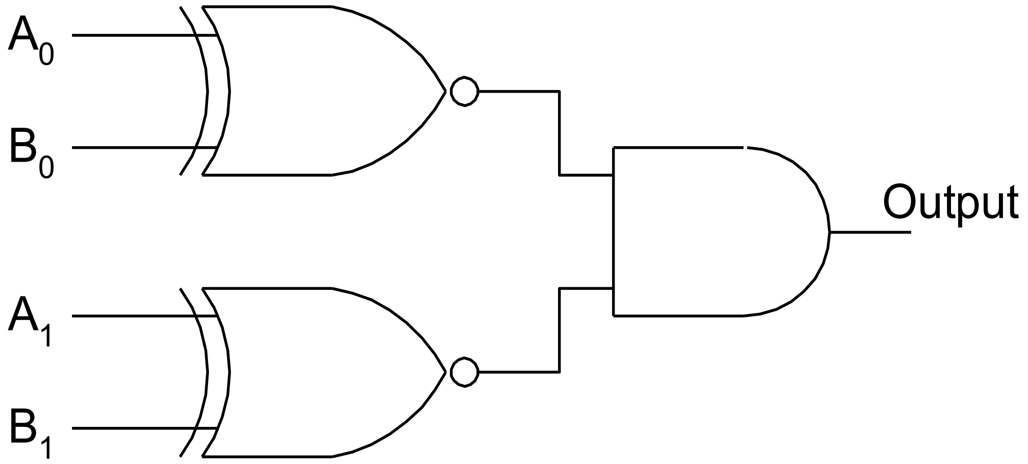
- output when X = Y
- complement gives X $\neq$Y

Other types give:

- X > Y
- X < Y
- complements give:
  - X $\leq$ Y
  - X $\geq$ Y

Queen Mary
University of London

Multiple-bit Magnitude Comparator



For more than 1-bit comparisons, the XNORs are ANDed together

Exercise: Show that output of above 2-bit comparator is given by:

$$(A_0B_0 + A_0'B_0')\cdot(A_1B_1 + A_1'B_1')$$

Queen Mary University of London

Arithmetic Units

Adders - the Half-adder

Half-adder:

- accepts two binary digit inputs (X & Y)
- produces Sum (S) & Carry (C) outputs

Arithmetically:

$$\begin{array}{cc} & X \\ & Y \quad + \\ \hline C & S \end{array}$$

$$\begin{array}{c|cc} X \backslash Y & 0 & 1 \\ \hline 0 & 0 & 1 \\ 1 & 1 & 0 \end{array}$$

Sum

$$S = X\overline{Y} + \overline{X}Y = X \oplus Y$$

Truth table:

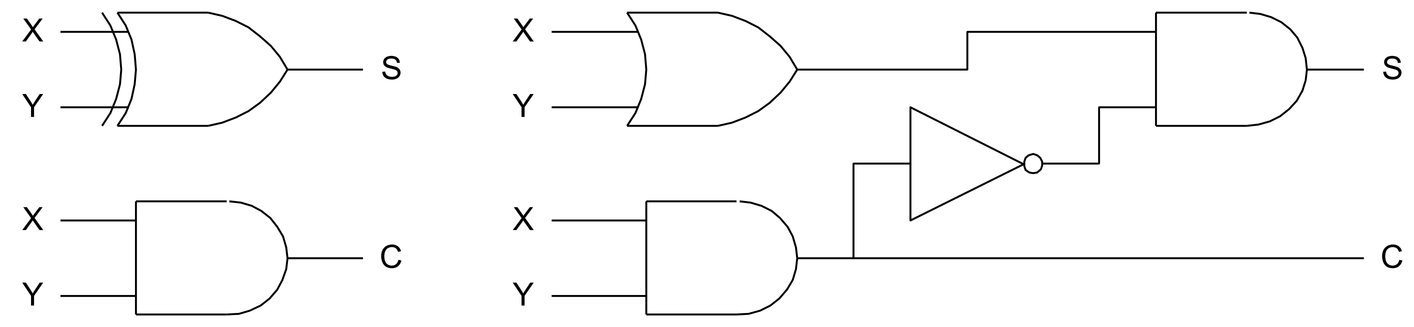
X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

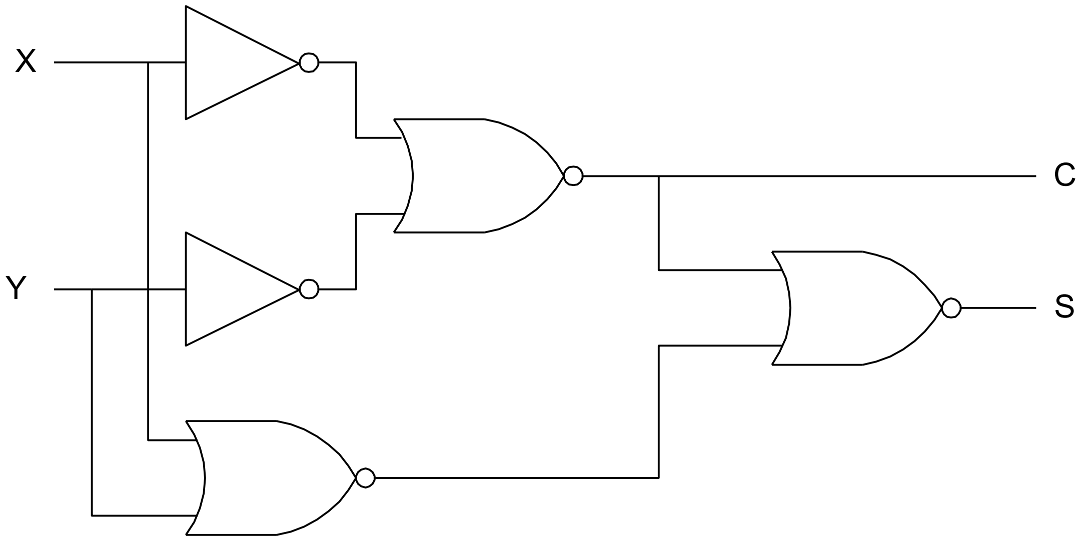
$$\begin{array}{c|cc} X \backslash Y & 0 & 1 \\ \hline 0 & 0 & 0 \\ 1 & 0 & 1 \end{array}$$

Carry

$$C = XY$$

Examples of half-adder implementations



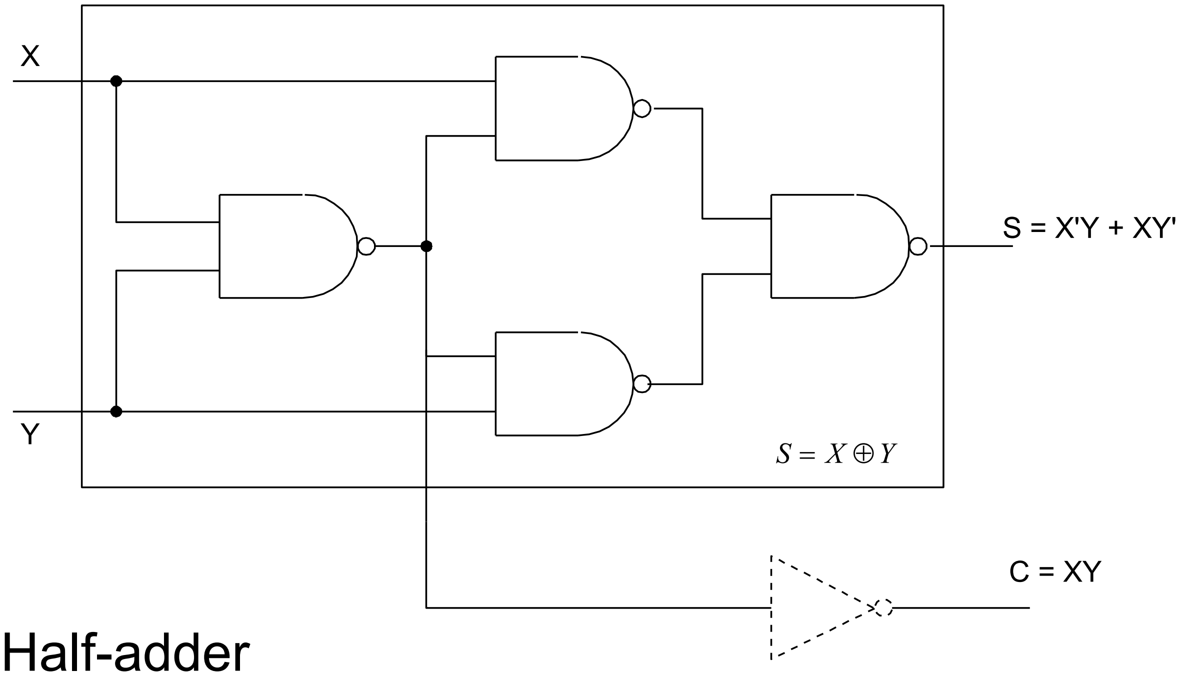


Exercise:

Show that for all circuits:

$$S = X\overline{Y} + \overline{X}Y$$
$$C = X \cdot Y$$

Half-adder NAND gate implementation



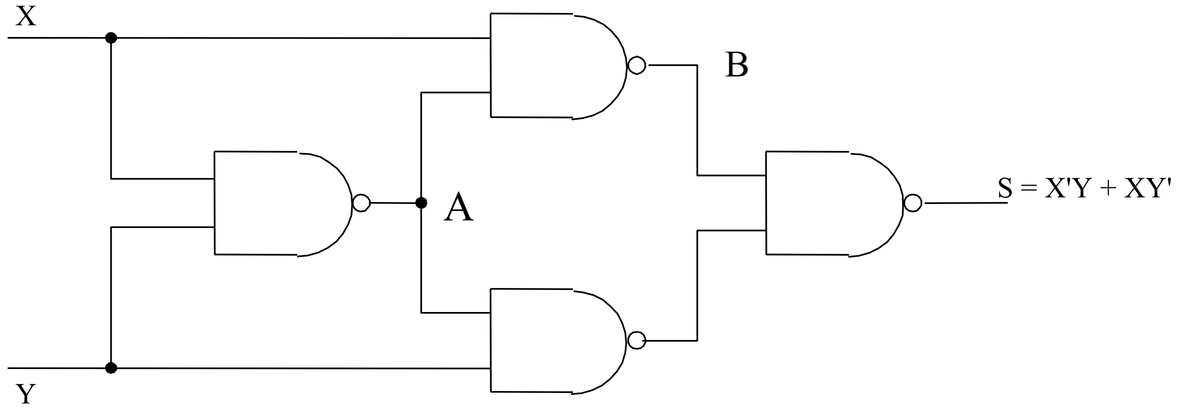
Half-adder

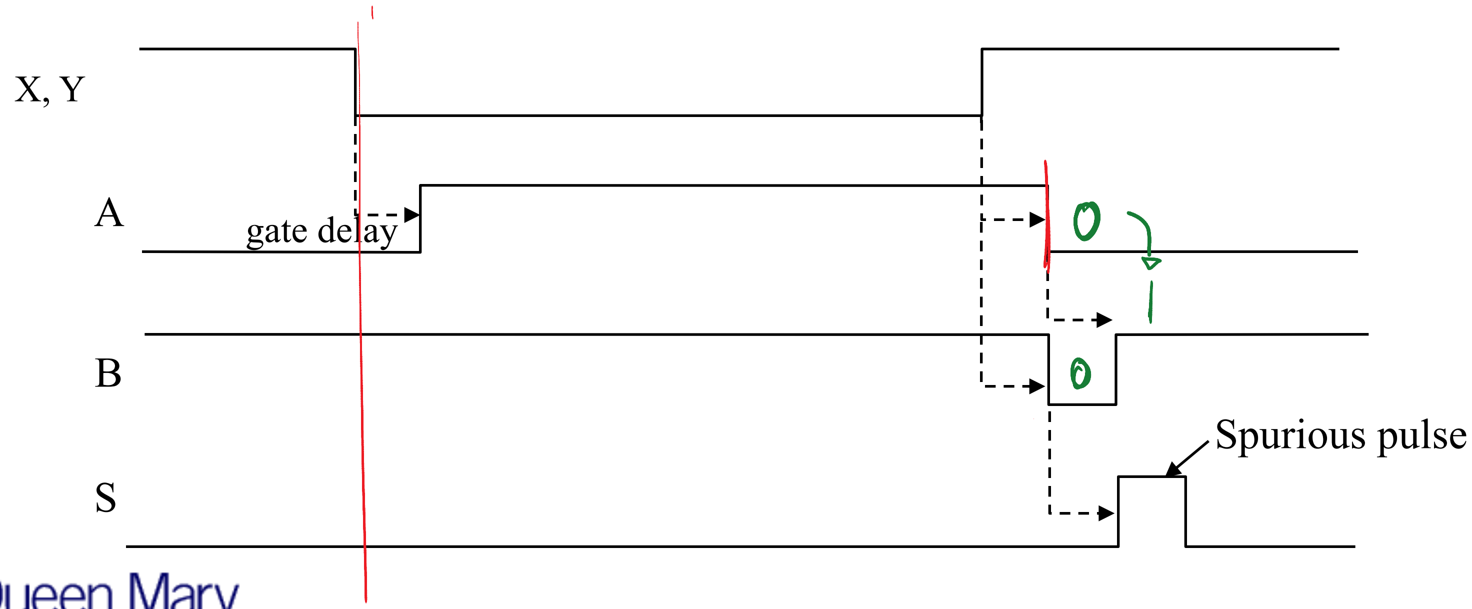
- Arithmetic X + Y
- Max gate delay 3 units

  Sum has 3 units delay

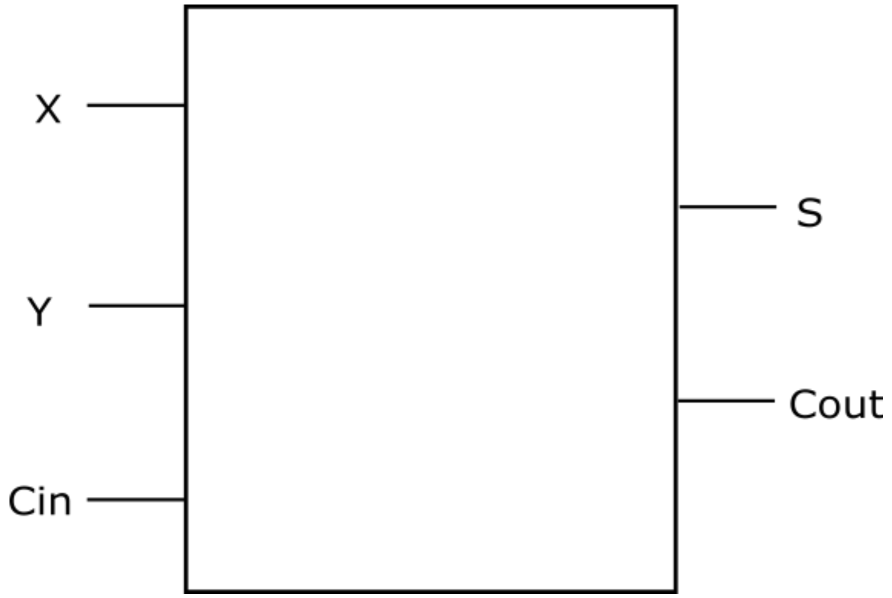
  Carry has 2 units delay (*Carry’* has 1 unit delay)

Effect of Delay





Full Adder

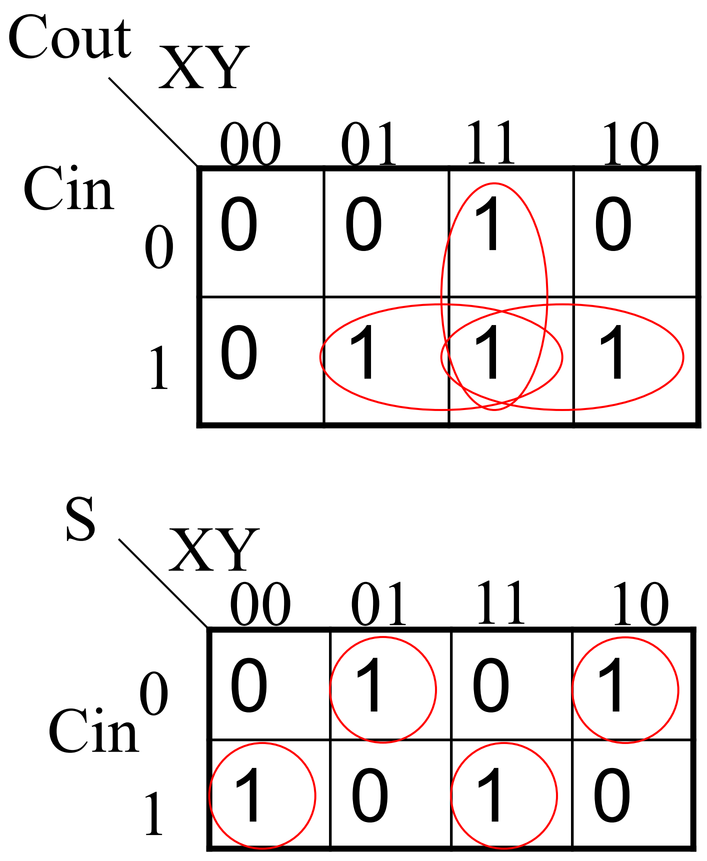


$$s = X \oplus Y \oplus c\_in$$

$$c\_out = (X.Y) + (X.c\_in) + (Y.c\_in)$$

Full Adder

C_{in}	X	Y	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



Cout

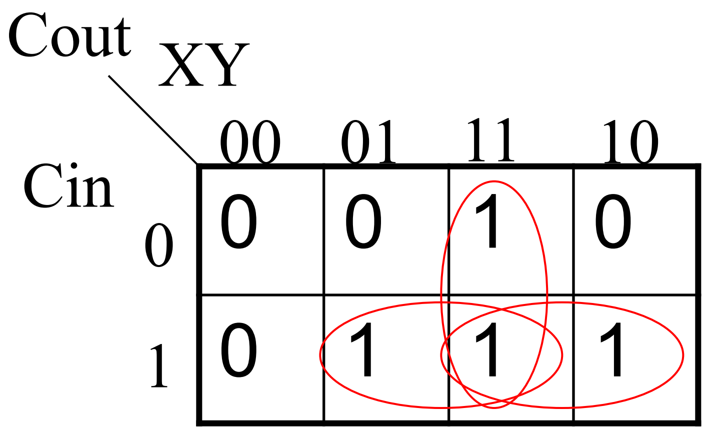
Cin \ XY	00	01	11	10
0	0	0	1	0
1	0	1	1	1

S

Cin \ XY	00	01	11	10
0	0	1	0	1
1	1	0	1	0

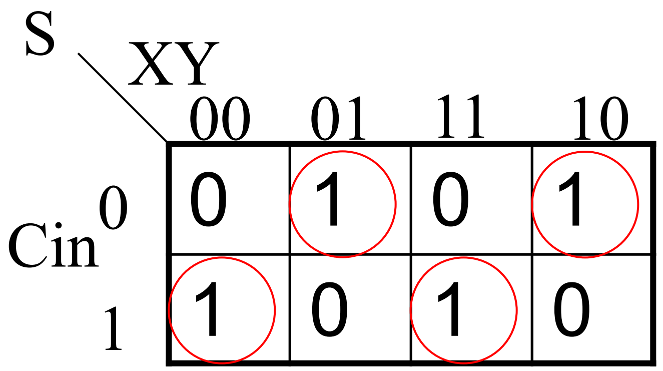
Full Adder

Cout = Cin.X + Cin.Y + X.Y



Cout: Cin \ XY	00	01	11	10
0	0	0	1	0
1	0	1	1	1

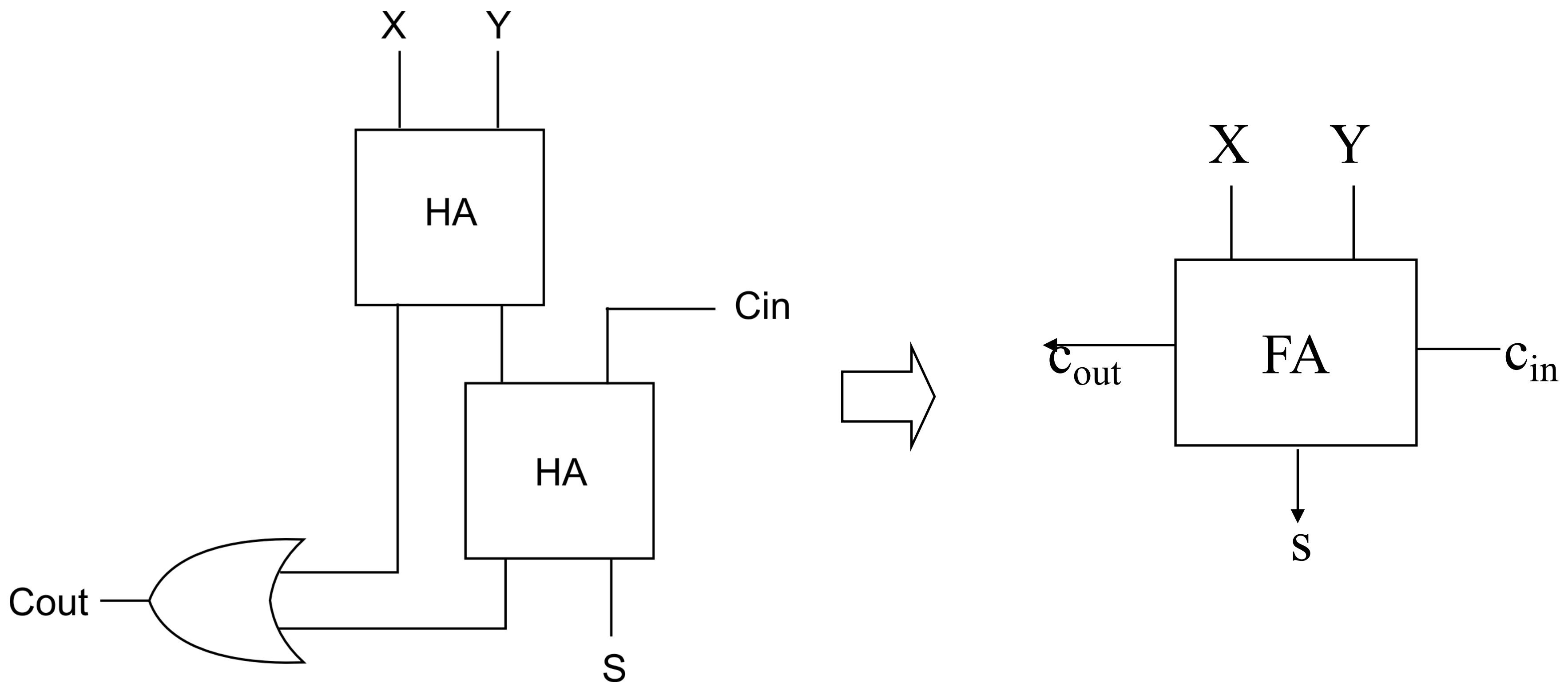
S = Cin’.X’.Y + Cin’.X.Y’
+ Cin.Y’.X’ + Cin.X.Y



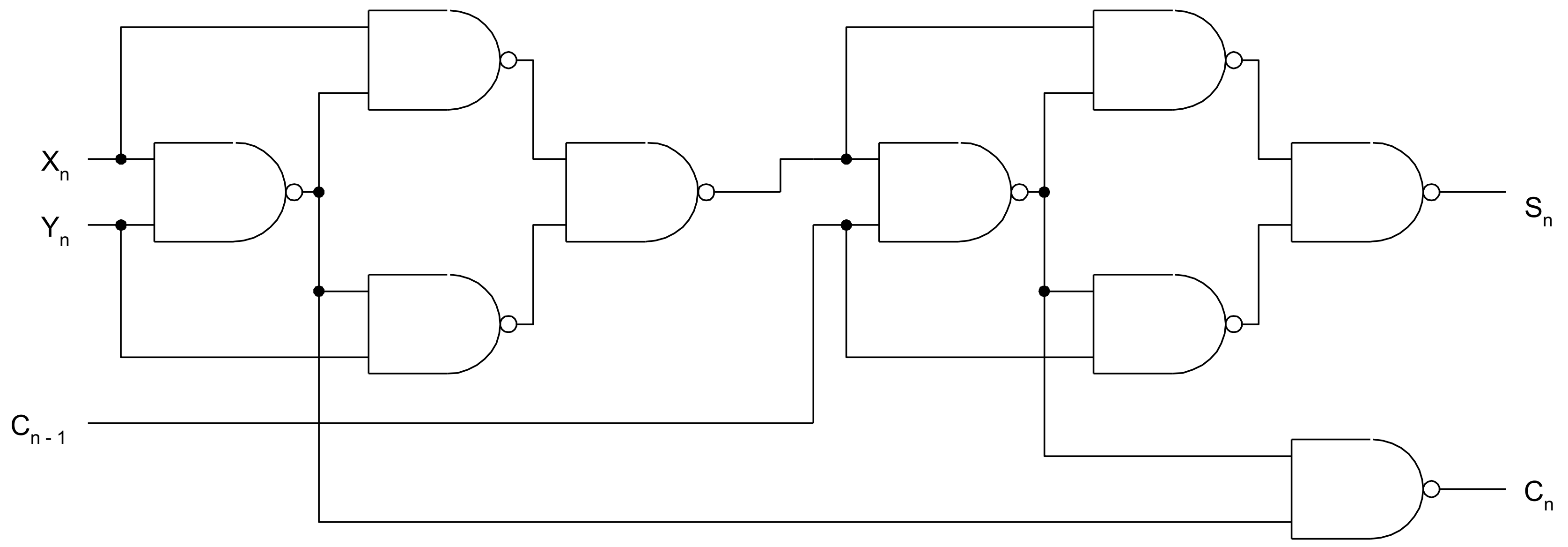
S: Cin \ XY	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Full Adder gate implementations

Can be implemented with 2 half adders and an OR gate:

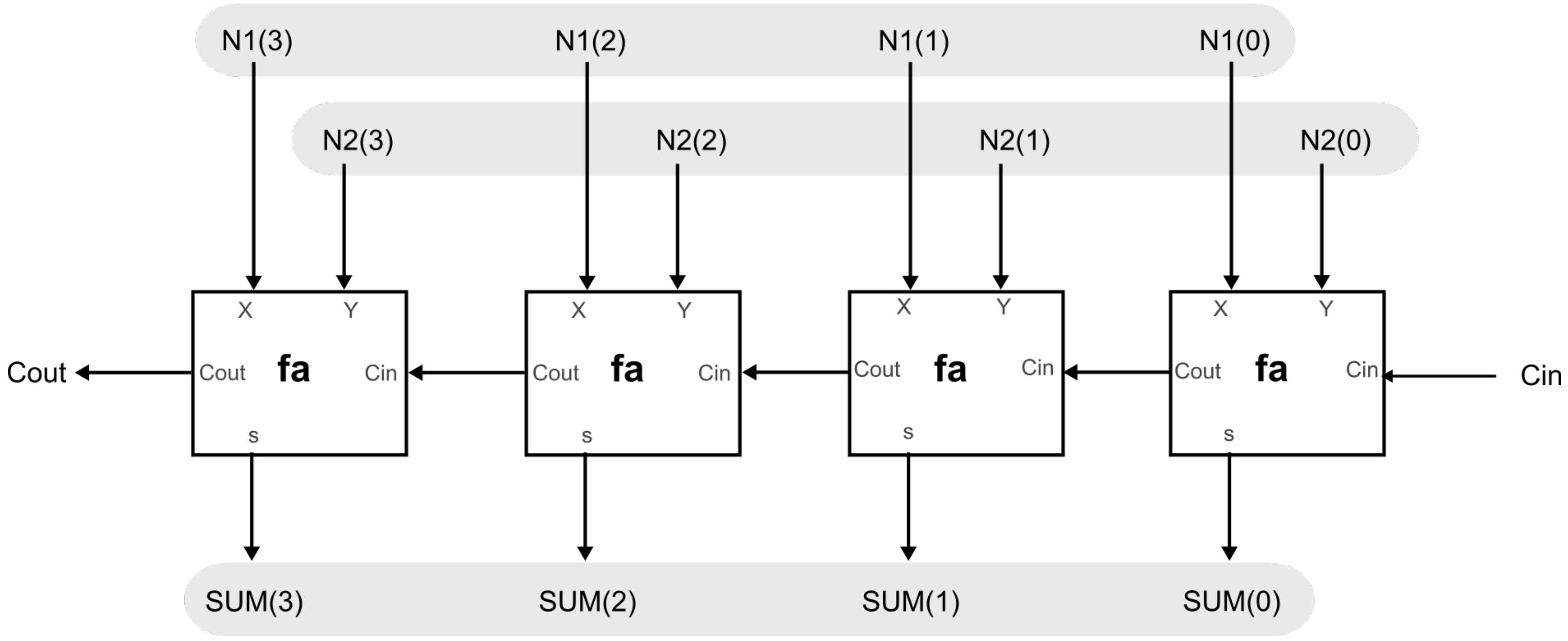


Full Adder: NAND-based Ex-OR blocks

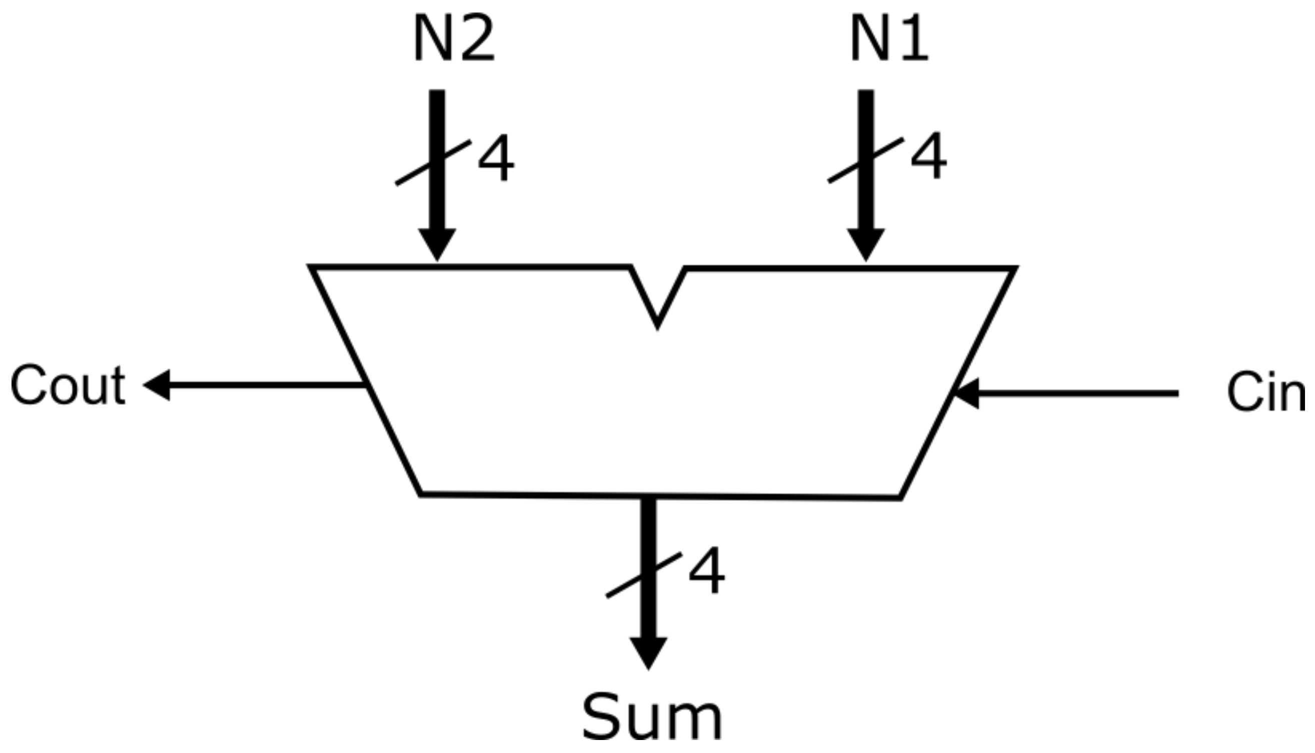


Consists essentially of two half-adders (XORs with Carry outputs)

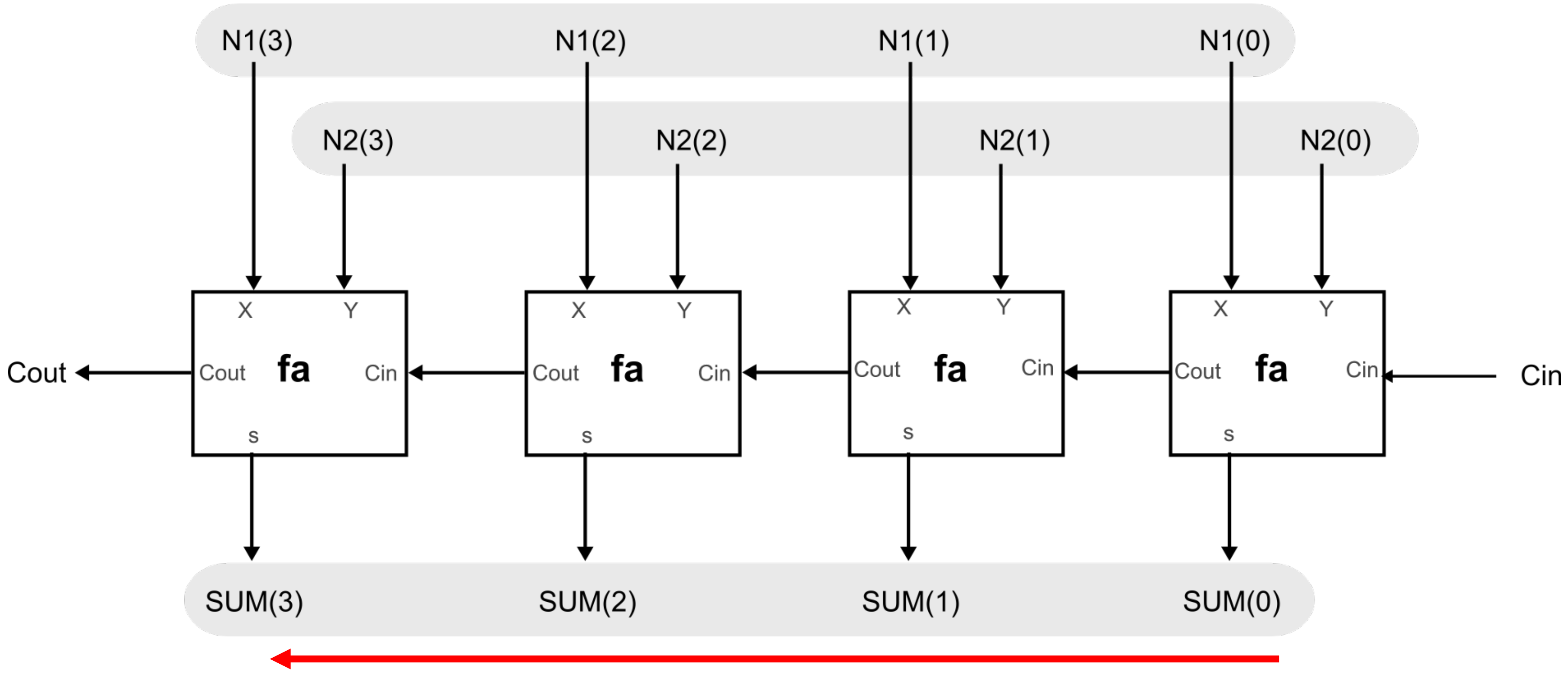
4-bit Parallel Adder



4-bit Parallel Adder

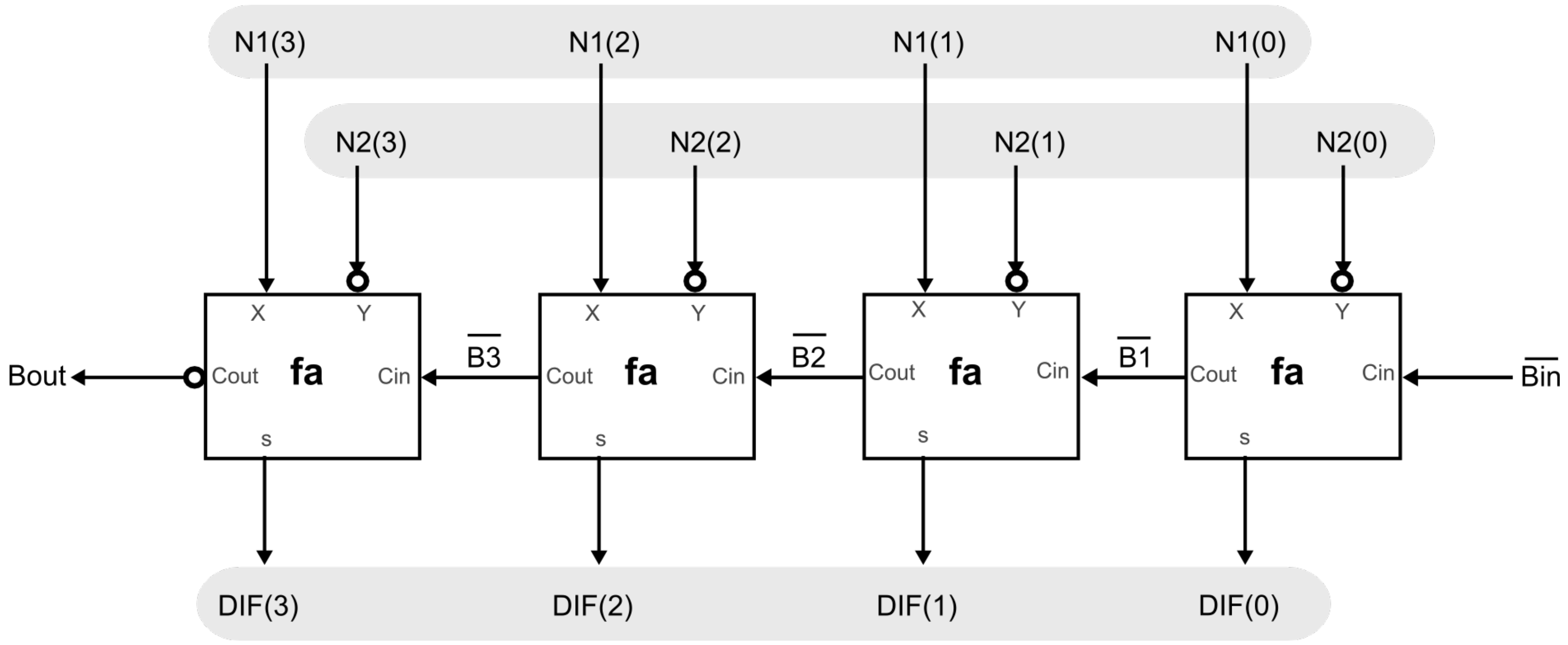


Ripple Delay



Delay caused by carry outputs at each stage

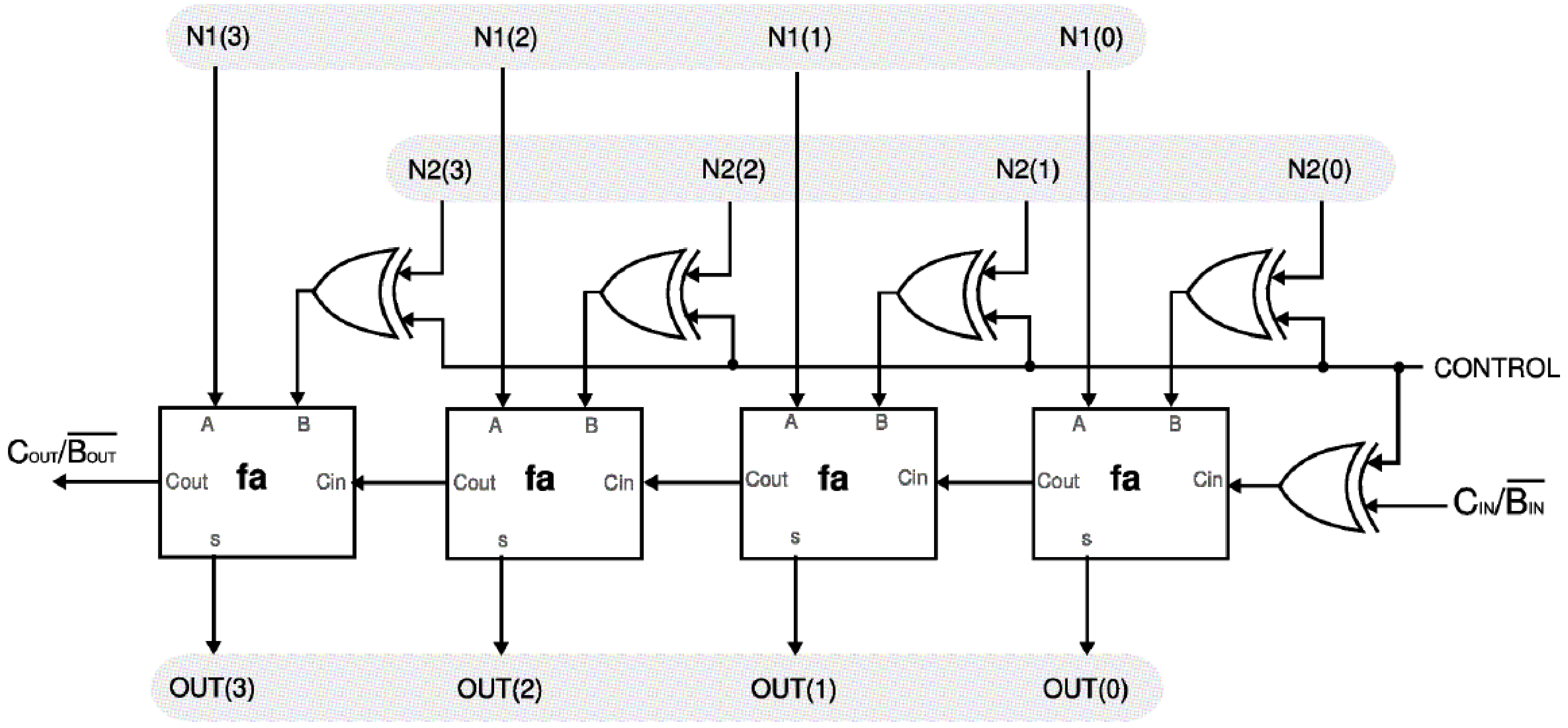
4-bit Ripple Subtractor using Adders



Use two's complement: A-B= A+(B'+1)

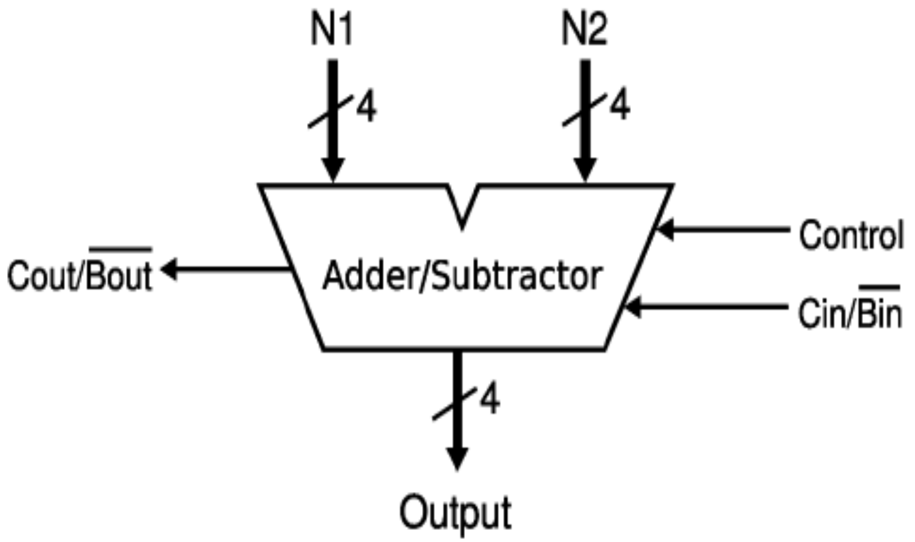
Queen Mary University of London

4-bit Adder/Subtractor



Queen Mary University of London

4-bit Adder/Subtractor

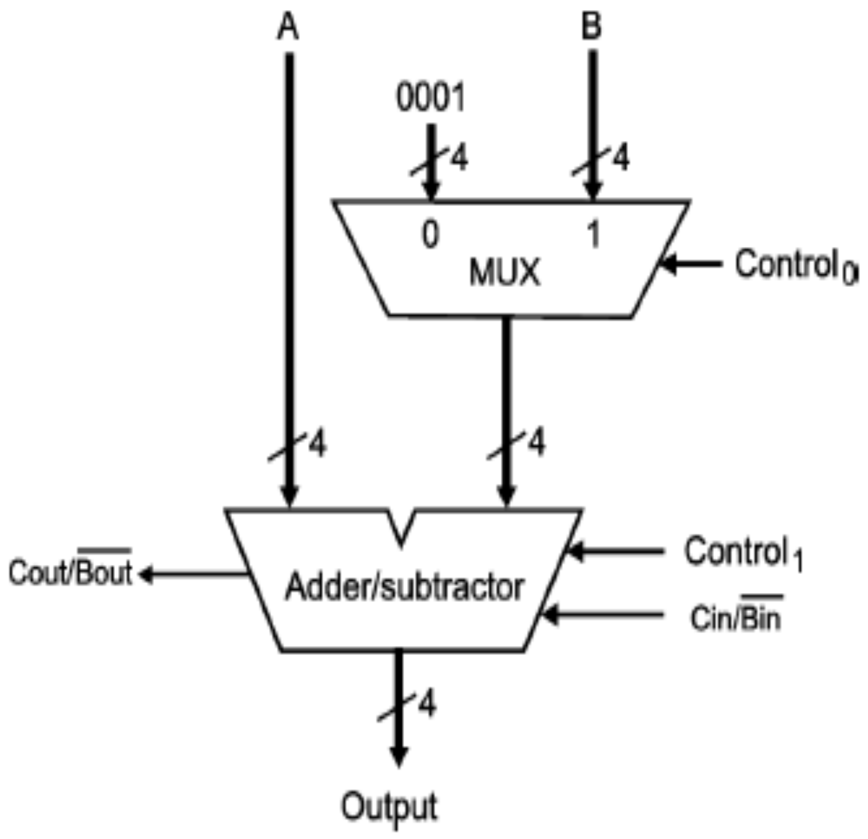


Control	Output Function
0	Add (N1 + N2)
1	Subtract (N1 – N2)

Queen Mary University of London

An Arithmetic Unit

Control 1	Control 0	Arithmetic Function
0	0	A + 1
0	1	A + B
1	0	A - 1
1	1	A - B

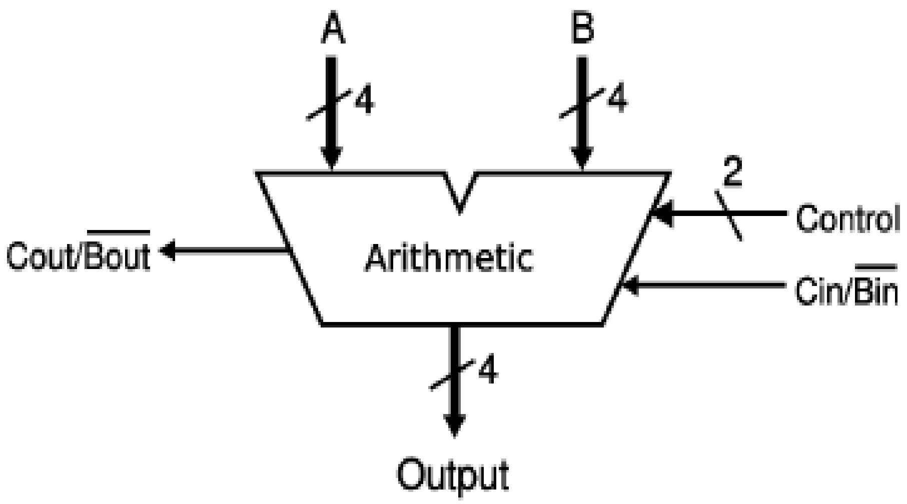


Queen Mary University of London

An Arithmetic Unit

- Can add and subtract
- Can also make it increment and decrement by 1

Control 1	Control 0	Arithmetic Function
0	0	A + 1
0	1	A + B
1	0	A - 1
1	1	A - B



A
4
B
4
2
Control
Arithmetic
$\text{Cout}/\overline{\text{Bout}}$
$\text{Cin}/\overline{\text{Bin}}$
4
Output