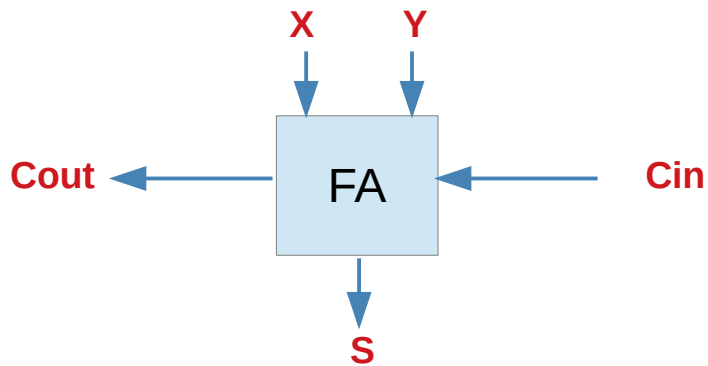# Carry Skip Adder (5A)

-
-

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 Kill, Propagate, Generate conditions (1)

| X	Y		
0	0	K	Kill (=$\overline{P}\overline{G}$)
0	1	P	Propagate
1	0	P	Propagate
1	1	G	Generate

https::/electronics.stackexchange.com/questions/21251/critical-path-for-carry-skip-adder

Carry Kill, Propagate, Generate conditions (2)

killed carry

	0	0	
0		**K** ✕ **+**	**1**
		1	

killed carry

	0	0	
0		**K** ✕ **+**	**0**
		0	

propagated carry

	0 1	1 0	
1		**P** **+**	**1**
		0	

propagated carry

	0 1	1 0	
0		**P** **+**	**0**
		1	

generated carry

	1	1	
1		**G** **+**	**1**
		1	

generated carry

	1	1	
1		**G** **+**	**0**
		0	
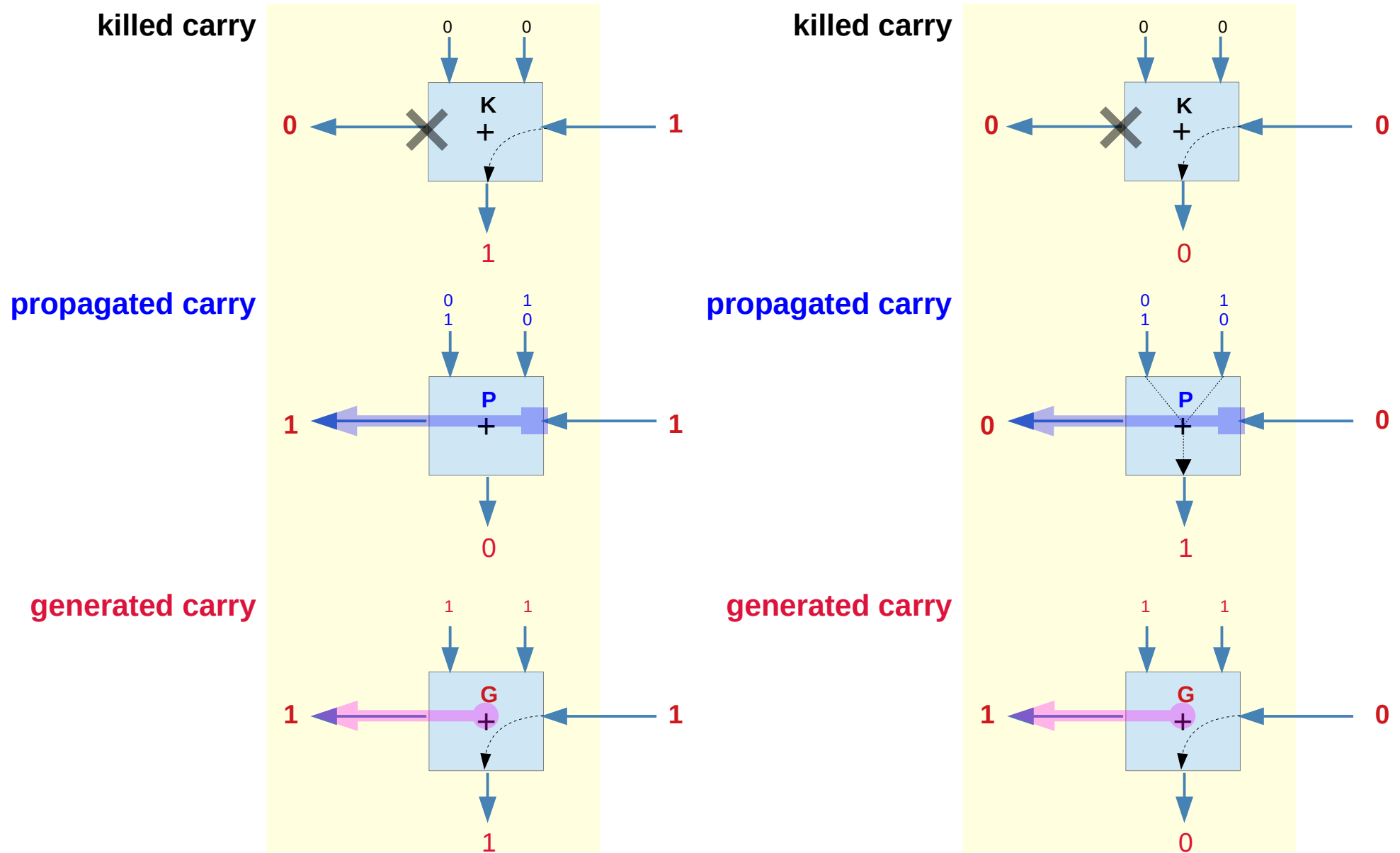
K, **P**, and **G** conditions in a 2-bit adder (1)

X	Y		
0	0	K	Kill (=$\overline{P}\overline{G}$)
0	1	P	Propagate
1	0	P	Propagate
1	1	G	Generate
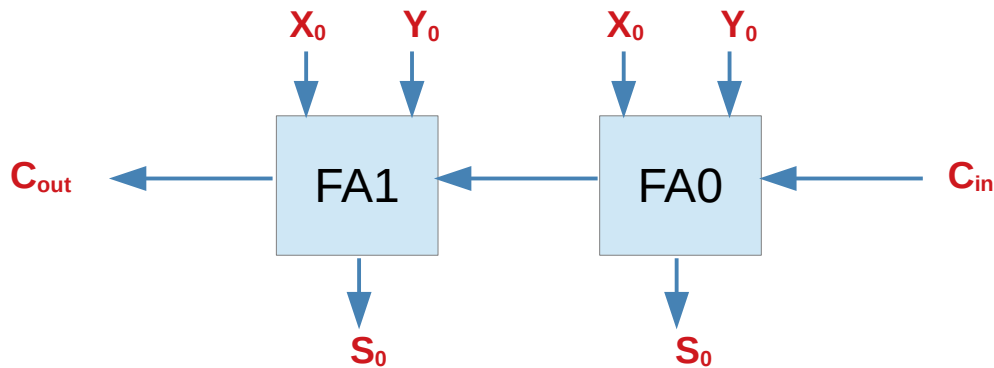
Unless the two FA's are in propagate mode,
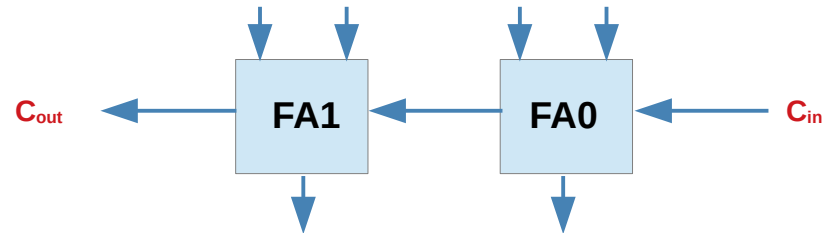the transition of Cin does not affect
the transition of Cout

Critical path – all FA's in propagate mode

Broken paths for any FA in other mode
- kill mode, generate mode

https::/electronics.stackexchange.com/questions/21251/critical-path-for-carry-skip-adder

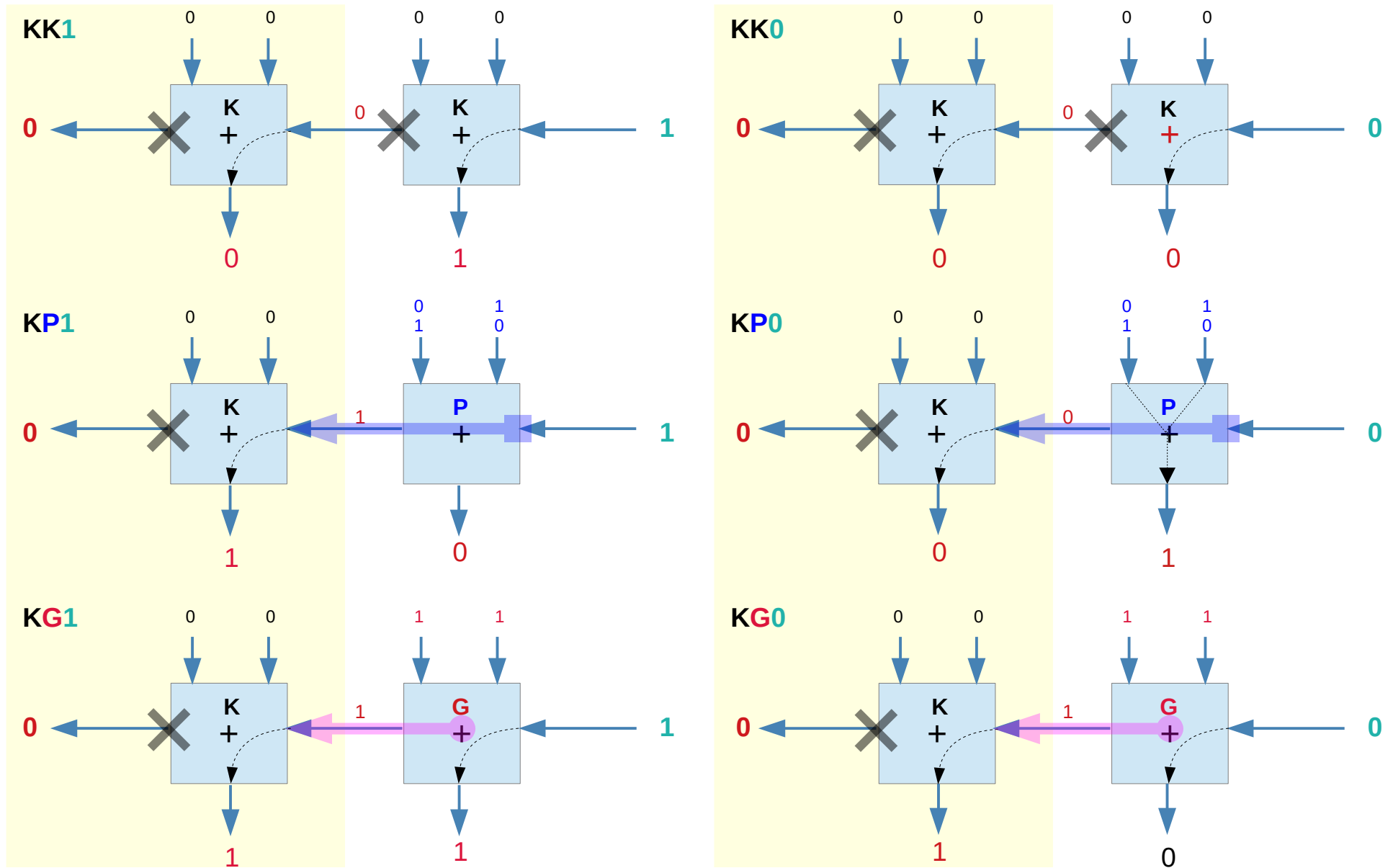# K, P, and G conditions in a 2-bit adder (2)

X	Y		
0	0	K	Kill (=$\overline{P}\,\overline{G}$)
0	1	P	Propagate
1	0	P	Propagate
1	1	G	Generate

C_{out} ← FA1 ← FA0 ← C_{in}

K	K	0
K	K	1
K	P	0
K	P	1
K	G	0
K	G	1
P	K	0
P	K	1
P	P	0
P	P	1
P	G	0
P	G	1
G	K	0
G	K	1
G	P	0
G	P	1
G	G	0
G	G	1

2. Cases when **FA1** is in the **P** mode

PK1

PP1

PG1

PK0

PP0

PG0

Cases for $C_{out} = 1$

Carry Skip Adder (5A)

Young W. Lim
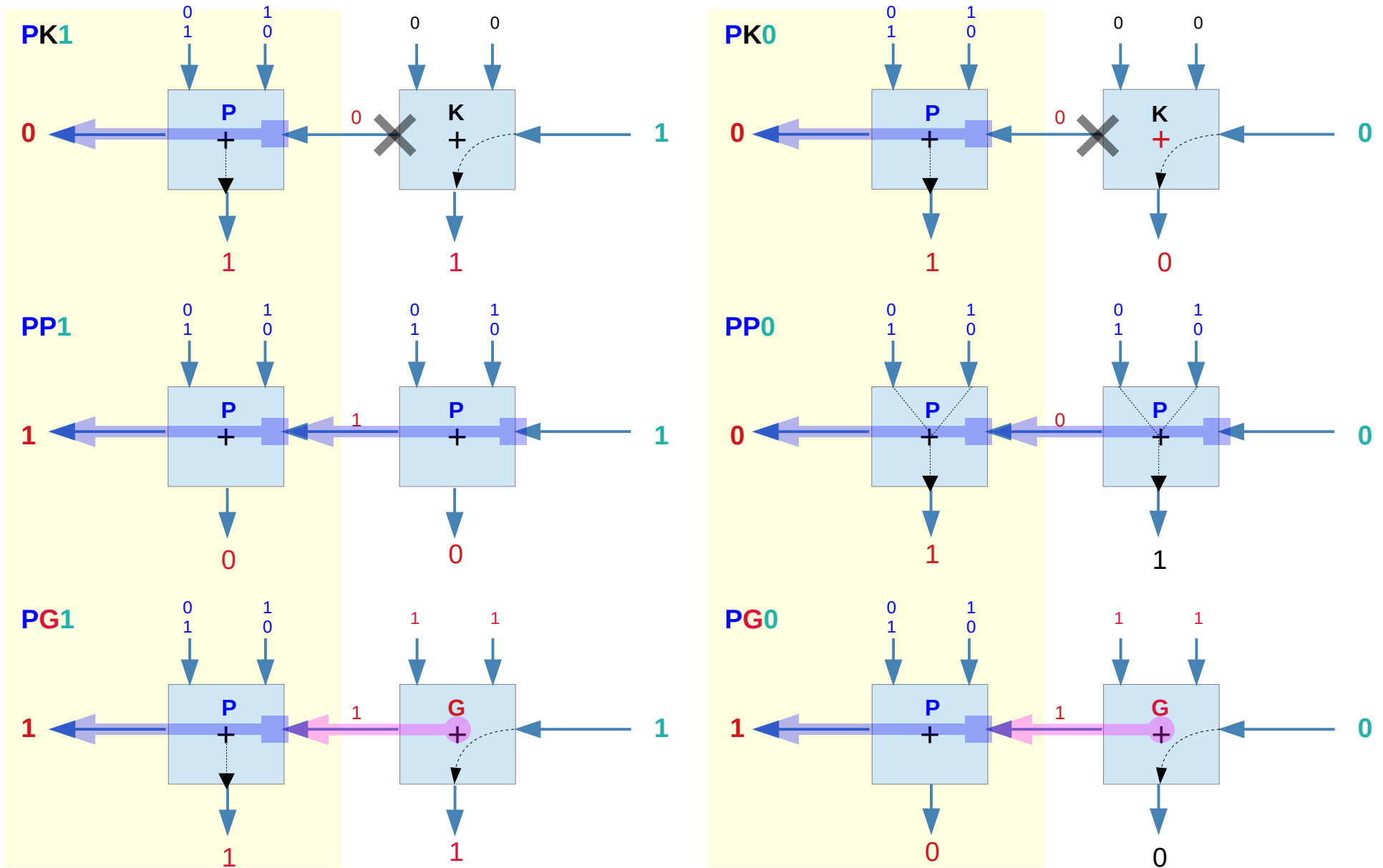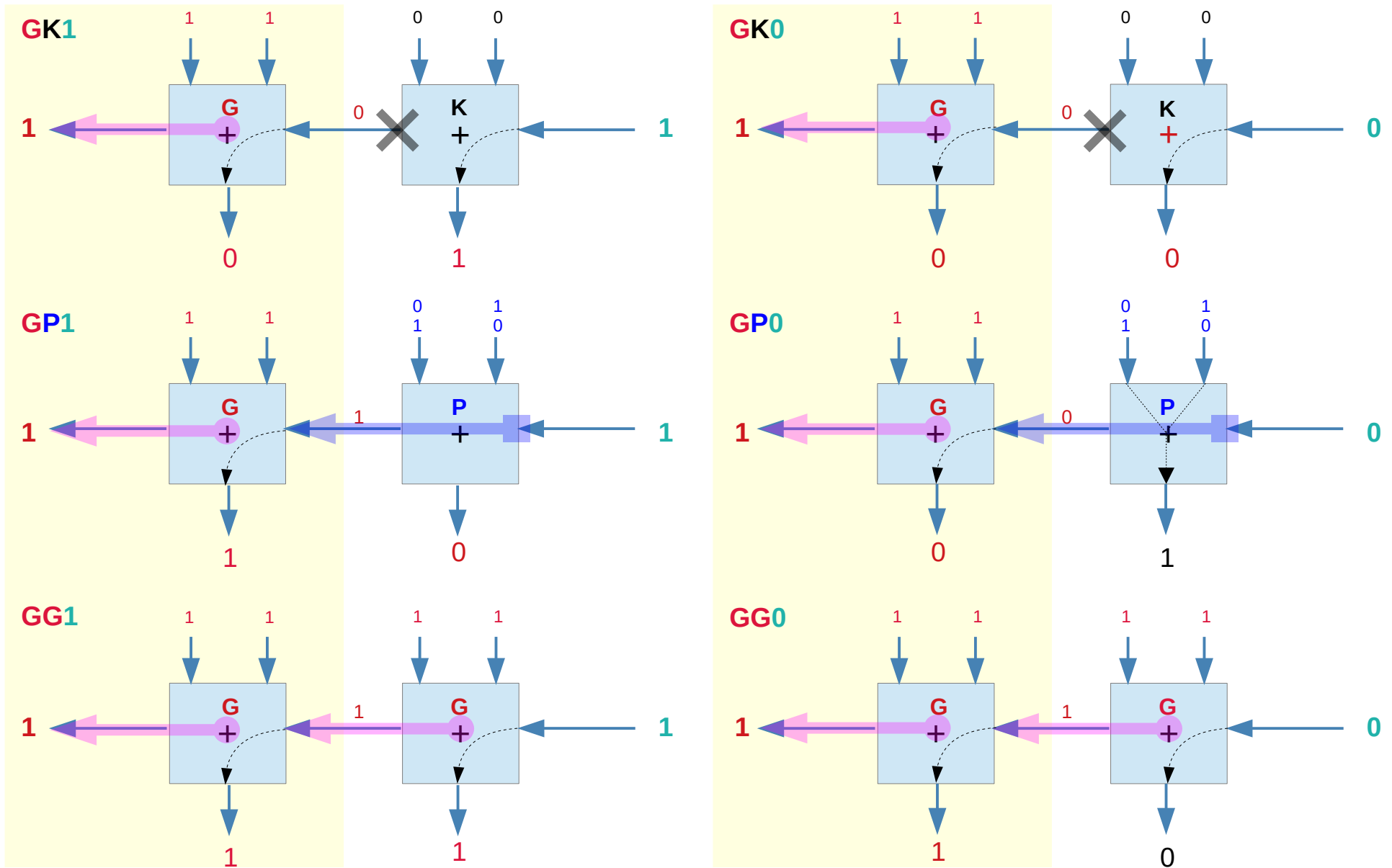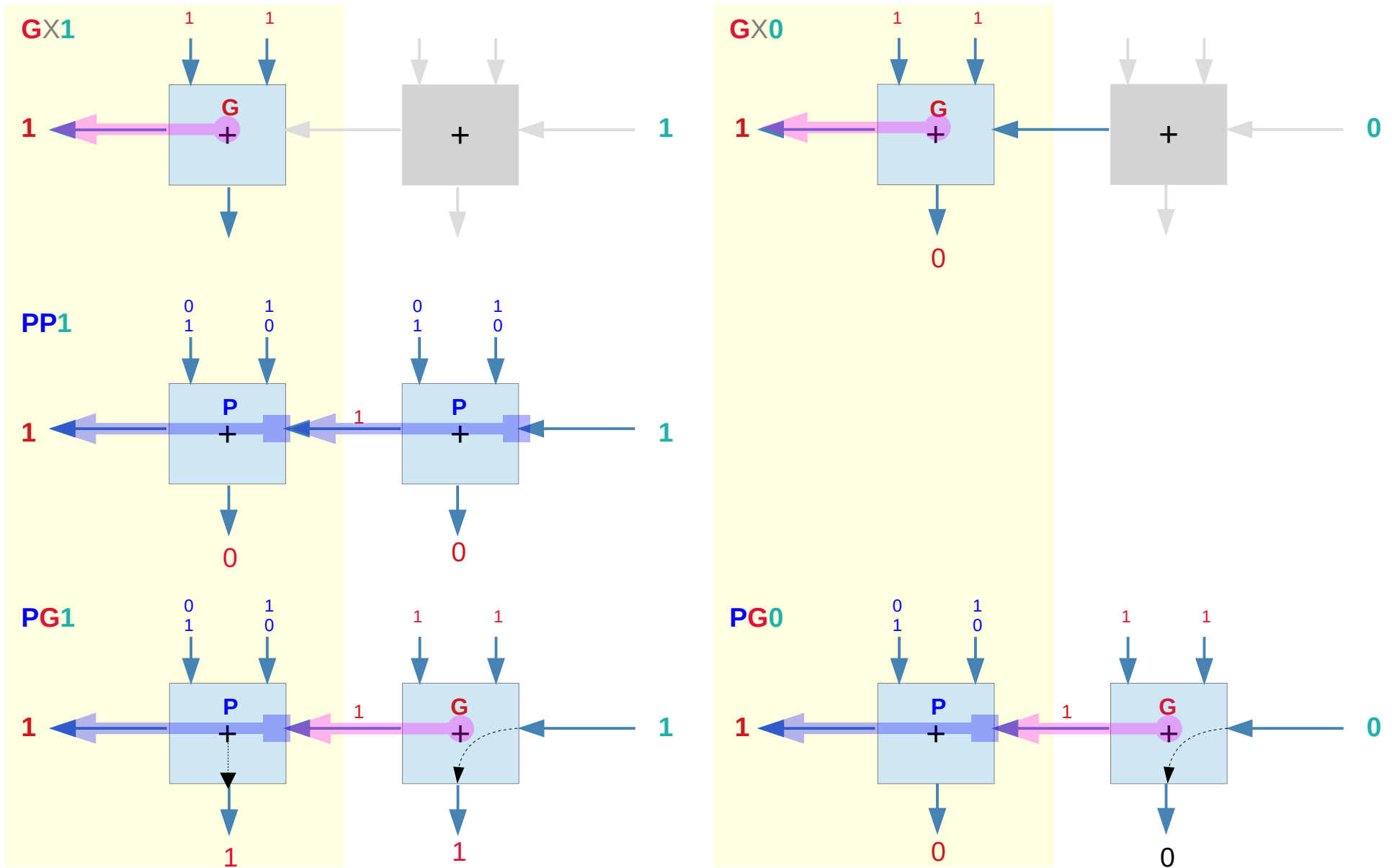
Cases for $C_{out} = 0$
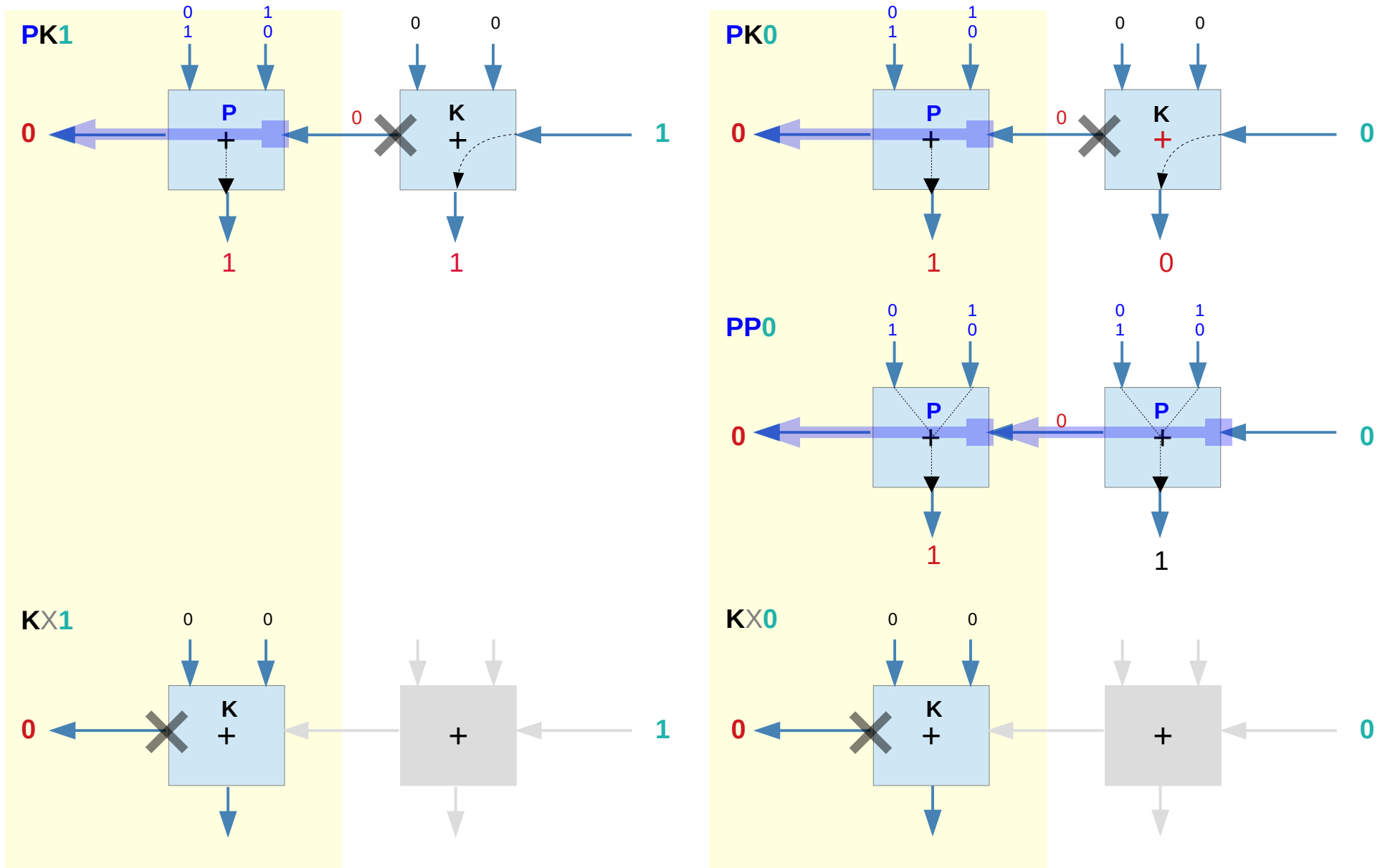
FA with P & G

Half Adder

$s = p \oplus c$

$p \wedge c$

$c_{out} = g + p \wedge c$

$p = a \oplus b$

$g = a \wedge b$
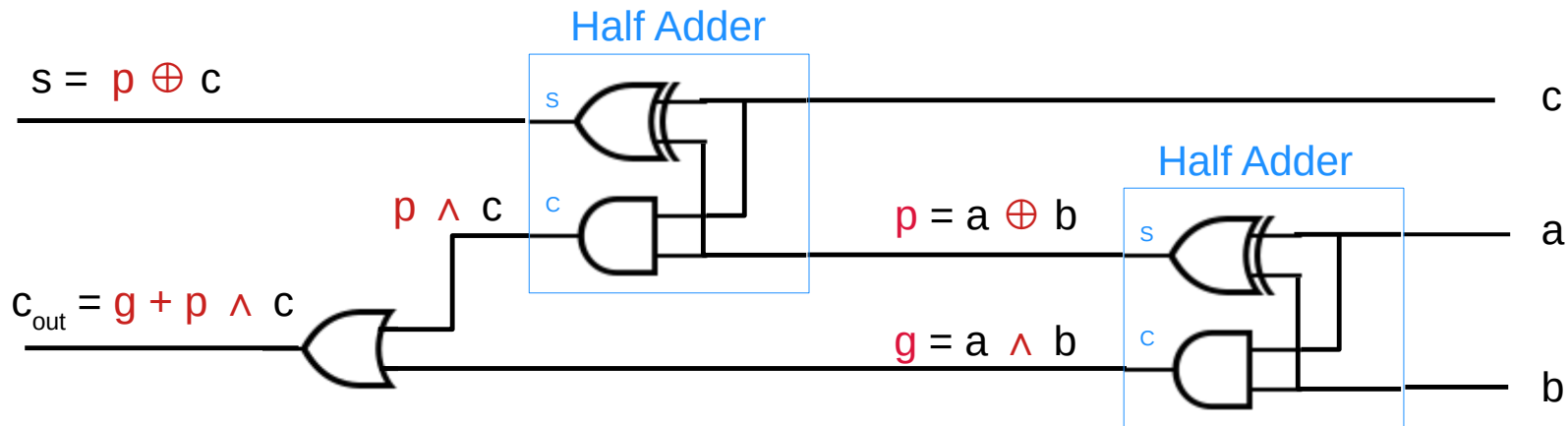
Half Adder

Half Adder
$S = a \oplus b$
$C = a \wedge b$

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

p $S = a \oplus b$

g $C = a \wedge b$

Half Adder
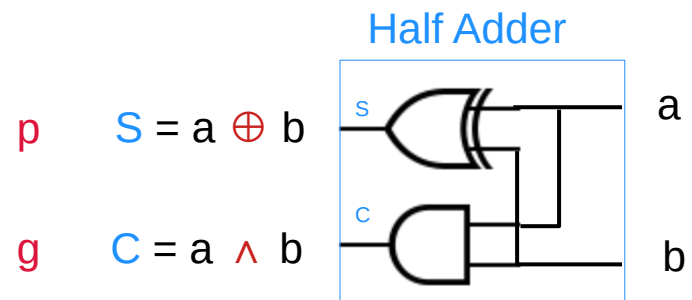
Full adder with additional generate and propagate signals.

Ripple Carry Adder
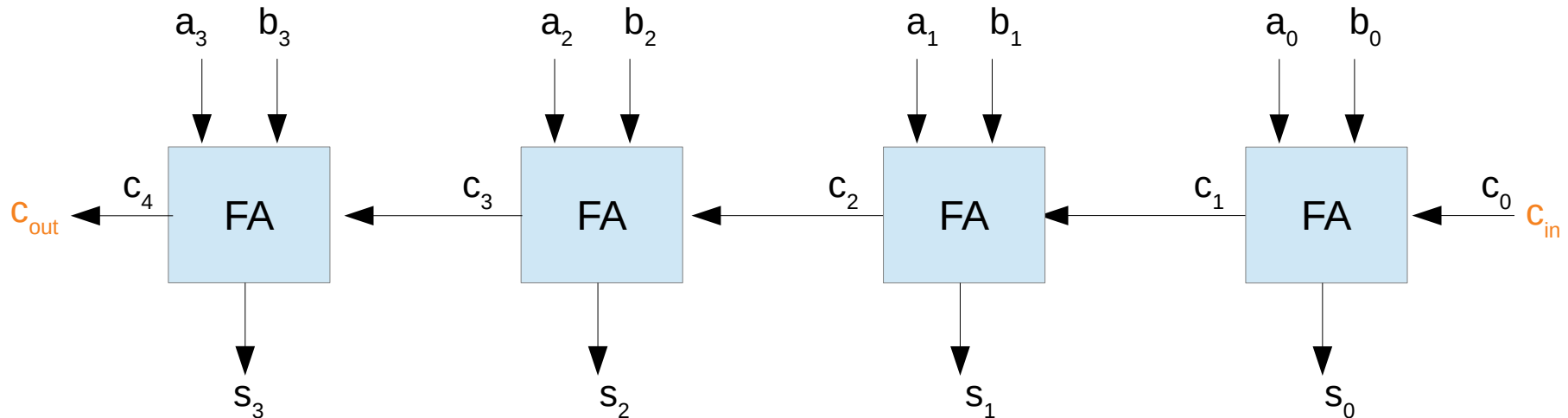
$p_i = a_i \oplus b_i$

$g_i = a_i \wedge b_i$

$c_1 = g_0 + p_0 \wedge c_0$

$c_2 = g_1 + p_1 \wedge c_1$

$c_3 = g_2 + p_2 \wedge c_2$
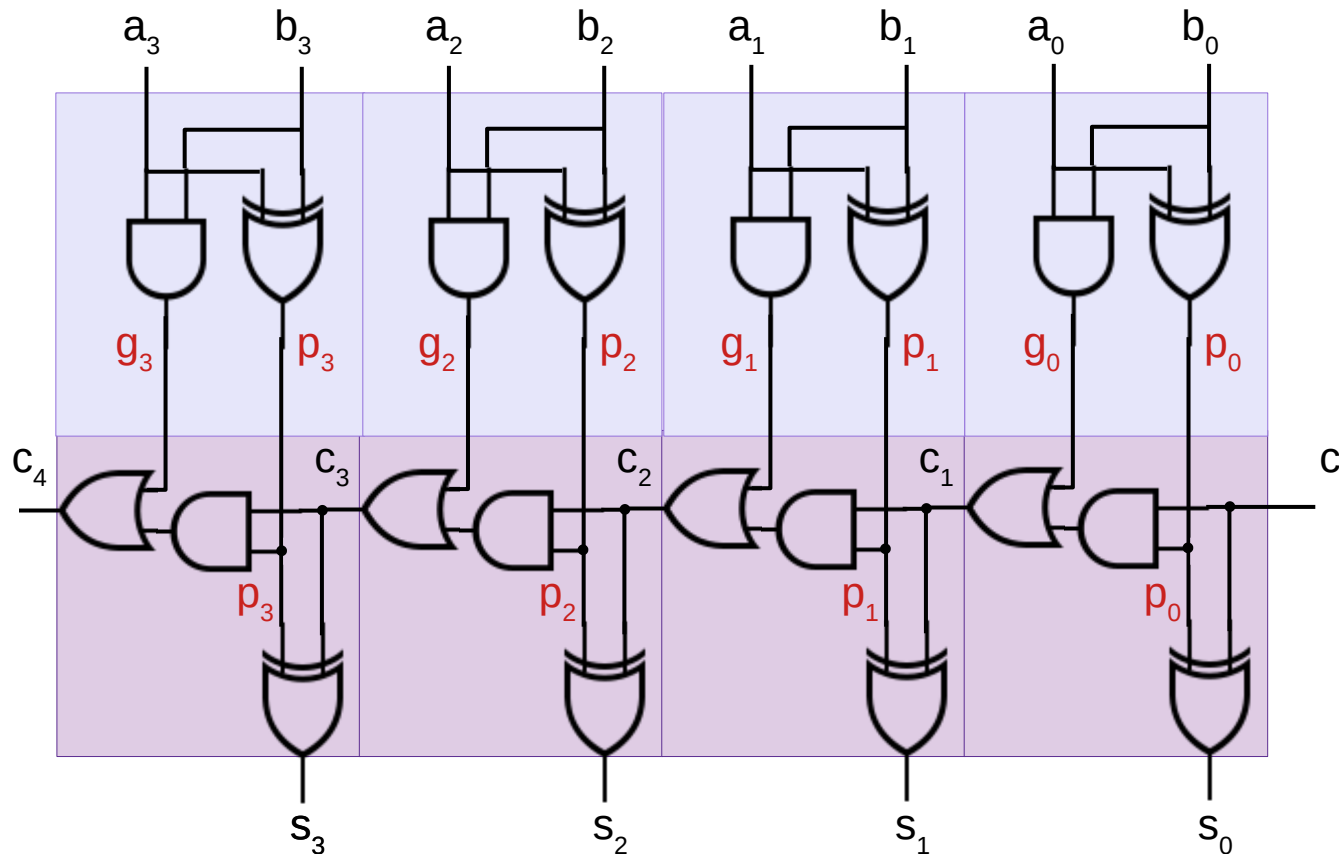
$c_4 = g_3 + p_3 \wedge c_3$

generated carry ←   → propagated carry

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

4-bit Full Adder with P and G

Half Adder

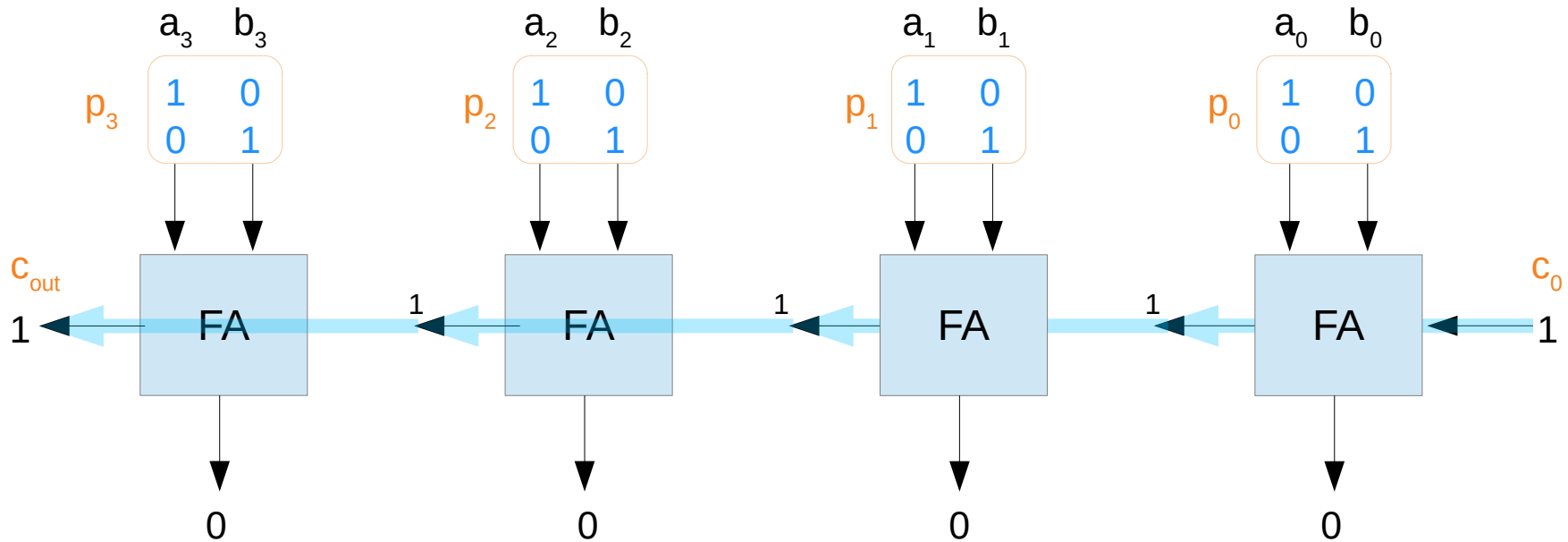$p_i = a_i \oplus b_i$

$g_i = a_i \wedge b_i$

Half Adder

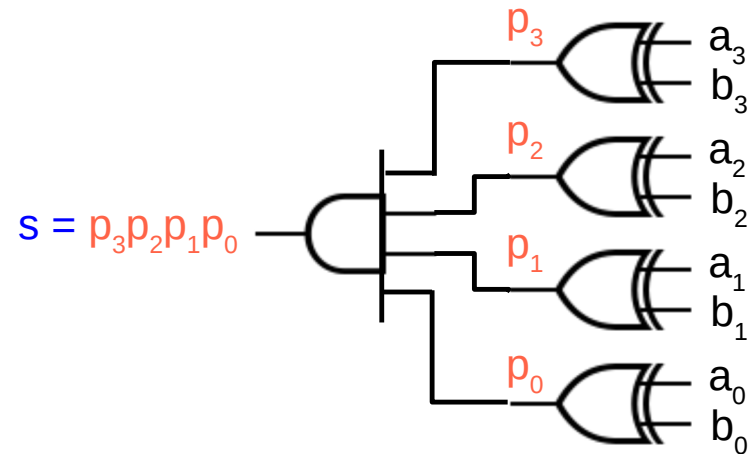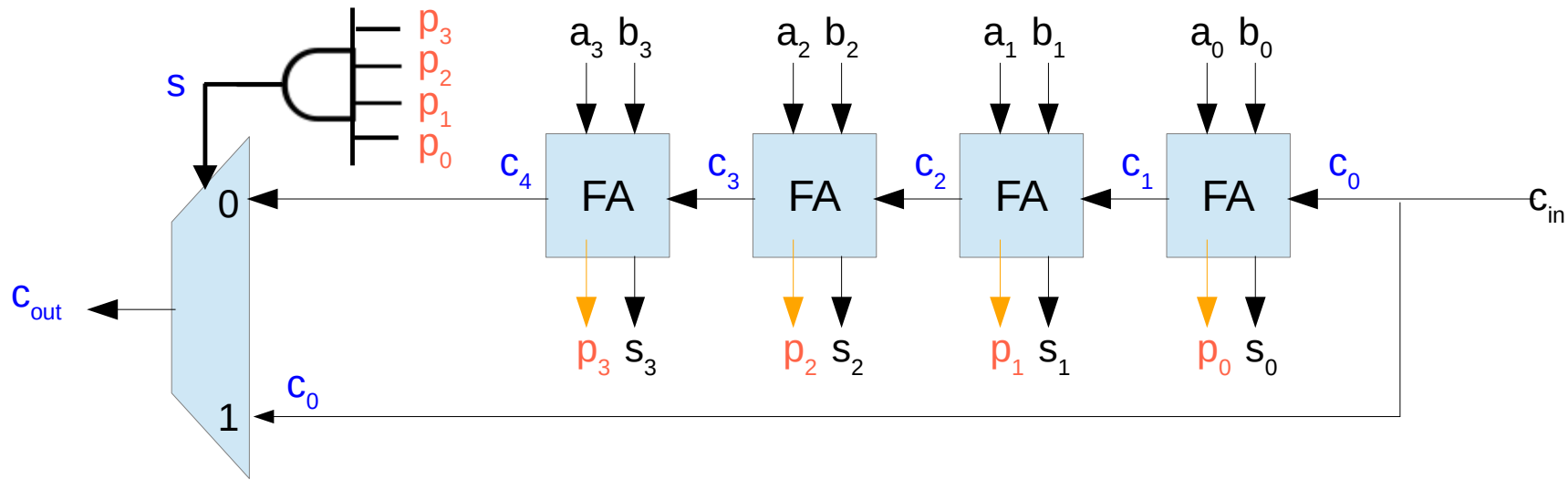$c_{i+1} = g_i + p_i \wedge c_i$

$s_i = p_i \oplus c_i$

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

C$_0$ propagation condition

a$_3$ b$_3$ a$_2$ b$_2$ a$_1$ b$_1$ a$_0$ b$_0$

p$_3$ $\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$ p$_2$ $\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$ p$_1$ $\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$ p$_0$ $\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$

c$_{out}$

1 ← FA ← 1 ← FA ← 1 ← FA ← 1 ← FA ← 1   c$_0$

0   0   0   0

c$_0$ can be propagated to c$_{out}$ only when s = 1

$$s \quad = p_3 \wedge p_2 \wedge p_1 \wedge p_0 = p_{[3\,:\,0]}$$
$$= (a_3 \oplus b_3)$$
$$\wedge \ (a_2 \oplus b_2)$$
$$\wedge \ (a_1 \oplus b_1)$$
$$\wedge \ (a_0 \oplus b_0)$$

p$_3$ ⊕ a$_3$ b$_3$

p$_2$ ⊕ a$_2$ b$_2$

s = p$_3$p$_2$p$_1$p$_0$

p$_1$ ⊕ a$_1$ b$_1$

p$_0$ ⊕ a$_0$ b$_0$

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
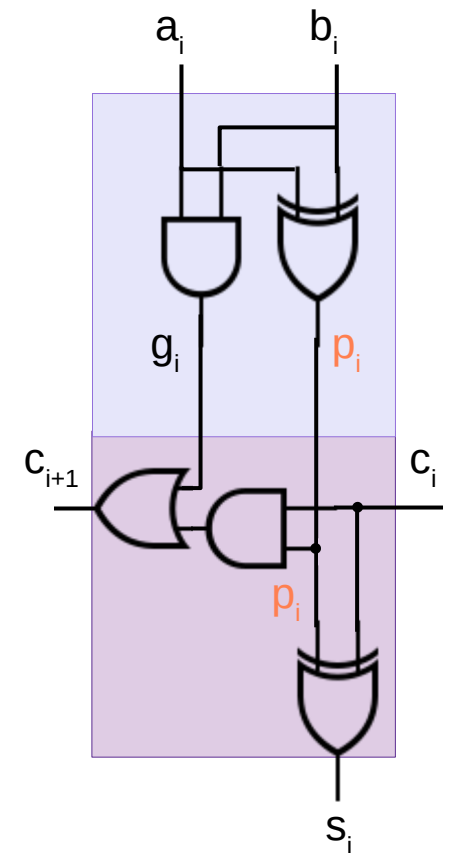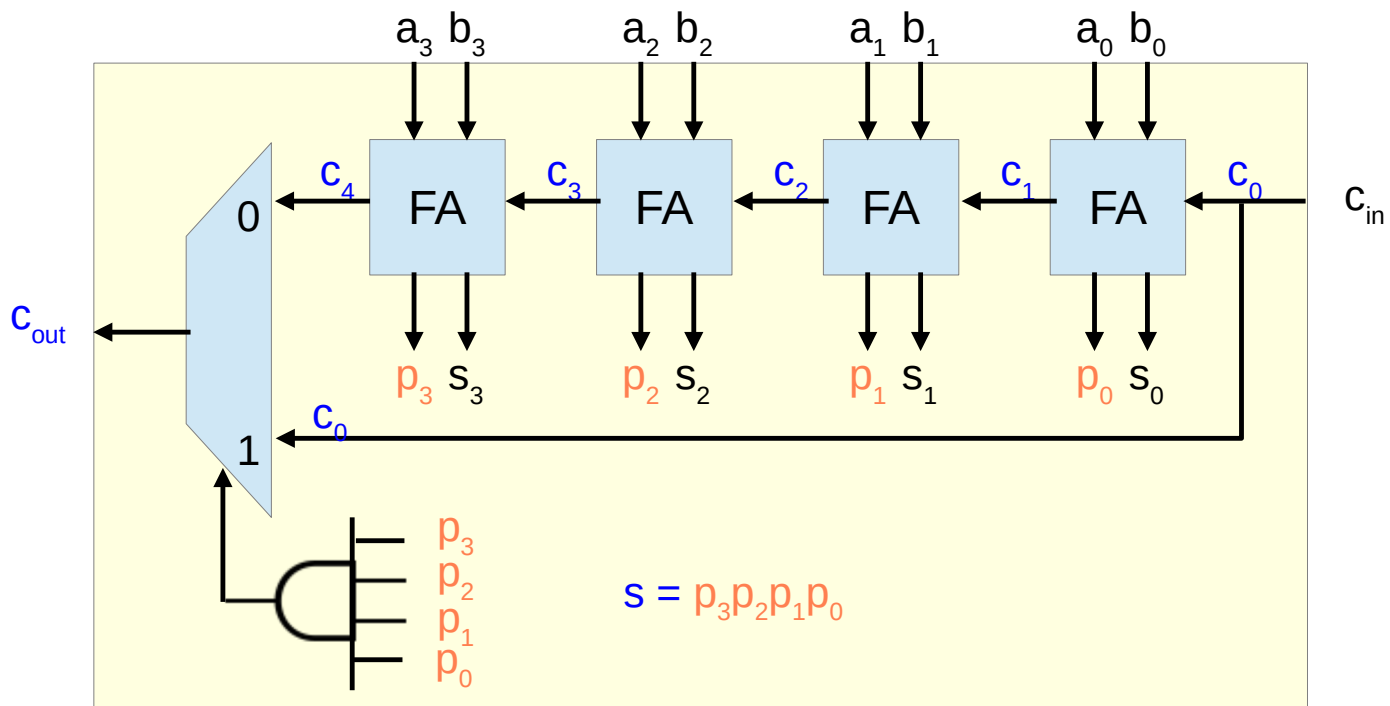to the carry-out signal c_{out}

$s = p_3 \wedge p_2 \wedge p_1 \wedge p_0 = p_{[3:0]}$

when $s = 1$, $c_{out} \leftarrow c_0$

otherwise, internally generated carries
can be propagated to $c_{out} \leftarrow c_4$

Carry Skip Adder

a_3 b_3 a_2 b_2 a_1 b_1 a_0 b_0

c_4 c_3 c_2 c_1 c_0 c_{in}

FA   FA   FA   FA

c_{out}

p_3 s_3 p_2 s_2 p_1 s_1 p_0 s_0

c_0

p_3
p_2
p_1
p_0

$s = p_3 p_2 p_1 p_0$

a_i b_i

g_i p_i

c_{i+1} c_i

p_i

s_i

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

Carry Skip Adder

The critical path of a Carry Skip Adder begins at the first full adder,
passes through all adders and ends at the sum bit s_{n-1}

Since a single *n-bit* Carry Skip Adder has no real speed benefit
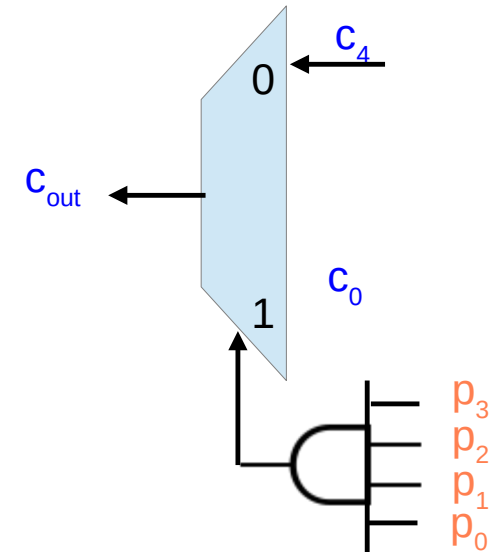compared to a *n-bit* Ripple Carry Adder

$$T_{CSA}(n) = T_{RCA}(n)$$

Carry Skip Adders are chained to reduce the overall critical path,
(Block Carry Skip Adders)

The skip logic consists of a m-input AND gate and one MUX
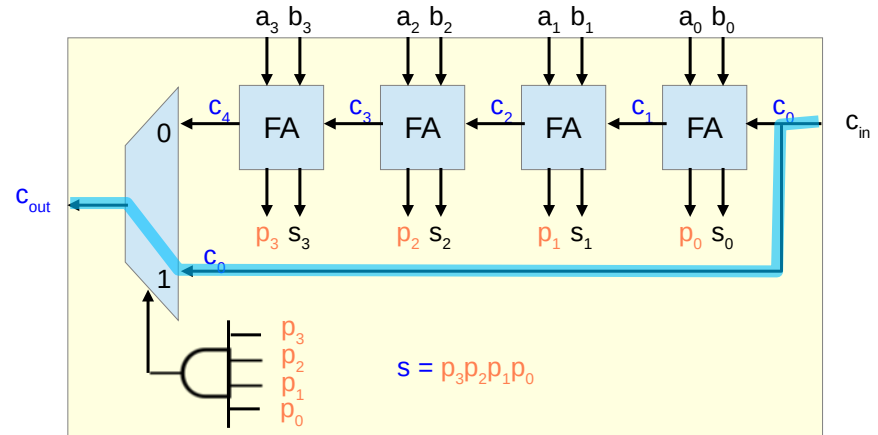
$$T_{SK} = T_{AND}(m) + T_{MUX}$$

Carry Skip Adder

As the propagate signals
are computed in parallel and
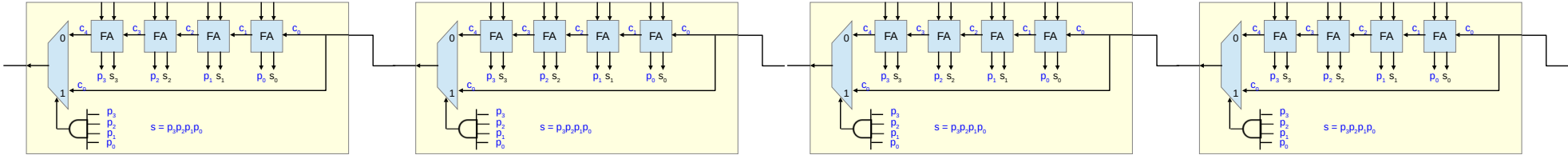are early available,

$$p_i = a_i \oplus b_i$$

the critical path for the skip logic in a Carry Skip Adder
consists of the delay imposed by the multiplexer (conditional skip)

$$T_{CSK} = T_{MUX} = 2D$$

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

Carry Skip Adder

Carry Skip Adder

Fixed size block carry skip adders split the n bits of the input bits
Into blocks of m bits each, resulting in k = n / m blocks.

The critical path consists of the ripple path and the skip element of the first block,
The skip apths that are enclosed between the first and the last block,
And finally the ripple path of the last block

$T_{FCSA}(n) = T_{CRA[0:cout]}(m) + T_{CSK} + (k-2)T_{CSK} + T_{CRA}(m)$
$= 3D + m\, 2D + (k-1)2D + (m+2)2D = (2m+k)2D + 5d$

The optimal block size for a given adder width n is derived by equating to 0

$dT_{FCSA}(n) / dm = 0$

$2D(2-n(1/m^2)) = 0$

M1,2 = +-sqrt(n/2)   m = sqrt(m/2)

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

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

Oklobdzija: High-Speed VLSI arithmetic units : adders and multipliers

Block Carry Skip Adder

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

Carry Skip Adder

Since the Cin-to-Cout represents the longest path
in the ripple-carry-adder, an obvious attempt is
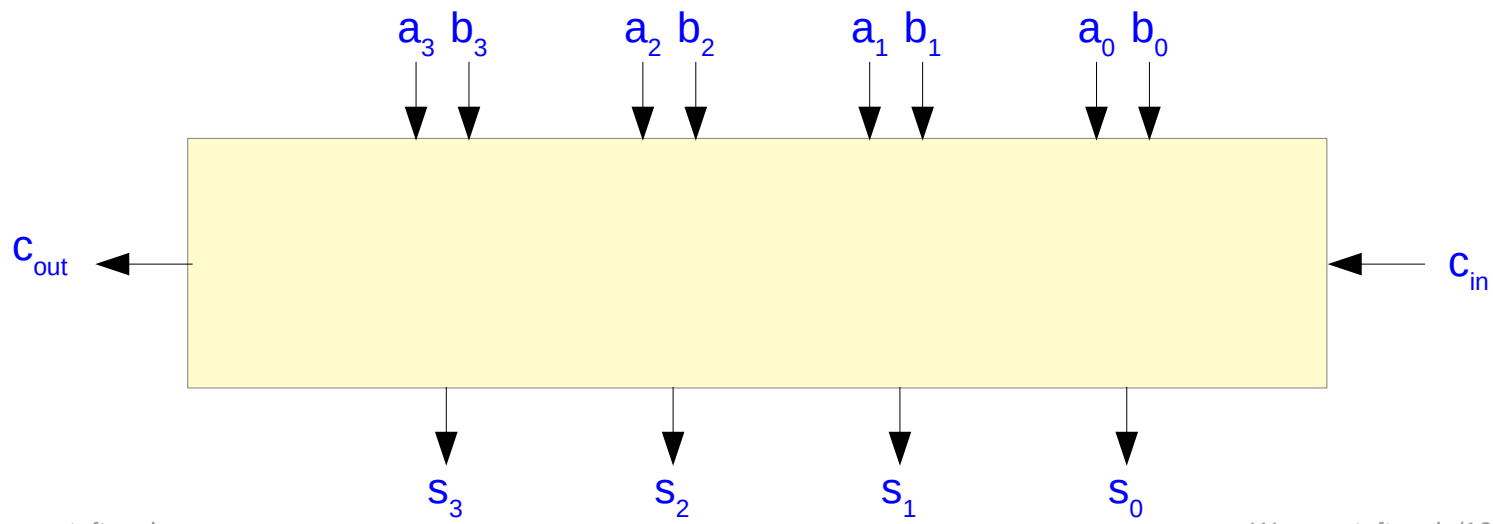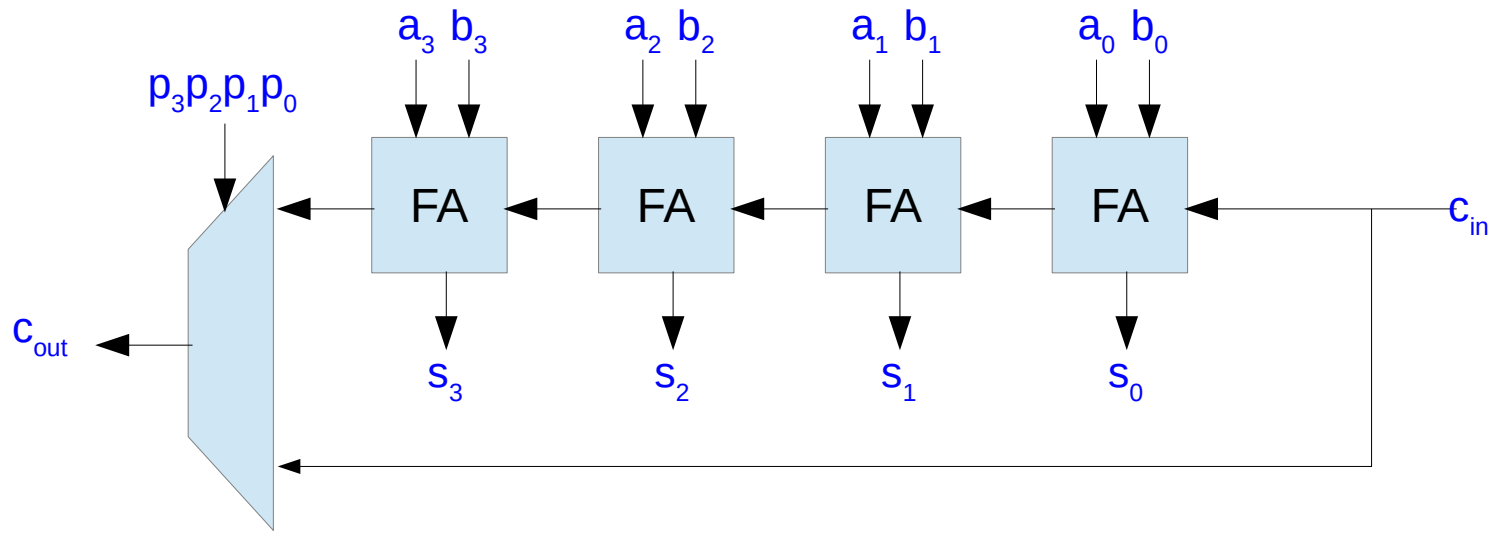to accelerate carry propagation through the adder.

This is accomplished by using Carry-Propagate
p_i signals within a group of bits.

If <u>all</u> the p_i signals within the group are $p_i = 1$,
the condition exist for the carry to bypass the entire group:

$$P = p_i \bullet p_{i+1} \bullet p_{i+2} \bullet \ldots\ldots \bullet p_{i+k-1}$$

Oklobdzija: High-Speed VLSI arithmetic units : adders and multipliers

Carry Skip Adder

a_3 b_3 a_2 b_2 a_1 b_1 a_0 b_0

$p_3p_2p_1p_0$

FA ← FA ← FA ← FA ← c_{in}

c_{out}

s_3 s_2 s_1 s_0

a_3 b_3 a_2 b_2 a_1 b_1 a_0 b_0

c_{out} c_{in}

s_3 s_2 s_1 s_0

Carry Skip Adder

The Carry Skip Adder (CSKA) <u>divides</u> the words
to be added into <u>groups</u> of <u>equal size</u> of **k-bits**.

The basic structure of an **N-bit Carry Skip Adder**

Within the group, carry propagates in a ripple-carry fashion.

In addition, an AND gate is used
to form the group propagate signal P.
$$P = p_i \bullet p_{i+1} \bullet p_{i+2} \bullet \ldots \bullet p_{i+k-1}$$

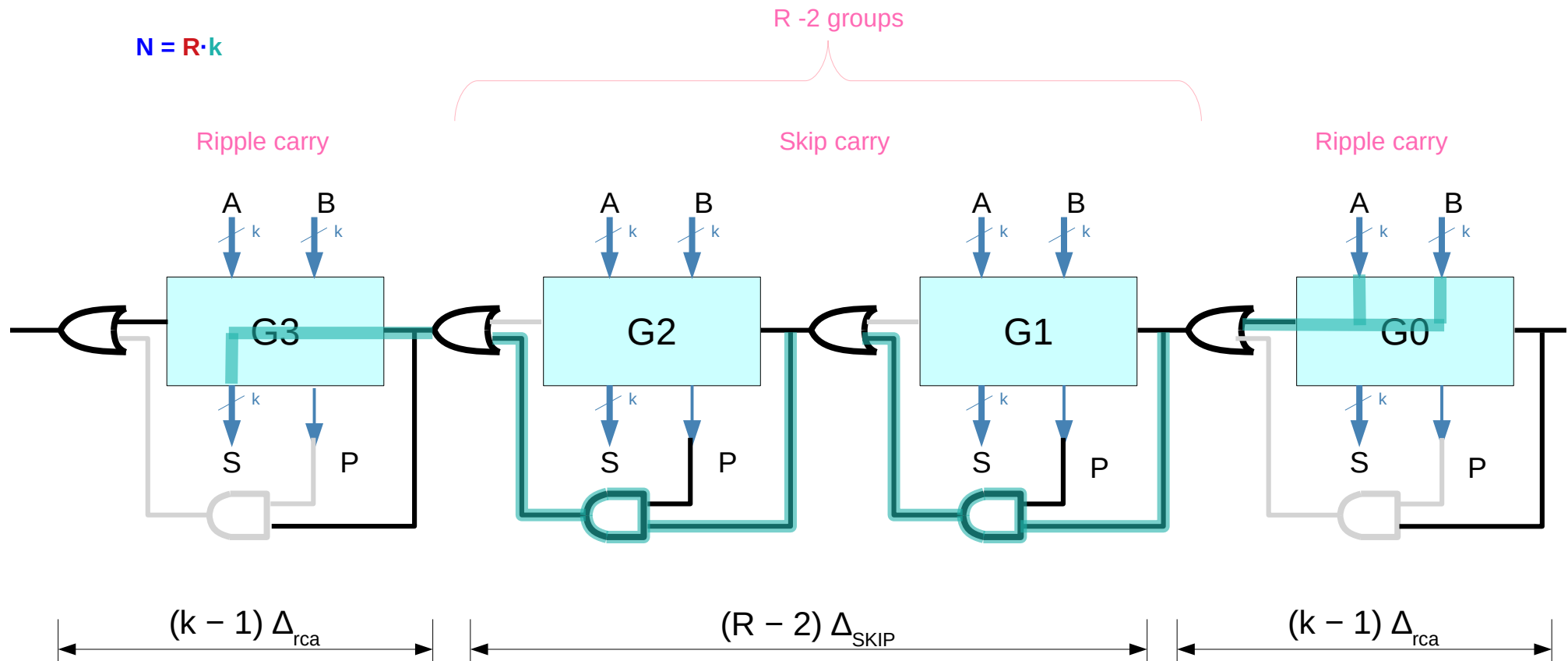If P = 1 the condition exists for carry to bypass (skip) over the group

Oklobdzija: High-Speed VLSI arithmetic units : adders and multipliers

Carry Skip Adder

$$N = R \cdot k$$

k bits   k bits   k bits   k bits

[(R)k-1 … (R-1)k]   [(r+1)k-1 … (r)k]   [(2)k-1 … (1)k]   [(1)k-1 … (0)k]

[N-1 … N-k-1] $r \in [R-1, \cdots, 0]$ [2k-1 … k]   [k-1 … 0]

A   B   A   B   A   B   A   B

k k   k k   k k   k k

G3   G2   G1   G0

k   k   k   k

S   P   S   P   S   P   S   P

Oklobdzija: High Speed VLSI arithmetic units : adders and multipliers

Carry Skip Adder

$N = R \cdot k$

R -2 groups

Ripple carry   Skip carry   Ripple carry

$(k - 1) \, \Delta_{rca}$ $(R - 2) \, \Delta_{SKIP}$ $(k - 1) \, \Delta_{rca}$
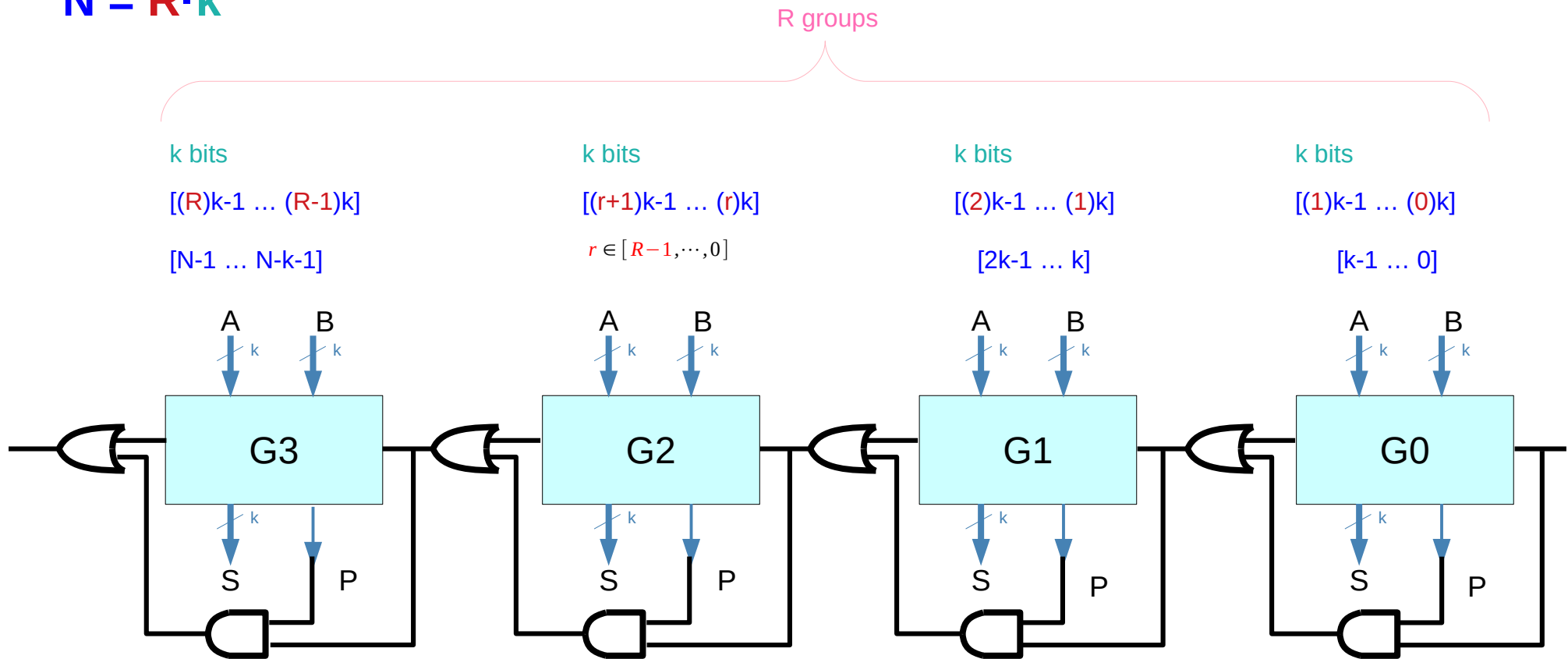
Any kill or generate condition results in divided (broken) critical paths

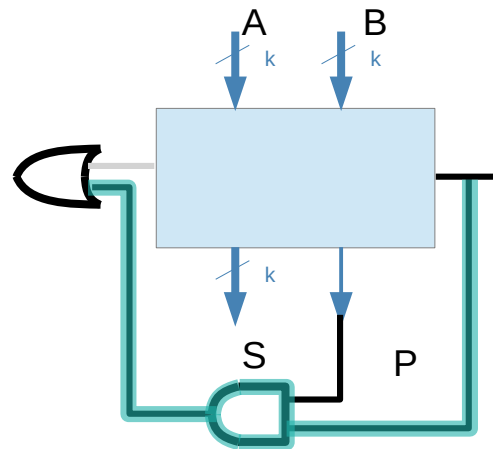All FA's in R-2 groups must have the propagate condition

Oklobdzija: High Speed VLSI arithmetic units : adders and multipliers

Carry Skip Adder

$$(k - 1) \Delta_{rca}$$
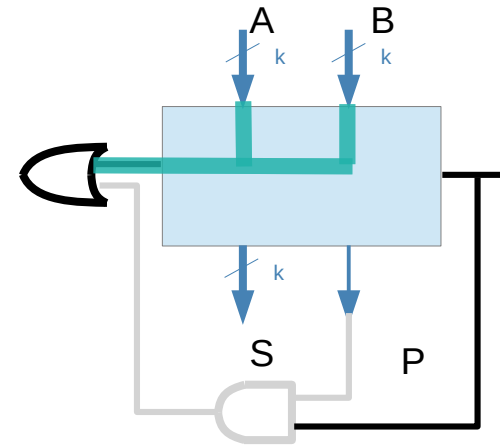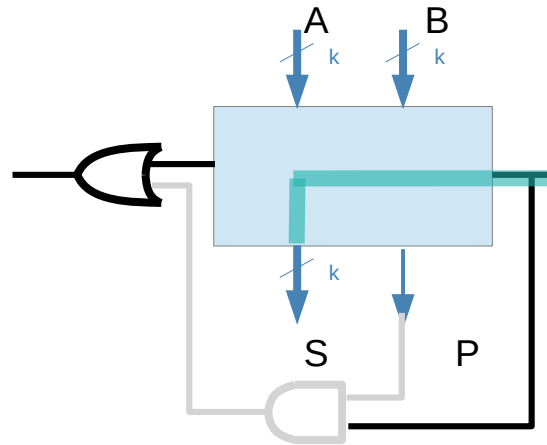
Skip carry

$$\Delta_{SKIP}$$

Oklobdzija: High-Speed VLSI arithmetic units : adders and multipliers

Carry Skip Adder

The <u>maximal delay</u> Δ of a Carry Skip Adder is encountered
<u>when</u> carry is generated in the least-significant bit position,

- rippling through *k-1* bit positions,
- skipping over *R-2 = N/k-2* groups in the middle,
- rippling to the *k-1* bits of most significant group and
- being assimilated in the *N-th* bit position to produce the sum S_N :

$$\Delta_{CSA} = (k - 1)\,\Delta_{rca} + (R - 2)\,\Delta_{SKIP} + (k - 1)\,\Delta_{rca}$$
$$= 2\,(k - 1)\,\Delta_{rca} + (R - 2)\,\Delta_{SKIP}$$
$$= 2\,(k - 1)\,\Delta_{rca} + (N/k - 2)\,\Delta_{SKIP}$$

Oklobdzija: High-Speed VLSI arithmetic units : adders and multipliers

Carry Skip Adder

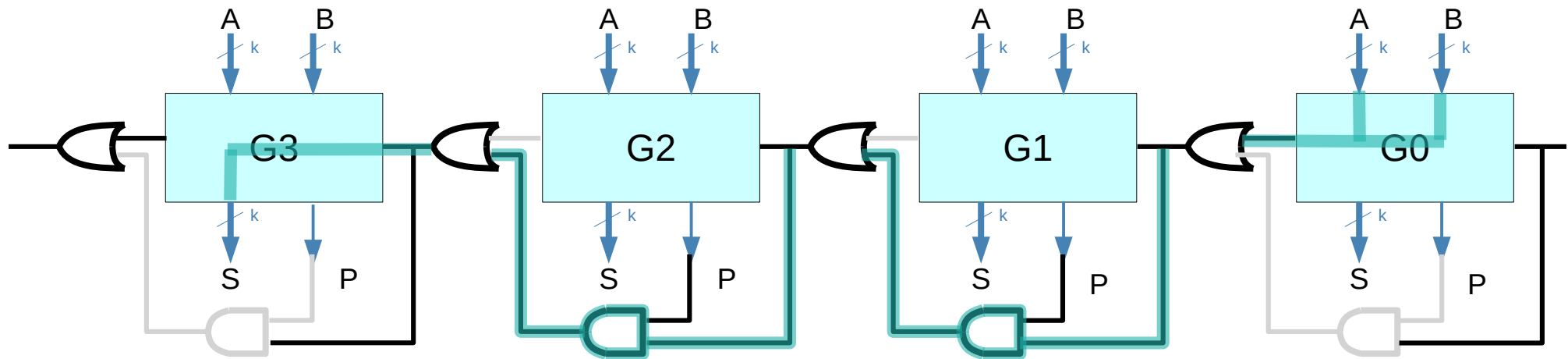$$\Delta_{CSA} = (k-1)\,\Delta_{rca} + (R-2)\,\Delta_{SKIP} + (k-1)\,\Delta_{rca}$$
$$= 2\,(k-1)\,\Delta_{rca} + (R-2)\,\Delta_{SKIP}$$
$$= 2\,(k-1)\,\Delta_{rca} + (N/k - 2)\,\Delta_{SKIP}$$

Carry Skip Adder is faster than RCA at the expense
of a few relatively simple modifications.

N = R·k
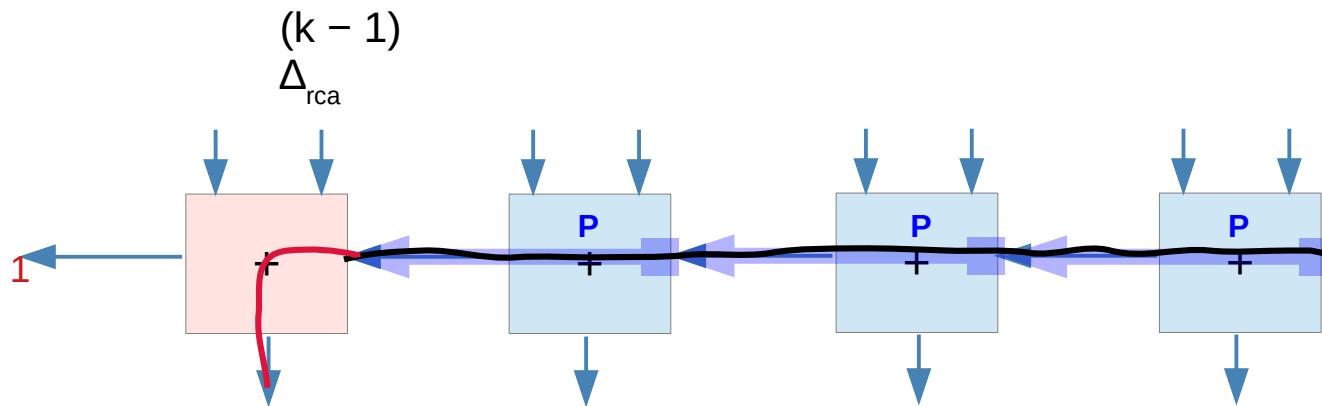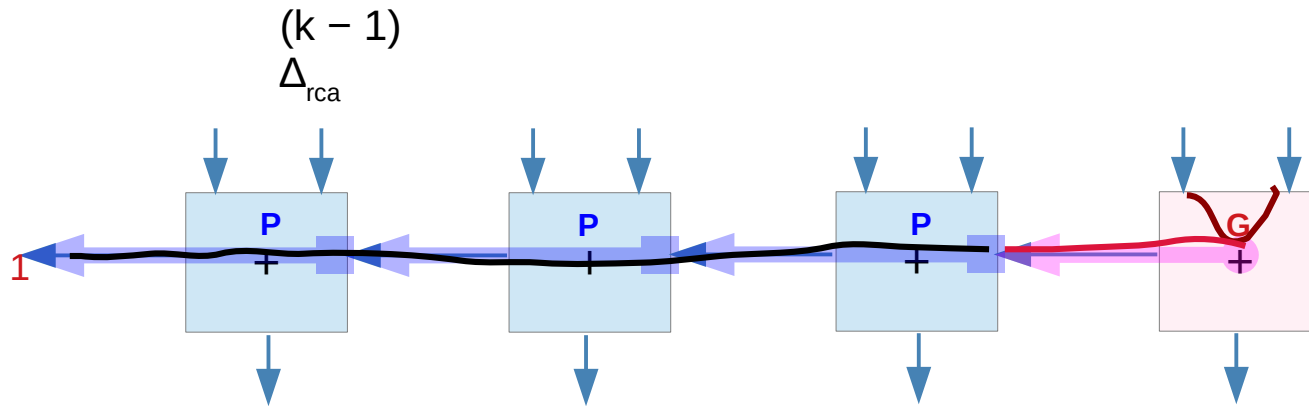
The delay is still linearly dependent on the size of the adder N,
however this linear dependence is reduced by a factor of *1/k*

Oklobdzija: High-Speed VLSI arithmetic units : adders and multipliers

(k − 1)
Δ_{rca}

(k − 1)
Δ_{rca}

High Performance Carry Chains for FPGAs, S. Hauck, M. M. Hosler, T. W. Fry

Carry Skip Adder

If an arbitrary block generated a carry by itself,
The carry will always propagate to the next block
However, if the second block generates a carry itself,
Or kill the carry, then that is the end of the critical path

If the second block propagates the carry, then we see
The advantage of the CSA architecture

https::/electronics.stackexchange.com/questions/21251/critical-path-for-carry-skip-adder

References

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