Carry Skip Adder (1A)

-
-

https://en.wikipedia.org/wiki/AND_gate
https://en.wikipedia.org/wiki/OR_gate
https://en.wikipedia.org/wiki/XOR_gate
https://en.wikipedia.org/wiki/NAND_gate

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

Carry Lookahead Adder

Carry Lookahead Adder

$p_i = a_i \oplus b_i$

$g_i = a_i \wedge b_i$

$c_1 = g_0 + p_0 \wedge c_0$

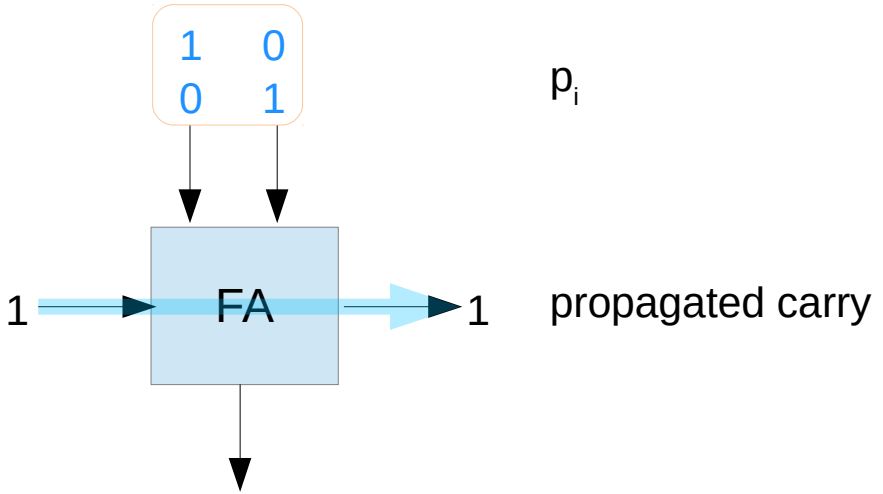
$c_2 = g_1 + p_1 \wedge c_1$

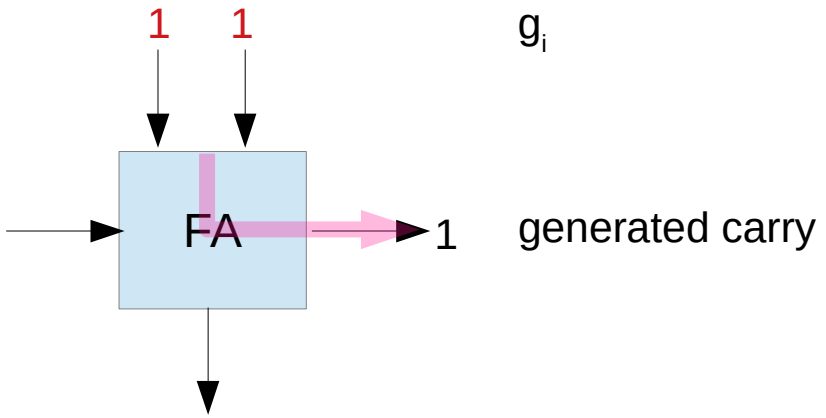
$c_3 = g_2 + p_2 \wedge c_2$

$c_4 = g_3 + p_3 \wedge c_3$

propagated carry

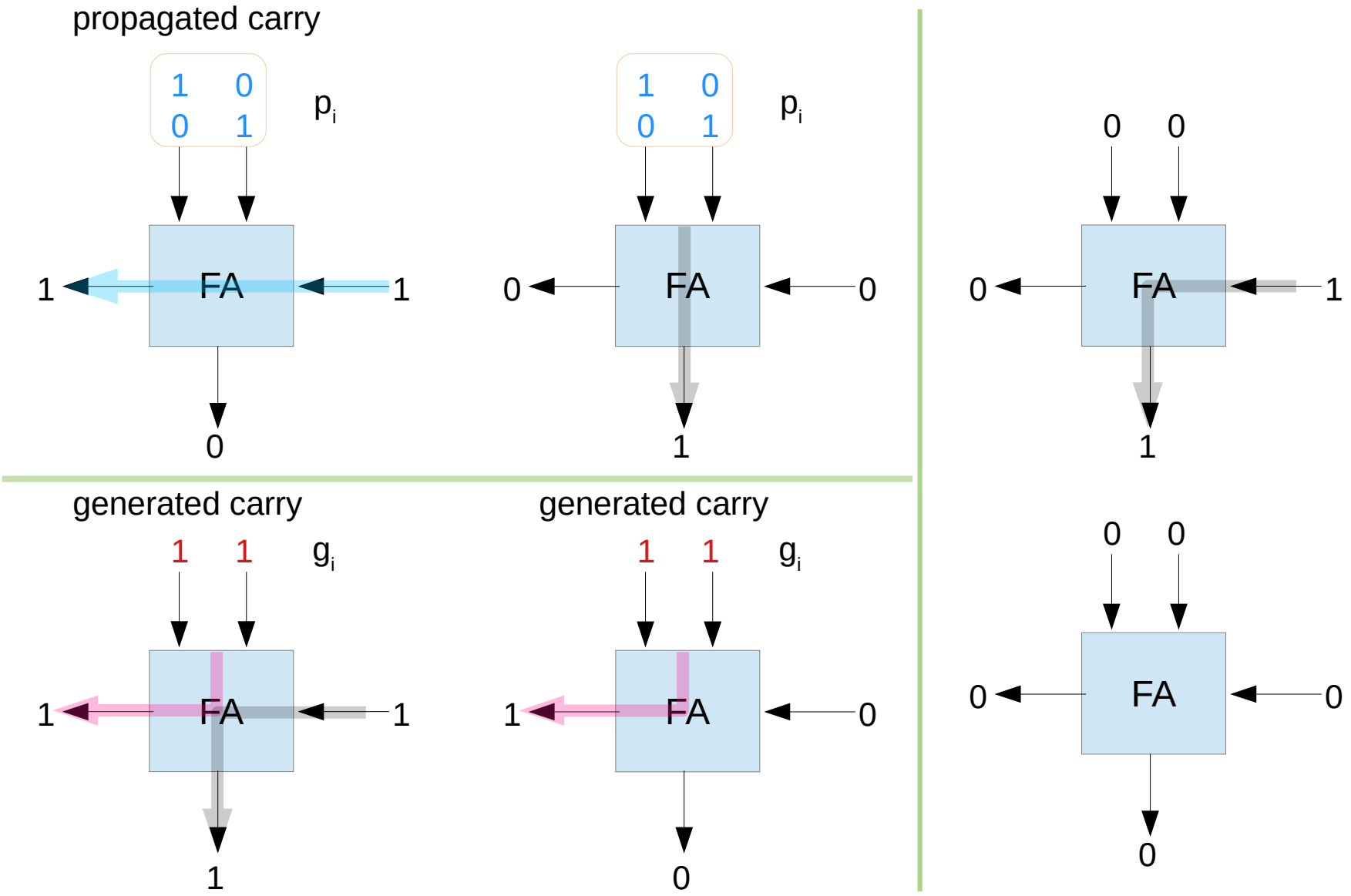
generated carray



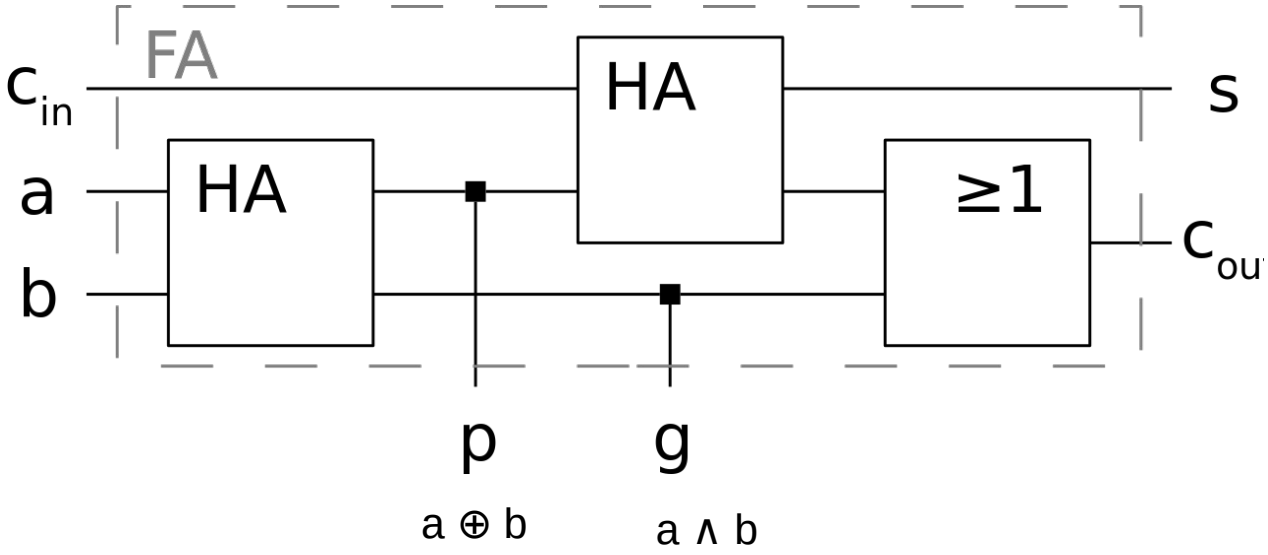


https://en.wikipedia.org/wiki/Carry-skip_adder

Propagated and Generated Carries



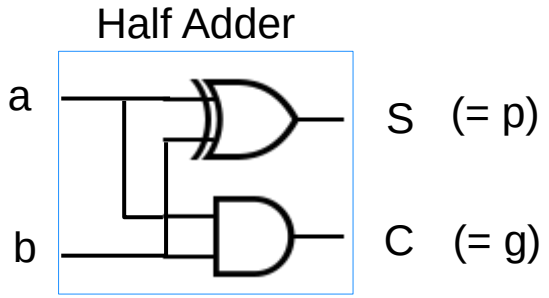
FA with P & G



$s = p_i \oplus c_i$

$c_{out} = g_i + p_i \wedge c_i$

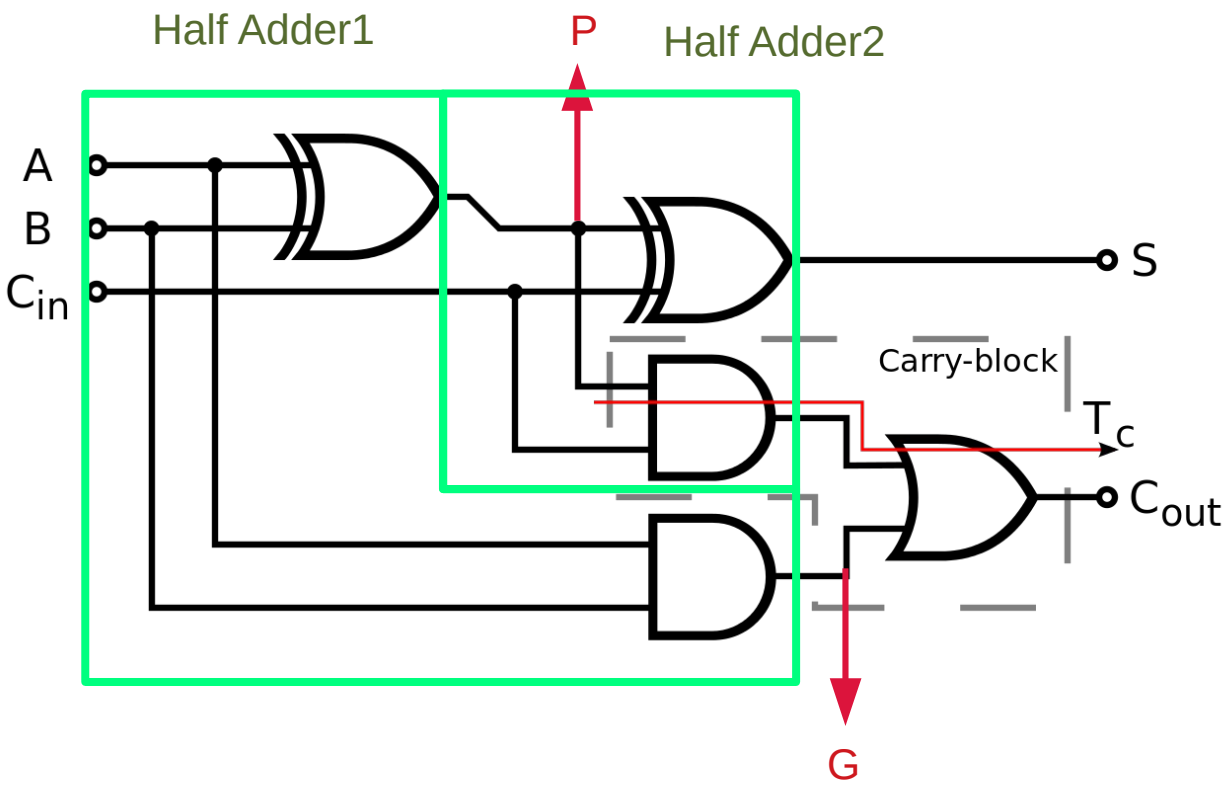
Half Adder
S = a ⊕ b
C = a ∧ b

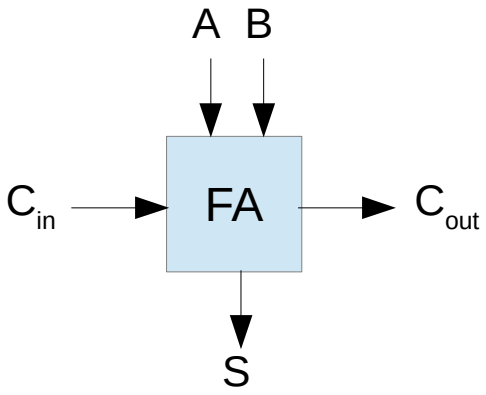


https://en.wikipedia.org/wiki/Carry-skip_adder

Full adder with additional generate and propagate signals.

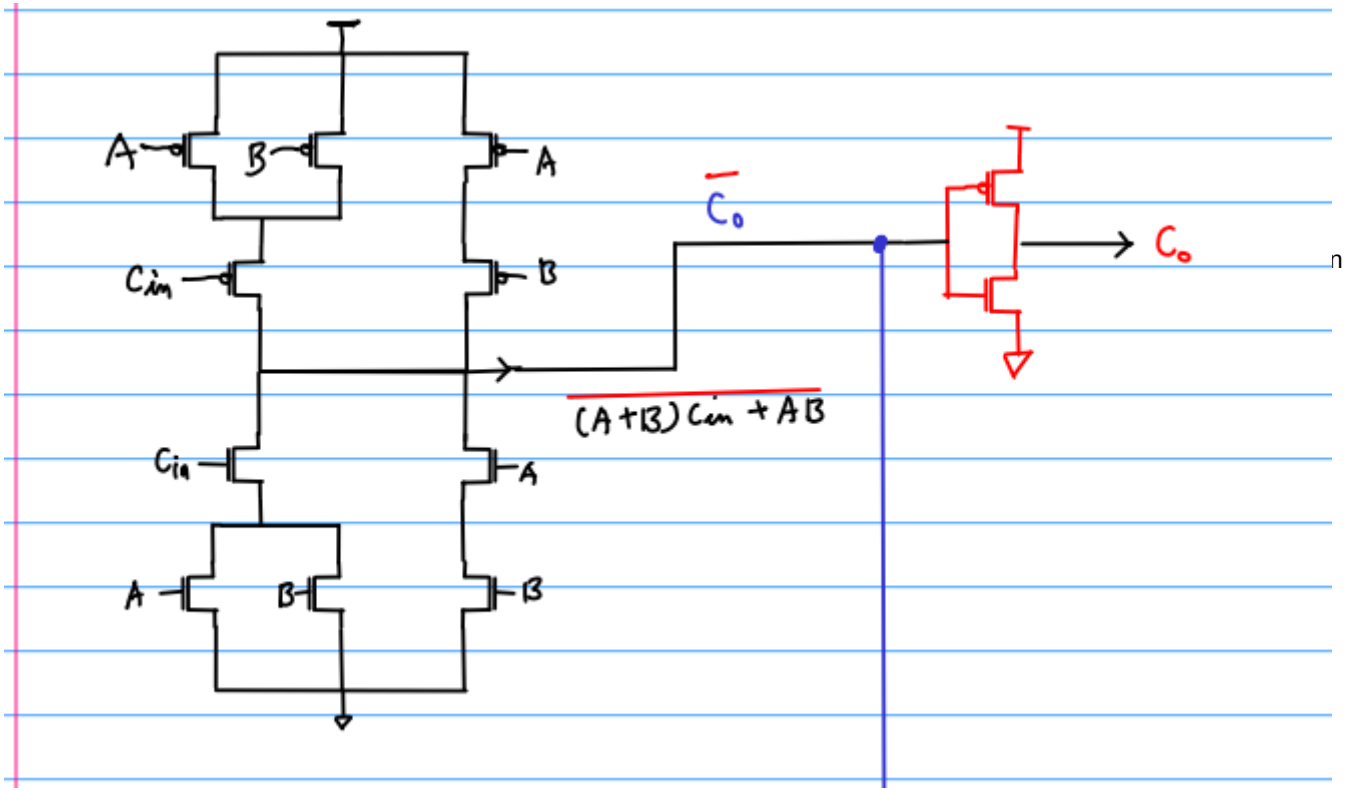
Gate Level Full Adder

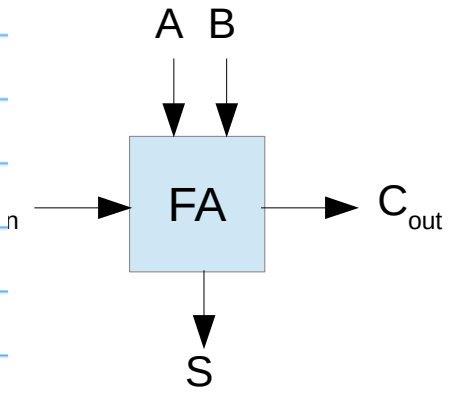




Www.cs.tufts.edu/103/notes/Lecture14(Adders-2).pdf

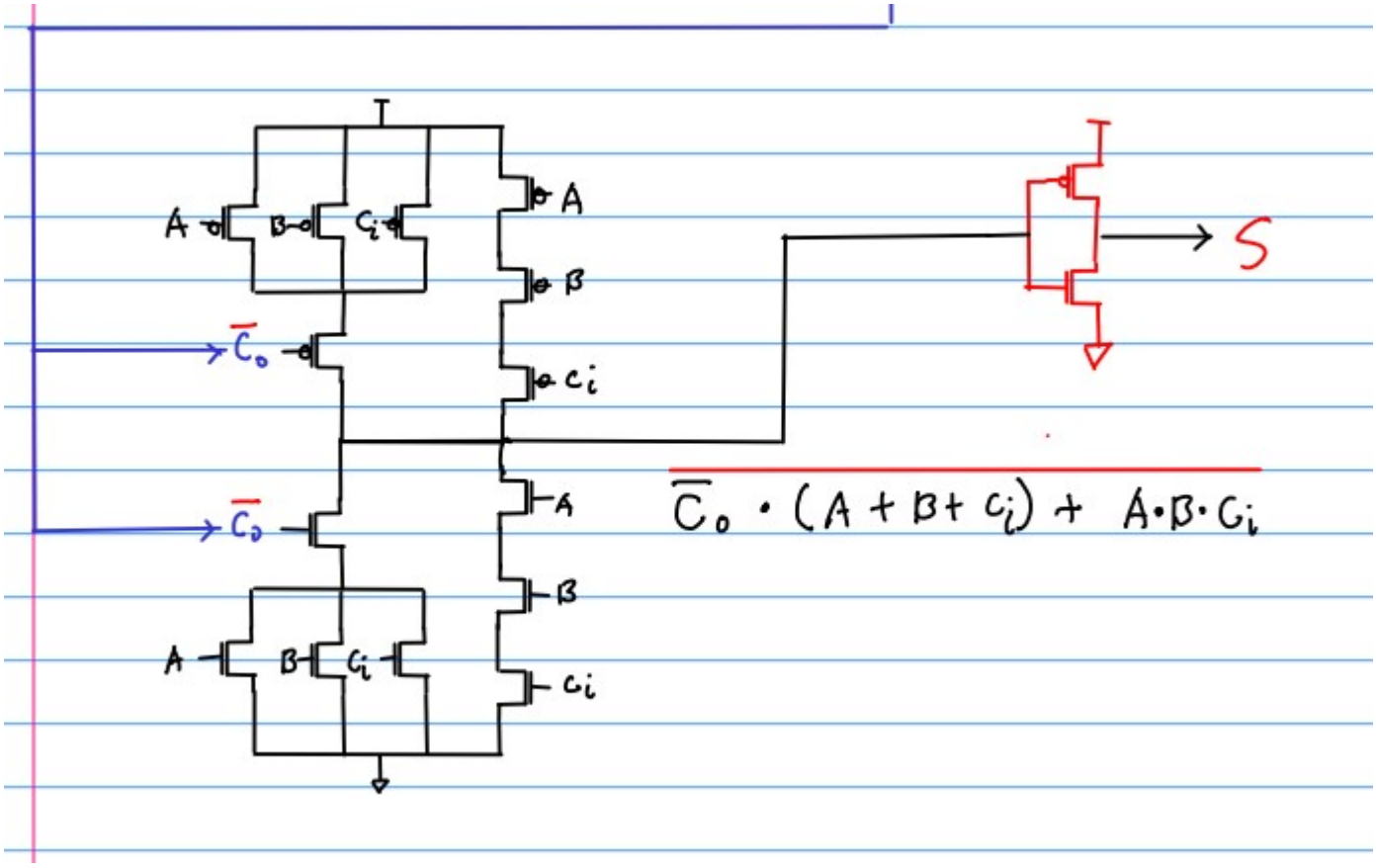
CMOS Level Full Adder

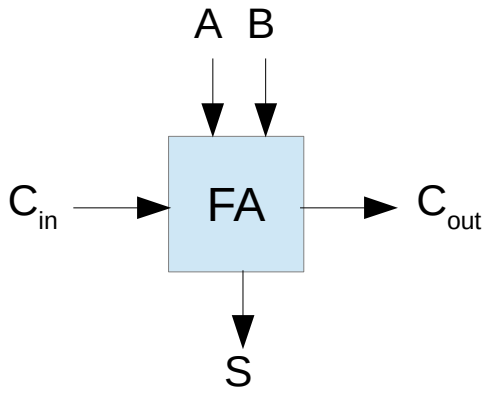




https://upload.wikimedia.org/wikiversity/en/1/18/RCA.Note.H.1.20151215.pdf

CMOS Level Full Adder

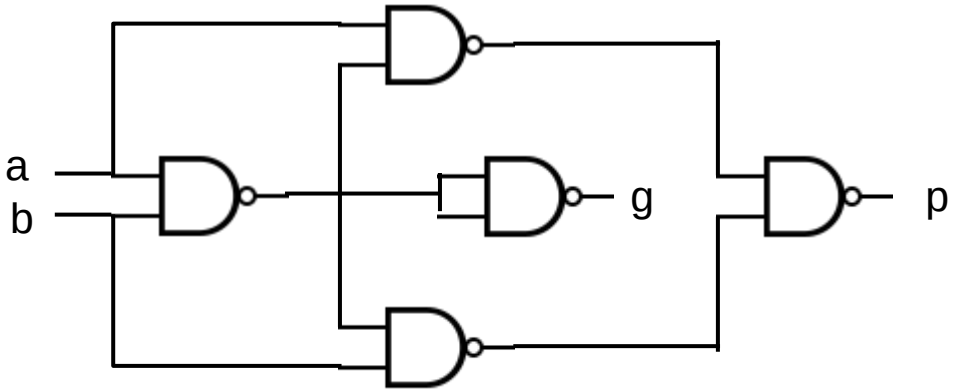


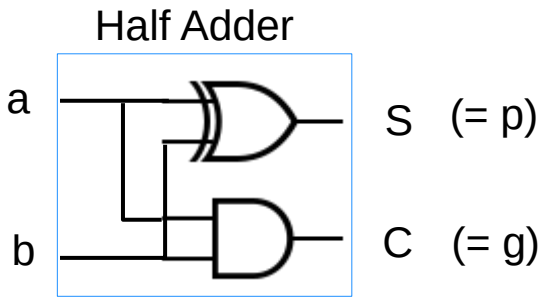


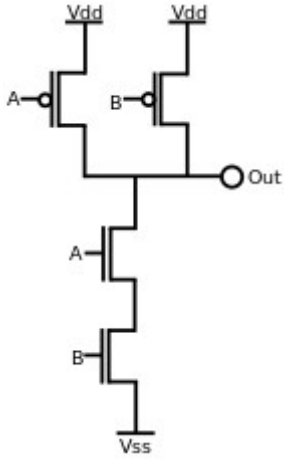
https://upload.wikimedia.org/wikiversity/en/1/18/RCA.Note.H.1.20151215.pdf

Half Adder Implementation

Half Adder
S = a ⊕ b
C = a ∧ b

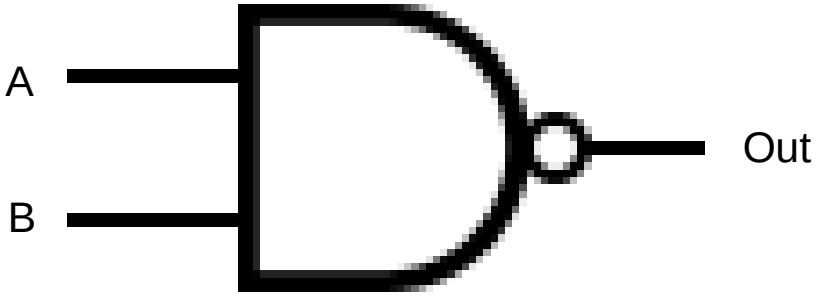


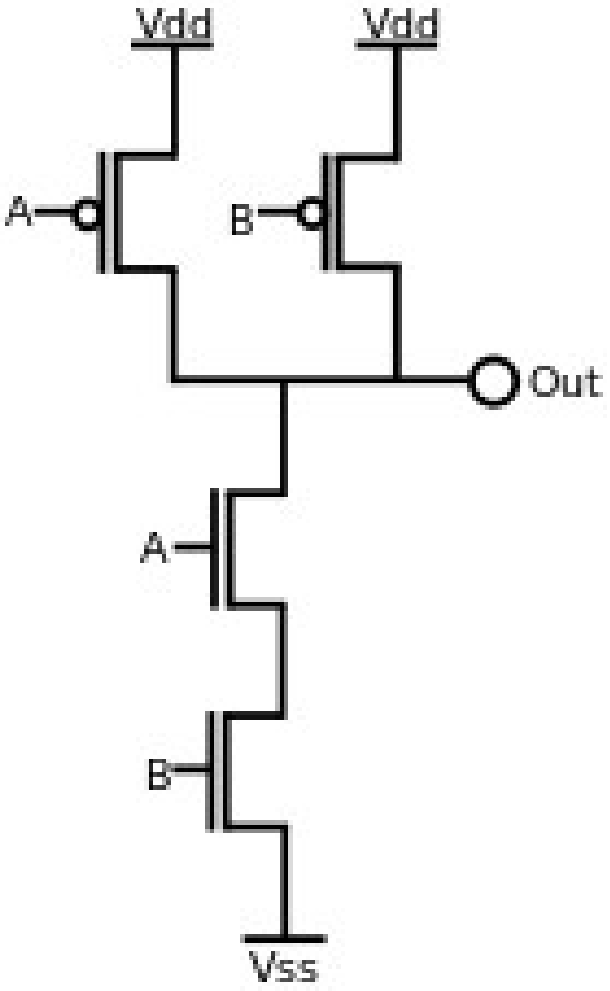




https://people.eecs.berkeley.edu/~newton/Classes/CS150sp98/lectures/week6_2/sld007.htm

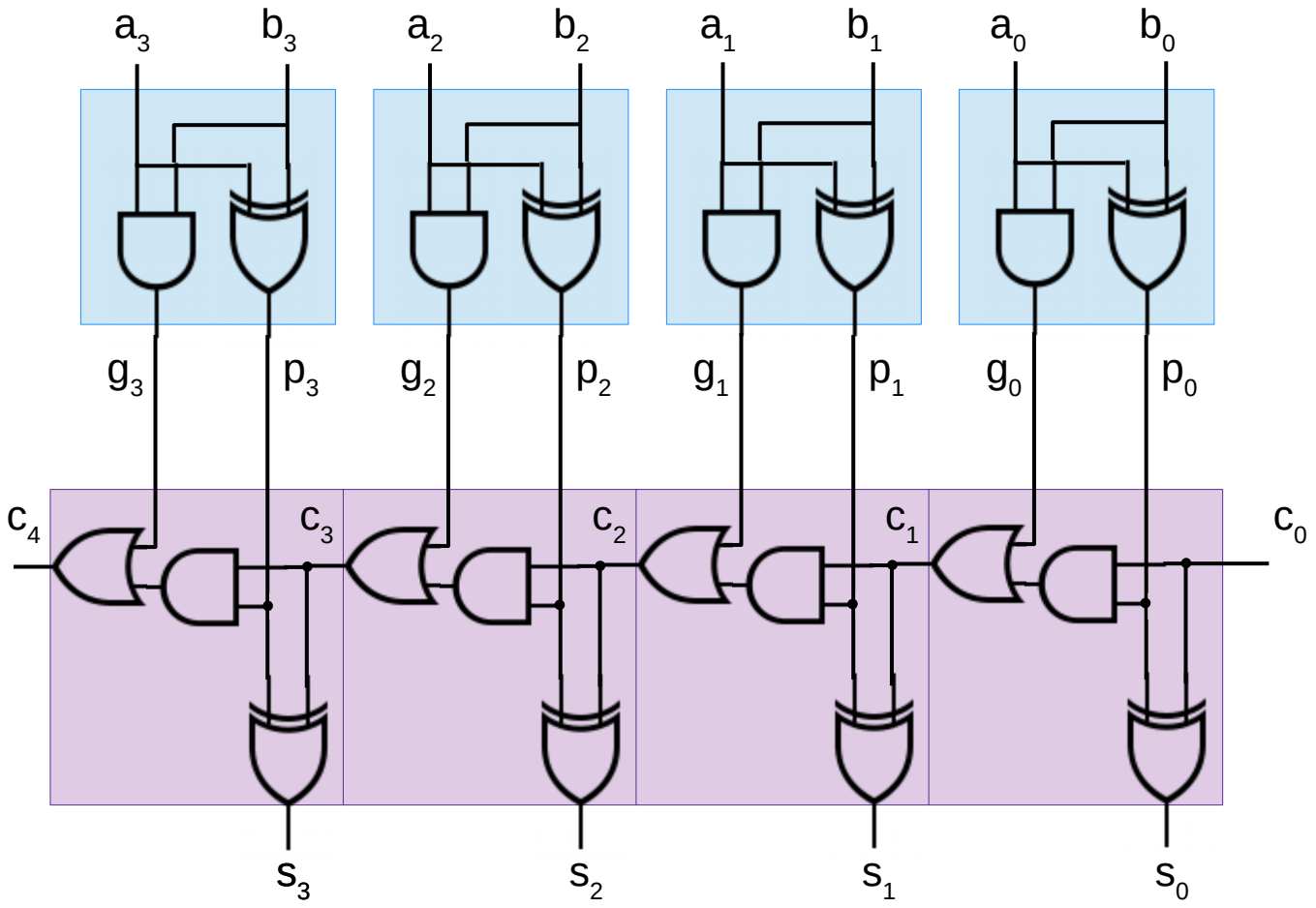
CMOS NAND Gate





https://people.eecs.berkeley.edu/~newton/Classes/CS150sp98/lectures/week6_2/sld007.htm

4-bit Full Adder with P and G



Half Adder

$p_i = a_i \oplus b_i$

$g_i = a_i \wedge b_i$

$c_{i+1} = g_i + p_i c_i$

$s_i = p_i \oplus c_i$

https://upload.wikimedia.org/wikiversity/en/1/18/RCA.Note.H.1.20151215.pdf

FA with P & G

For each operand input bit pair (a_i , b_i)

the propagate-conditions $p_i = a_i \oplus b_i$ are determined using an XOR-Gate .

When all propagate-conditions are true,

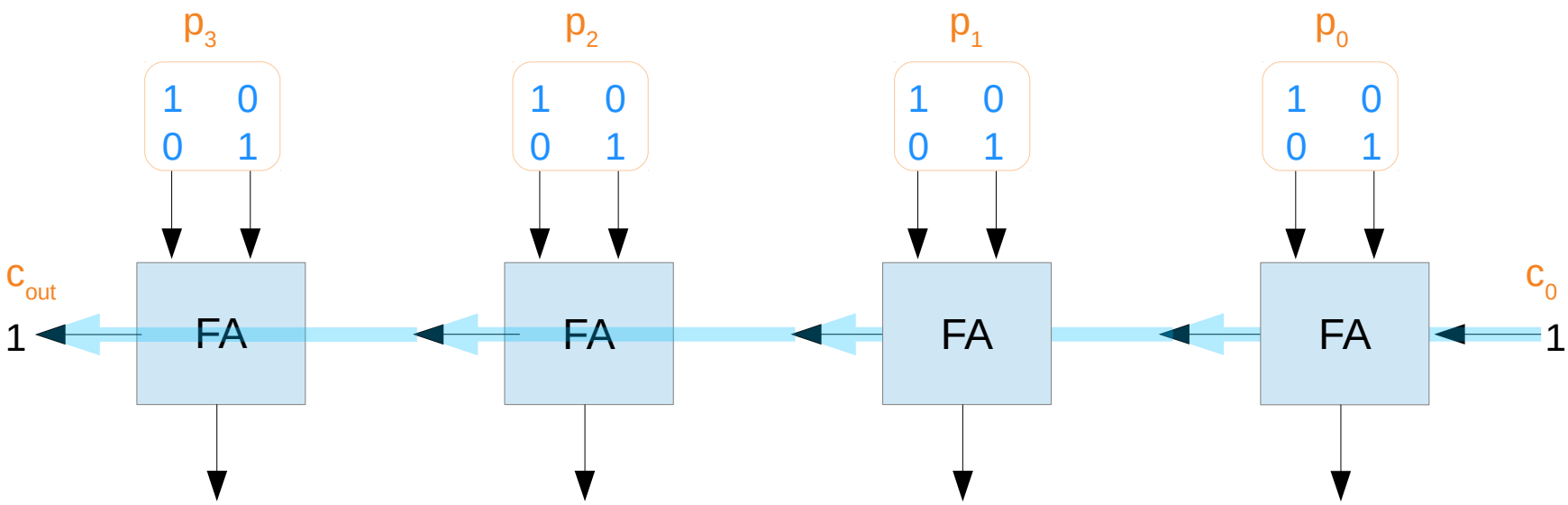
$$s = p_{n-1} \wedge p_{n-2} \wedge \cdots \wedge p_1 \wedge p_0 = p_{[0:n-1]}$$
$$= (a_{n-1} \oplus b_{n-1}) \wedge (a_{n-2} \oplus b_{n-2}) \wedge \cdots \wedge (a_1 \oplus b_1) \wedge (a_0 \oplus b_0)$$

then the carry-in bit c_0 determines the carry-out bit.

c_0 can be propagated to c_{out} only when s = 1

https://en.wikipedia.org/wiki/Carry-skip_adder

C_0 propagation condition



c_0 can be propagated to c_{out} only when s = 1

$s = p_{n-1} \wedge p_{n-2} \wedge \cdots \wedge p_1 \wedge p_0 = p_{[0:n-1]}$

$= (a_{n-1} \oplus b_{n-1}) \wedge (a_{n-2} \oplus b_{n-2}) \wedge \cdots \wedge (a_1 \oplus b_1) \wedge (a_0 \oplus b_0)$

https://en.wikipedia.org/wiki/Carry-skip_adder

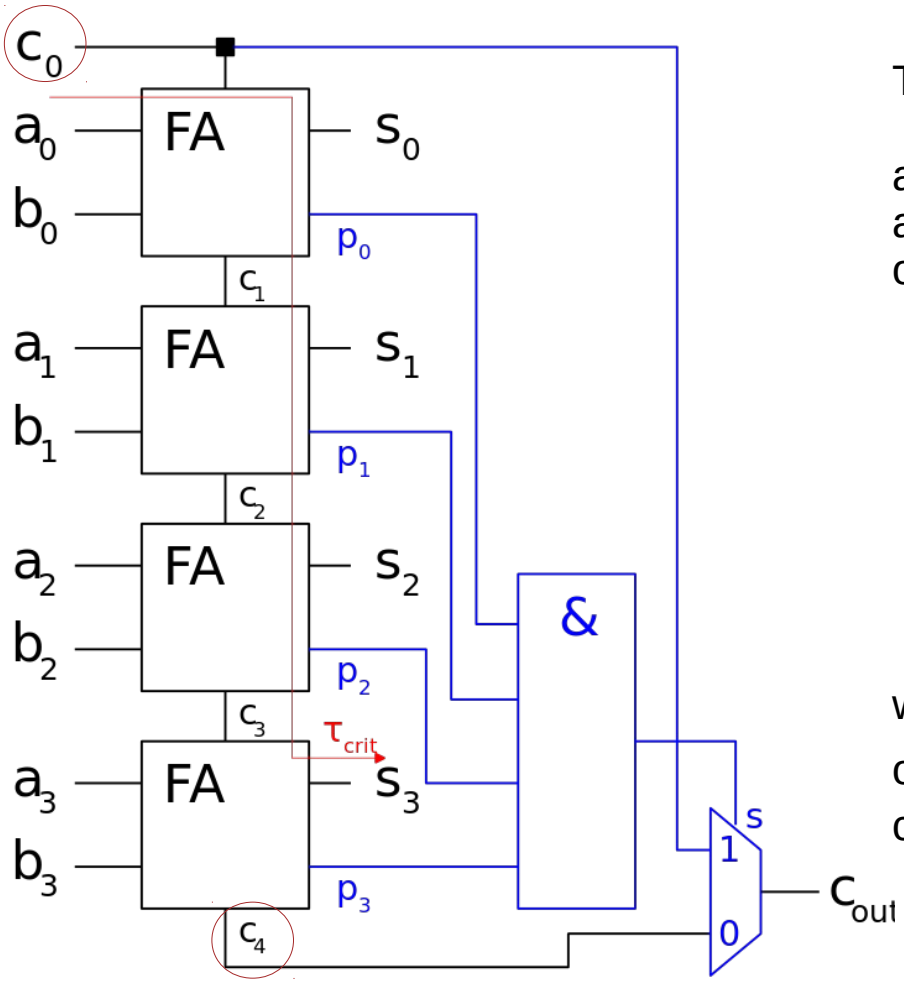
FA with P & G

The n-bit-carry-skip adder consists of
a n-bit **carry-ripple-chain**,
a n-input **AND-gate** and
one **multiplexer**.
Each propagate bit p_i that is provided by the carry-ripple-chain
is connected to the n-input AND-gate.
The resulting bit is used as the select bit of a multiplexer
that switches either the last carry-bit c_n or the carry-in c_0
to the carry-out signal c_{out}

$$s = p_{n-1} \wedge p_{n-2} \wedge \cdots \wedge p_1 \wedge p_0 = p_{[0:n-1]}$$

https://en.wikipedia.org/wiki/Carry-skip_adder

4-bit Carry Skip Adder



The n-bit-carry-skip adder consists of

a n-bit **carry-ripple-chain**,
a n-input **AND-gate** and
one **multiplexer**.

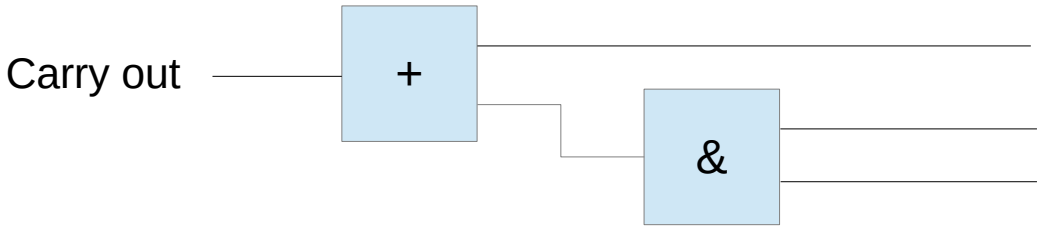
a multiplexer switches
either the last carry-bit c_n or the carry-in c_0
to the carry-out signal c_{out}

$$s = p_{n-1} \wedge p_{n-2} \wedge \cdots \wedge p_1 \wedge p_0 = p_{[0:n-1]}$$

when $s = 1$, $c_{out} \leftarrow c_0$
otherwise, internally generated carries
can be propagated to c_{out}

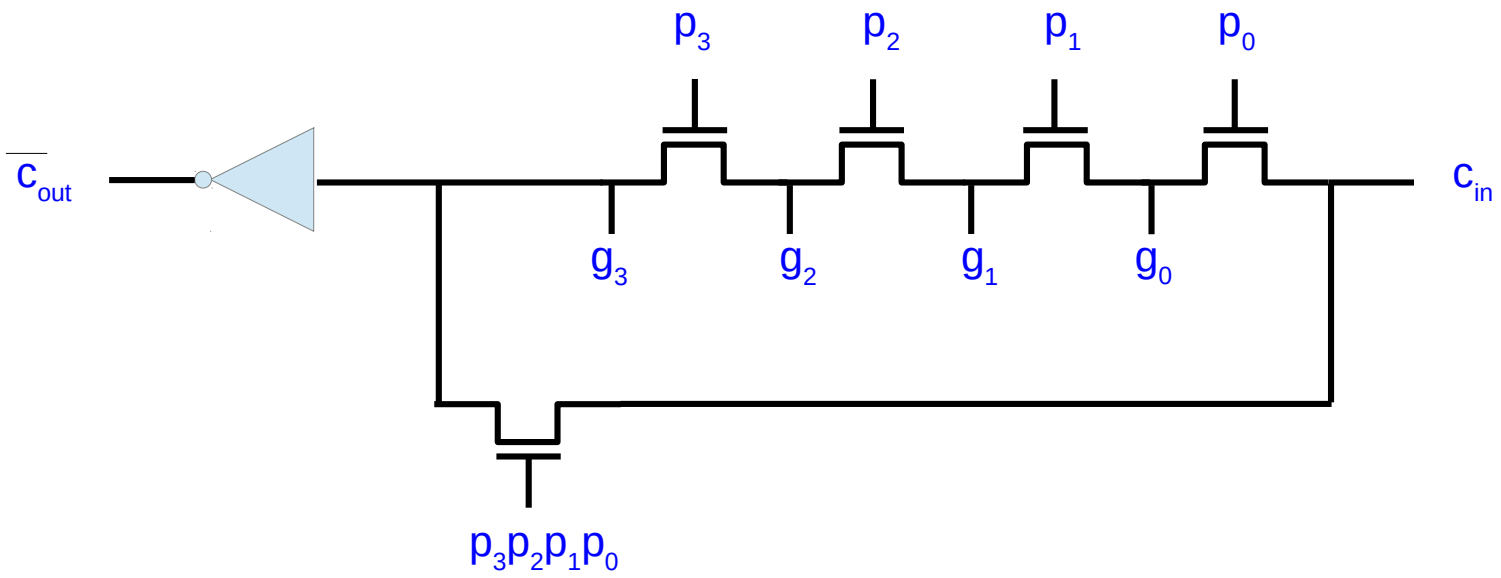
https://en.wikipedia.org/wiki/Carry-skip_adder

Carry Skip Chain Implementation



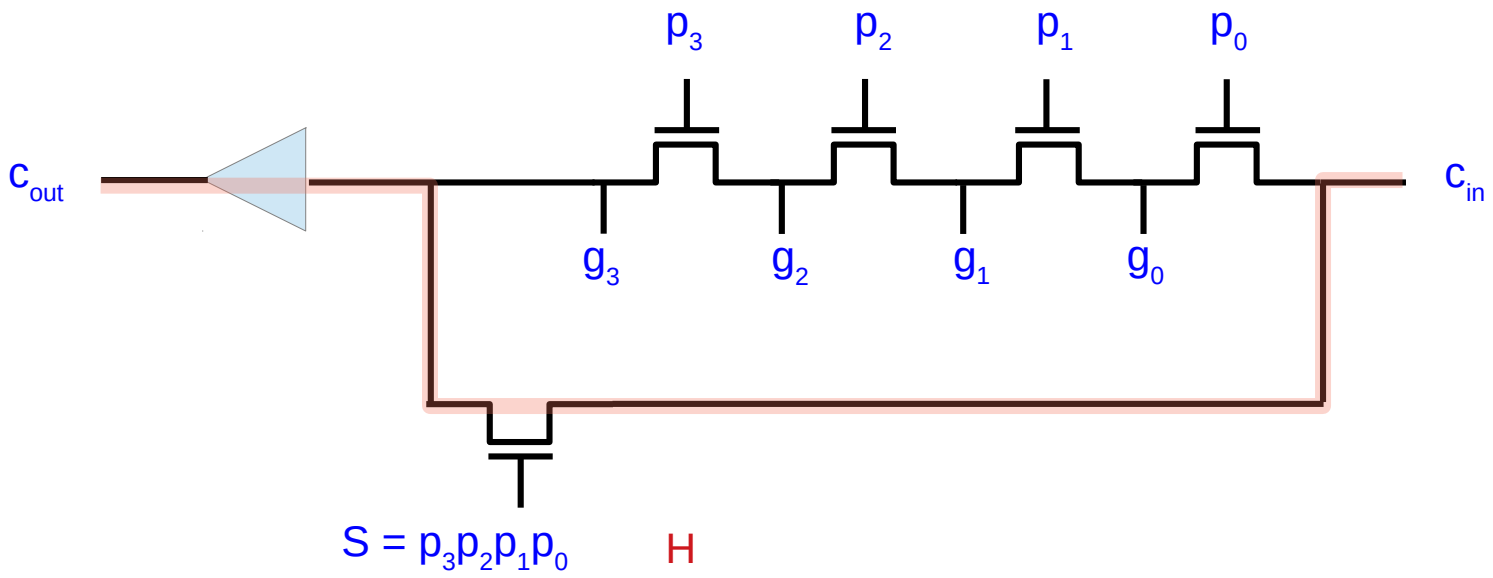
$p_3 \wedge p_2 \wedge p_1 \wedge p_0$

Pass Transistor Logic



Www.cs.tufts.edu/103/notes/Lecture14(Adders-2).pdf

Carry Skip Chain Implementation

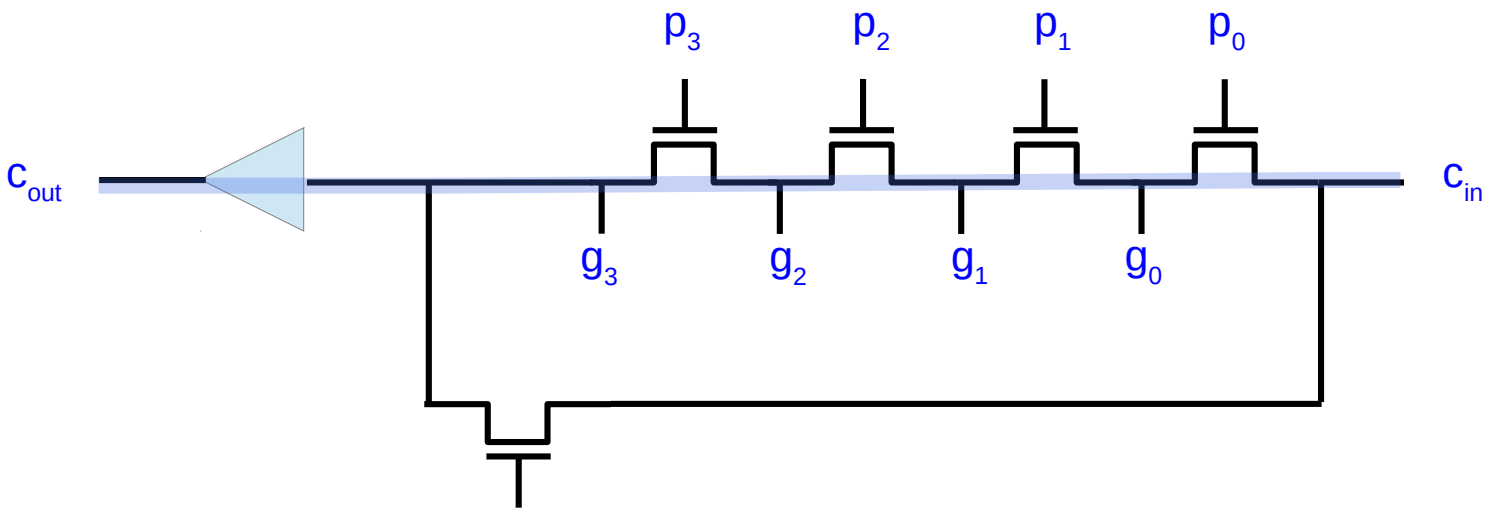


p_i =H ⇒ g_i =L

S =H ⇒ All g_i =L, i=0,1,2,3

Www.cs.tufts.edu/103/notes/Lecture14(Adders-2).pdf

Carry Skip Chain Implementation



$S = p_3p_2p_1p_0$ L

$g_i = H$ Pull Up to VDD

$g_i = L$ Pull Down to GND

$p_3p_2p_1g_0$

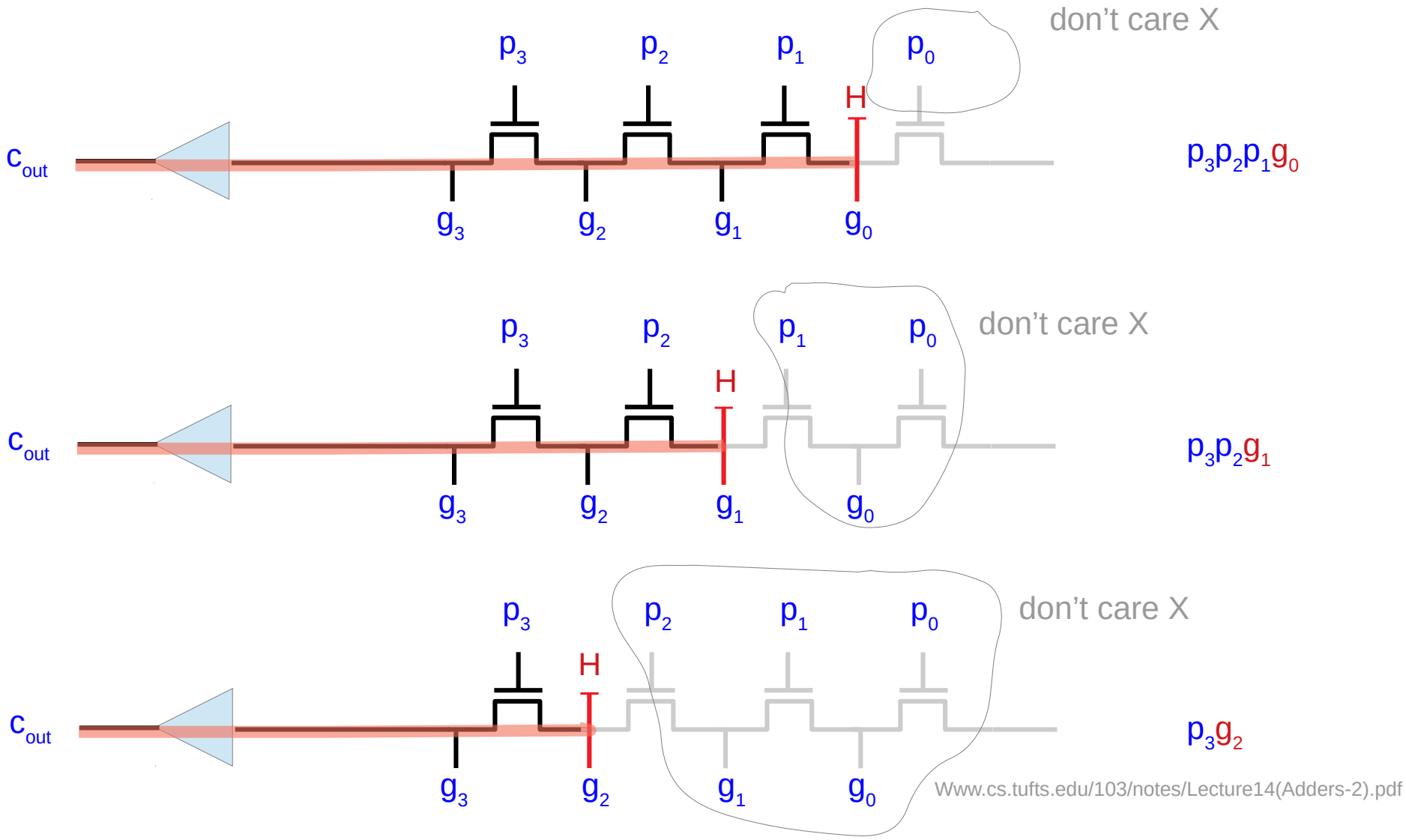
$p_3p_2g_1$

p_3g_2

g_3

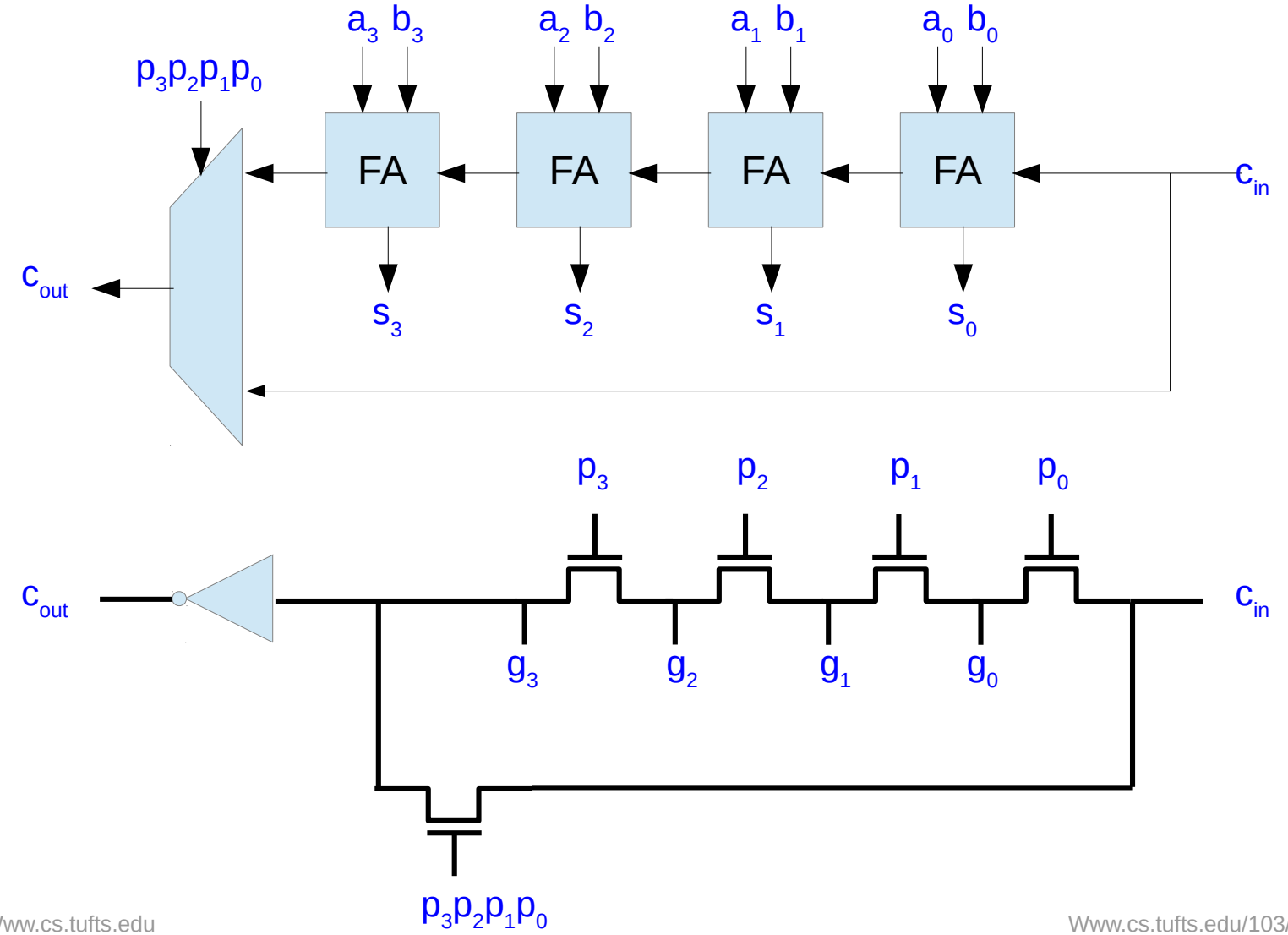
Www.cs.tufts.edu/103/notes/Lecture14(Adders-2).pdf

Carry Skip Chain Implementation



Www.cs.tufts.edu/103/notes/Lecture14(Adders-2).pdf

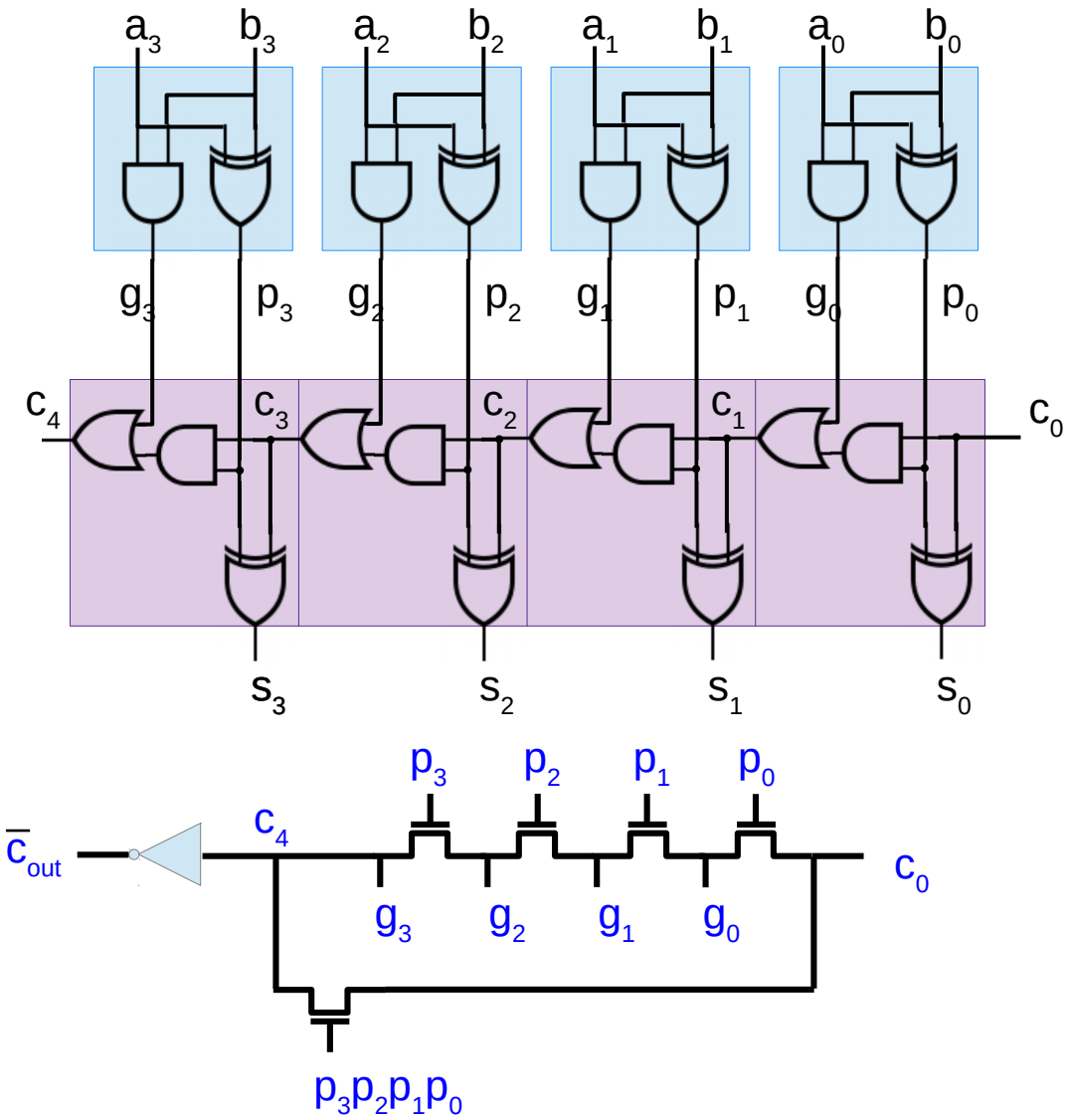
Carry Skip Adder



Www.cs.tufts.edu

Www.cs.tufts.edu/103/notes/Lecture14(Adders-2).pdf

4-bit Carry Skip Adder with P and G



Half Adder

$p_i = a_i \oplus b_i$

$g_i = a_i \wedge b_i$

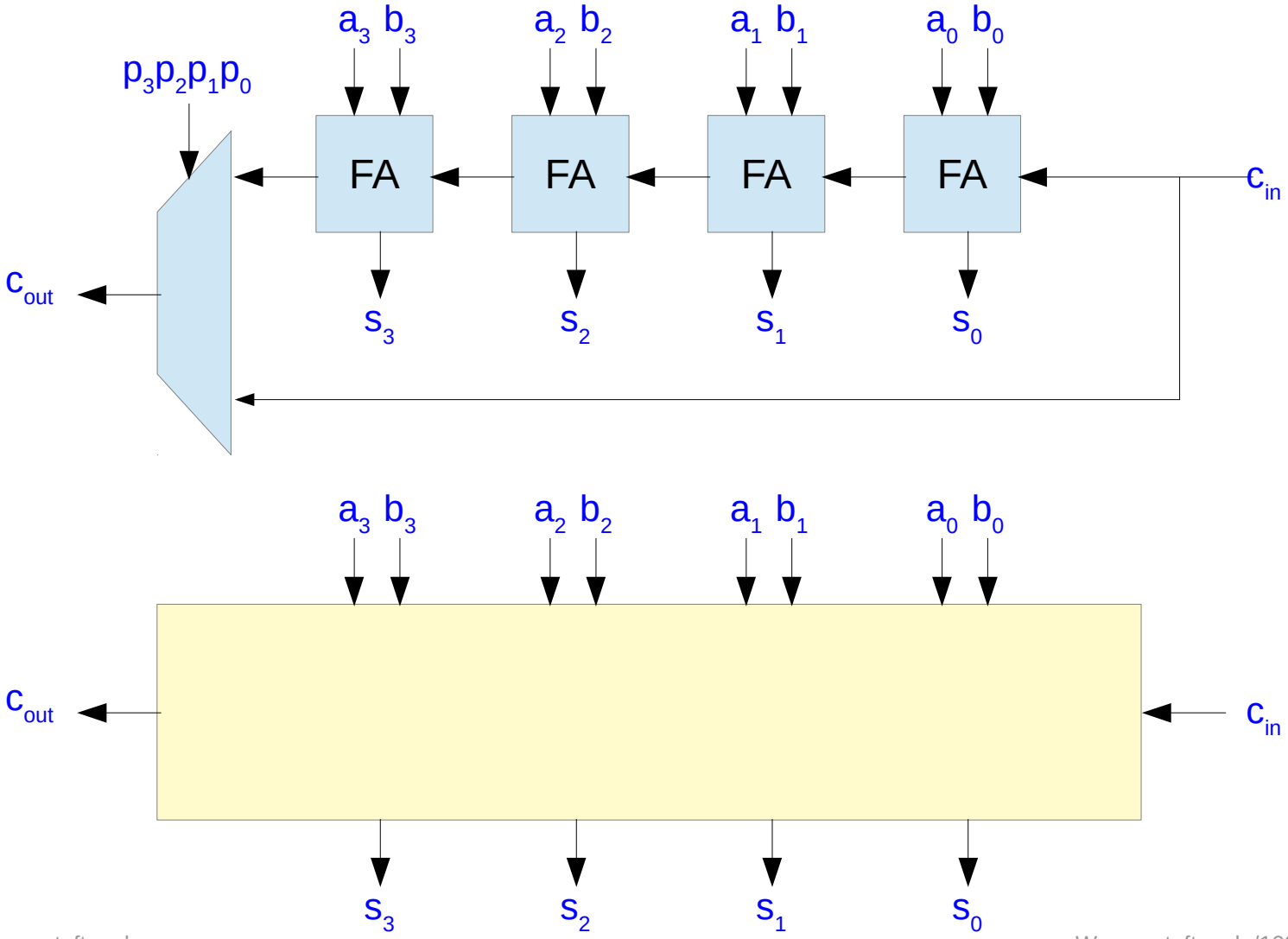
Half Adder

$c_{i+1} = g_i + p_i \wedge c_i$

$s_i = p_i \oplus c_i$

https://upload.wikimedia.org/wikiversity/en/1/18/RCA.Note.H.1.20151215.pdf

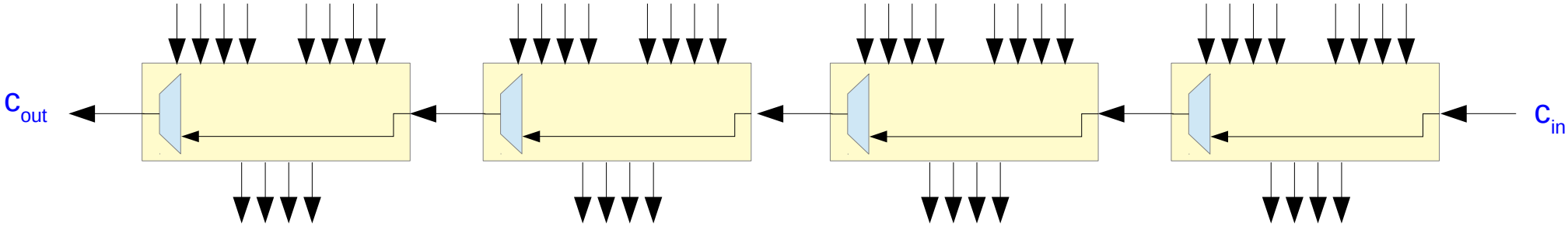
Carry Skip Adder



Www.cs.tufts.edu

Www.cs.tufts.edu/103/notes/Lecture14(Adders-2).pdf

Carry Skip Adder



Www.cs.tufts.edu

Www.cs.tufts.edu/103/notes/Lecture14(Adders-2).pdf

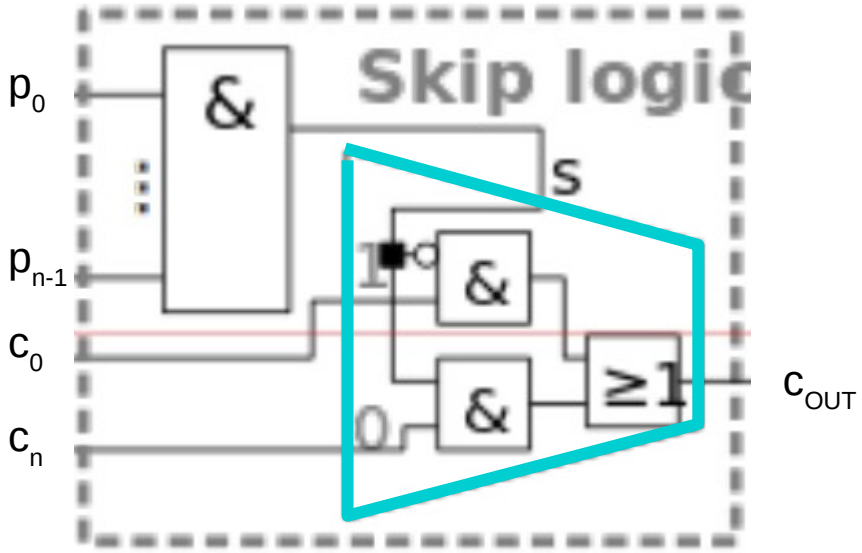
Block Carry Skip Adder

Block-carry-skip adders are composed of a number of carry-skip adders. There are two types of block-carry-skip adders The two operands $A = (a_{n-1}, a_{n-2}, \ldots, a_1, a_0)$ and $B = (b_{n-1}, b_{n-2}, \ldots, b_1, b_0)$ are split in k blocks of $(m_k, m_{k-1}, \ldots, m_2, m_1)$ bits.

- Why are block-carry-skip-adders used?
- Should the block-size be constant or variable?
- Fixed block width vs. variable block width

https://en.wikipedia.org/wiki/Carry-skip_adder

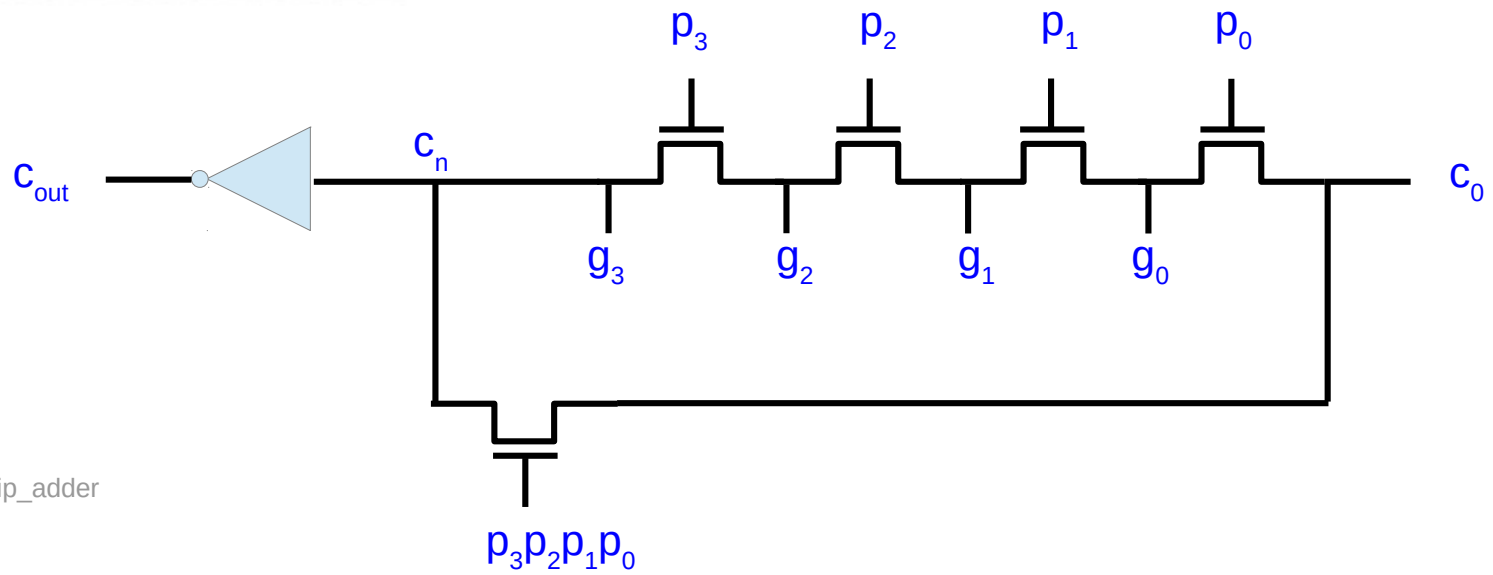
Block Carry Skip Adder



$p_i = a_i \oplus b_i$

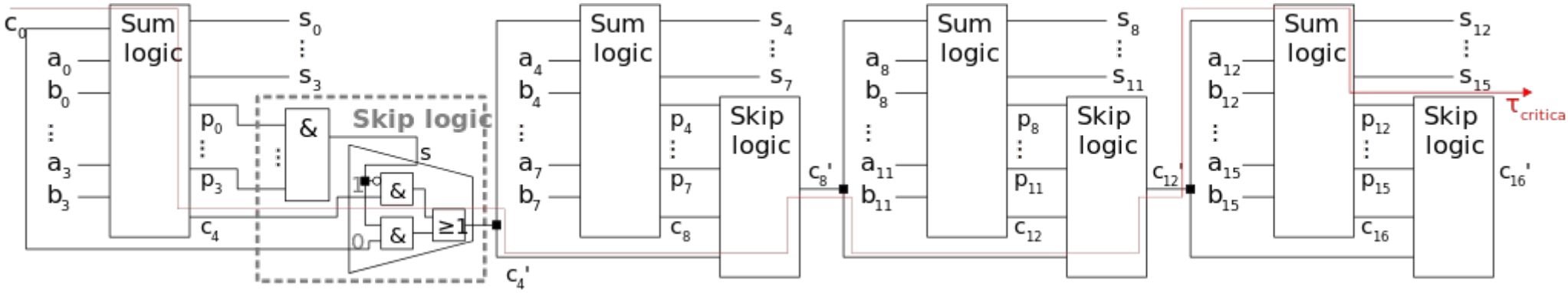
$c_{out} = c_o$ if $p_0 \wedge p_1 \wedge p_2 \wedge p_3$

$c_{out} = c_n$ otherwise



https://en.wikipedia.org/wiki/Carry-skip_adder

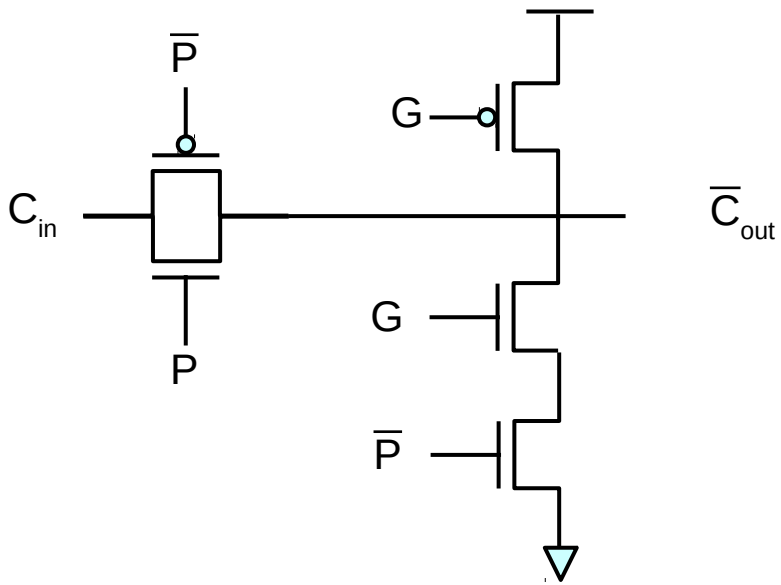
Block Carry Skip Adder

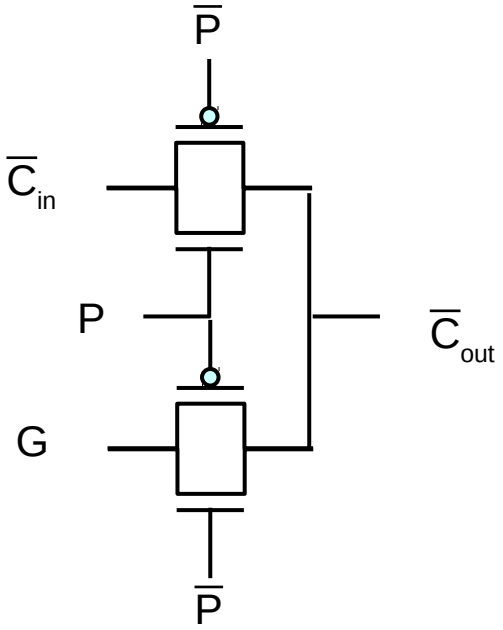


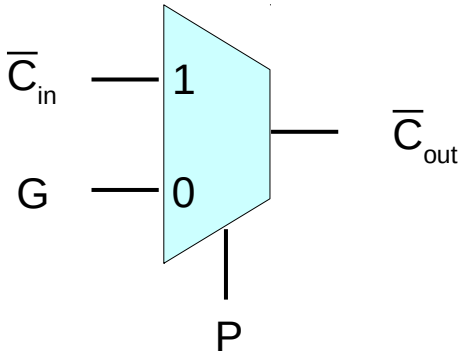
https://en.wikipedia.org/wiki/Carry-skip_adder

Static Carry Circuit – using pass transistors

$$G = a \cdot b$$

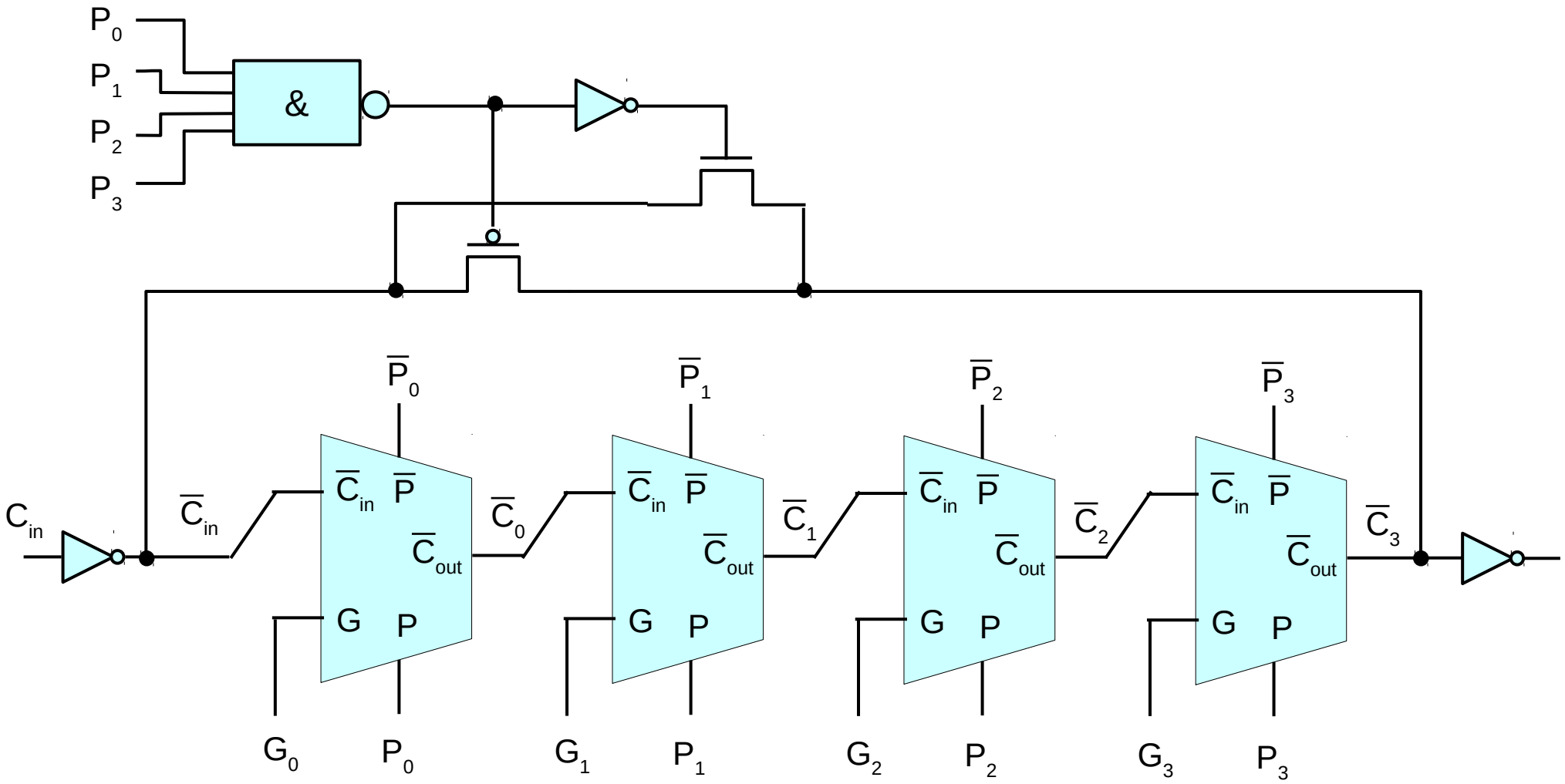






Digital Electronics and Design with VHDL,V, A< Pedroni

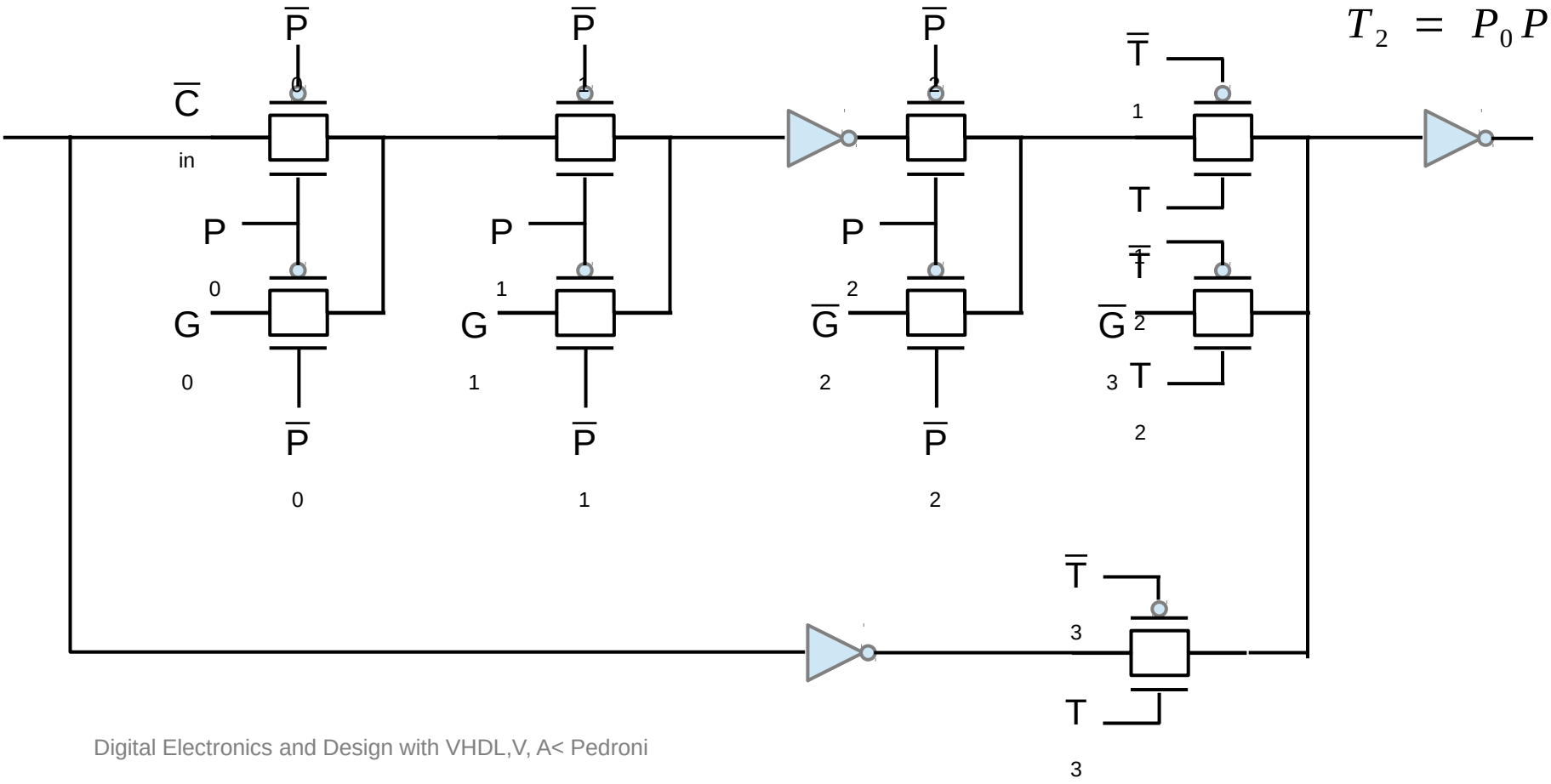
Static Carry Circuit – using pass transistors



Digital Electronics and Design with VHDL,V, A< Pedroni

Static Carry Circuit – using pass transistors

$$T_1 = \overline{P_0 P_1 P_2 P_3}$$
$$T_2 = \overline{P_3}$$
$$T_2 = P_0 P_1 P_2 P_3$$



Digital Electronics and Design with VHDL,V, A< Pedroni

References

[1] en.wikipedia.org
[2] Parhami, “Computer Arithmetic Algorithms and Hardware Designs”