Carry Chain Adder (6A)

-
-

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

This document was produced by using OpenOffice and Octave.

Manchester Carry Chain

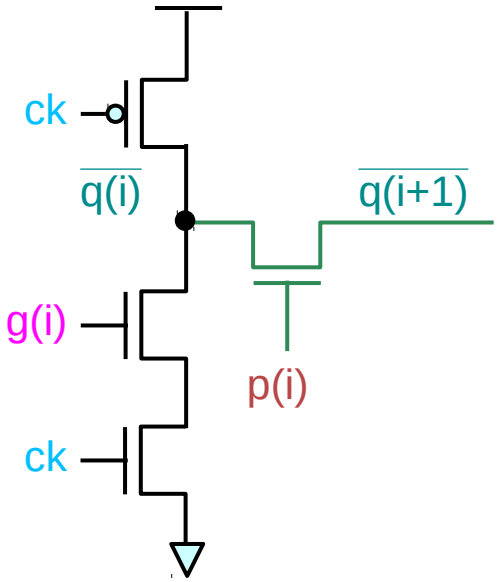
the output node $\overline{q(i+1)}$ is precharged,
when the synchronization signal is equal to 0 (ck=0),

when ck=1, the output node is discharged
if either p(i) =1 and the preceding node $\overline{q(i)}$
has been discharged, or
if g(i)=1.

In order that it works properly
g(i) and p(i) should not be equal to 1 simultaneously
so that the definition of g(i) cannot be relaxed
as in the preceding case

p(i)	0	1
0	0	1
1	1	0

g(i)	0	1
0	0	0
1	0	1



Manchester Carry Chain

Synthesis of Arithmetic Circuits: FPGA, ASIC and Ebedded Systems, J-P Deschamps et al

Faster Adders

Bottle neck : any stage needs information regarding all preceding carry bits
to be able to compute its own sum and carry-out bits

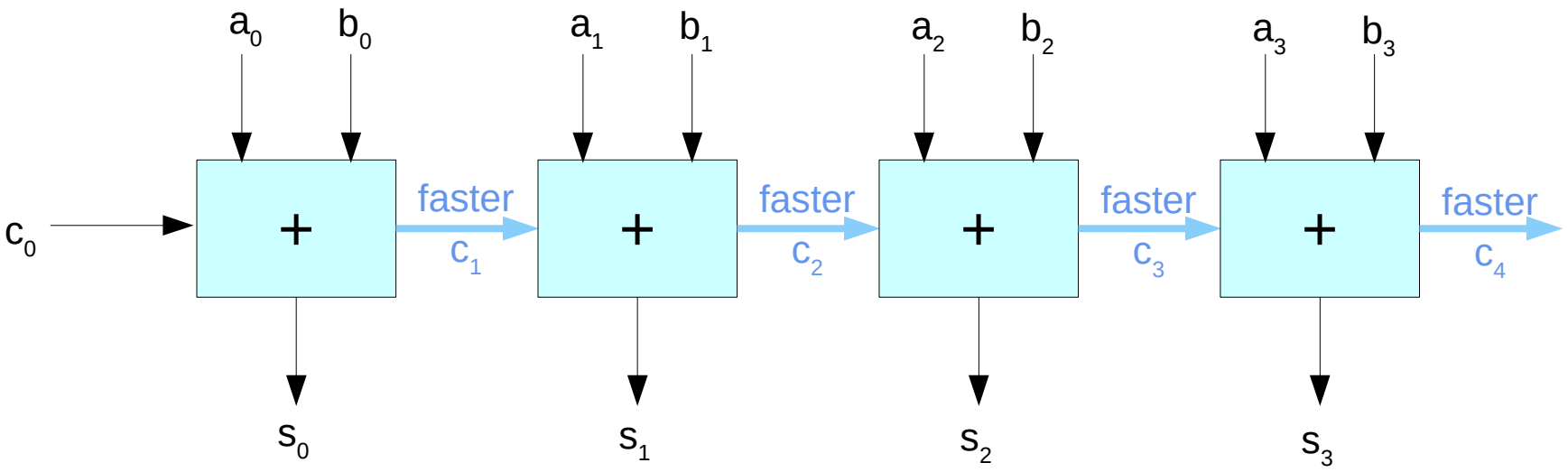
Faster Adders

1) **faster carry propagation** Manchester Carry Chain Adder
reduction of the time required
for the carry signal to propagate to the cell

2) **faster carry generation** Carry Lookahead Adder
local computation of the carry, without having to wait for the carry-out
produced by preceding stages

Principles of CMOS VLSI design – A Systems Perspective, N Weste, K Eshraghian

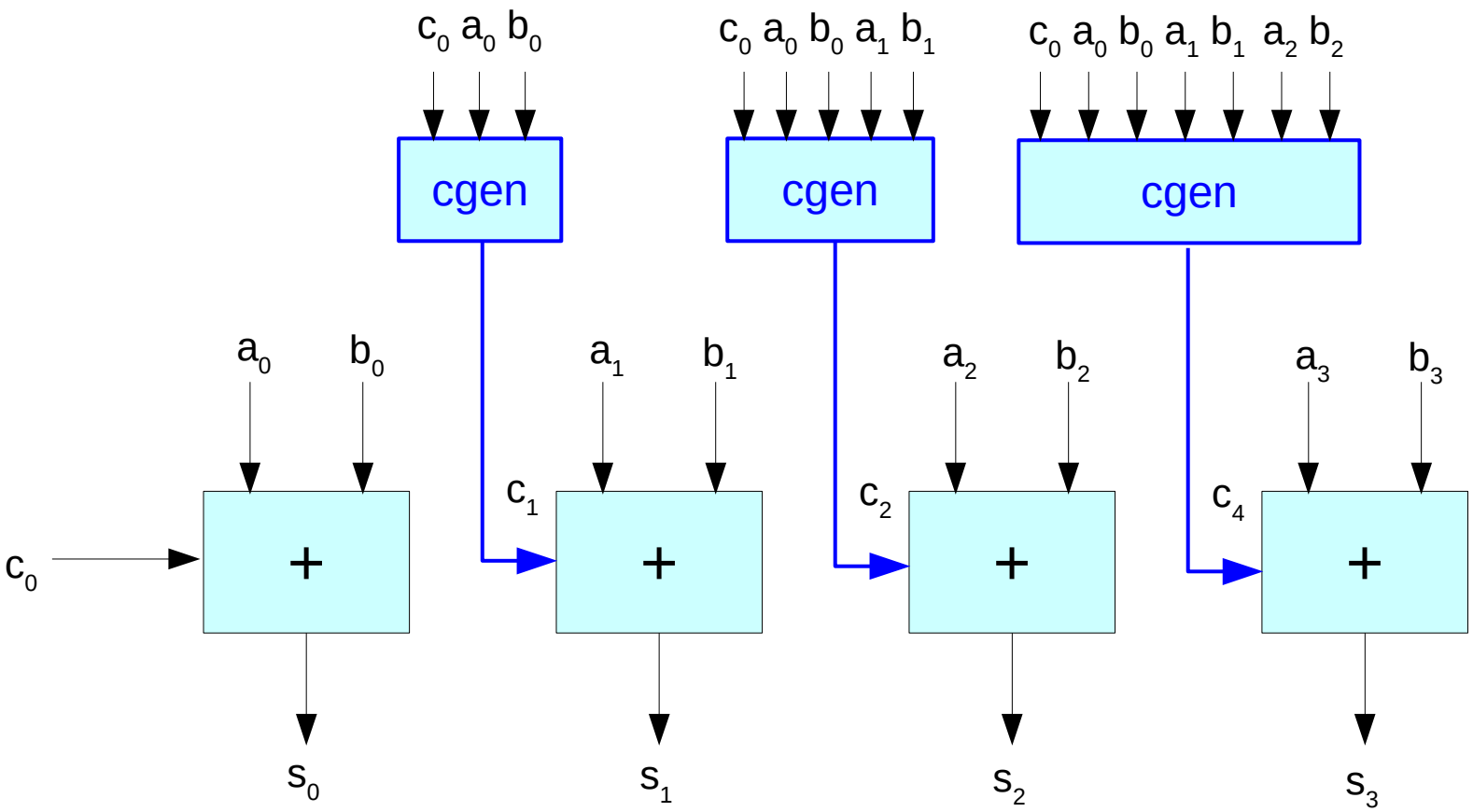
Faster Carry Propagation



Reduces the time needed for the carry to propagate through the cells

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

Faster Carry Generation



Each stage computes its own carry-in bit

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

Manchester Carry-Chain Adder

Manchester Carry-Chain Adder
--- faster carry propagation
carry propagate adder in which
the delay through the carry cells is reduced

Static or Dynamic Circuits

Thanks to the parameter G and P,
The delay is just one gate-delay

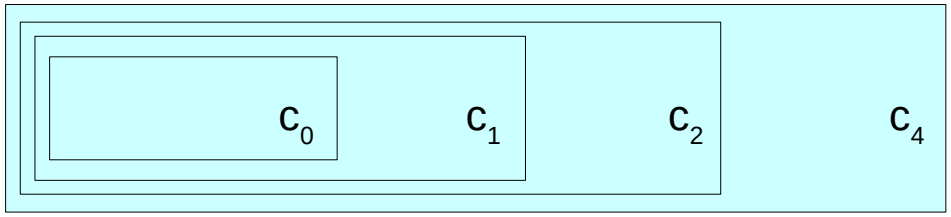
Principles of CMOS VLSI design – A Systems Perspective, N Weste, K Eshraghian

Manchester Carry Chain – Shared Logic

The **Manchester carry chain** is
a variation of the **carry-lookahead adder**
that uses **shared logic** to lower the transistor count.

the logic for **generating** each carry contains
all of the logic used to **generate the previous carries**.

A Manchester carry chain
generates the **intermediate carries**
by tapping off nodes in the gate that calculates
the **most significant carry value**.

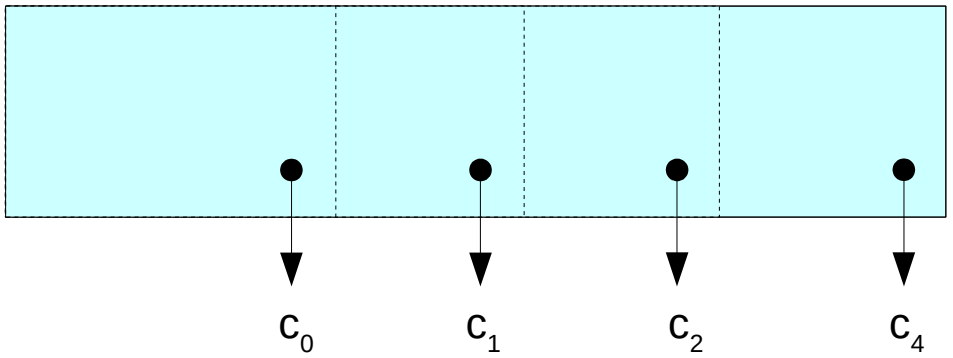


https://en.wikipedia.org/wiki/Carry-lookahead_adder

Manchester Carry Chain – tap to internal nodes

The Manchester adder stage improves on the carry-lookahead implementation by using a single $\mathbf{C_3}$ circuit

$\mathbf{C_2}$, $\mathbf{C_1}$, $\mathbf{C_0}$, are tapped to the internal nodes of the single $\mathbf{C_3}$ circuit



Principles of CMOS VLSI design – A Systems Perspective, N Weste, K Eshraghian

In addition to four Manchester stages,
the adder requires four PG generator blocks
One representative implementation

Four SUM generate blocks an XNOR gate complete the adder
This worst case propagation time can be improved
by bypassing the four stages if all carry propagate signals are true

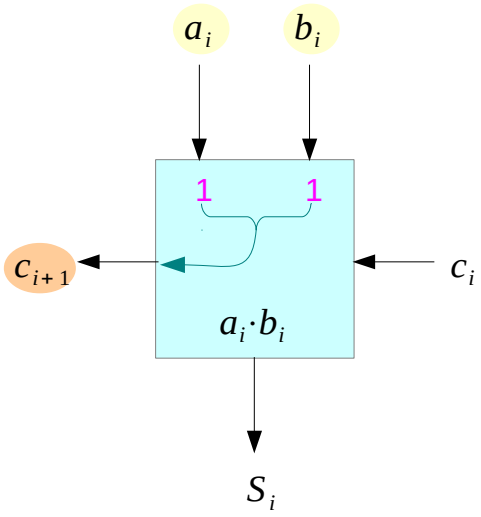
The optimum number of cascaded stages
may be calculated for a given technology by simulation
A final implementation of a 4-bit Manchester adder

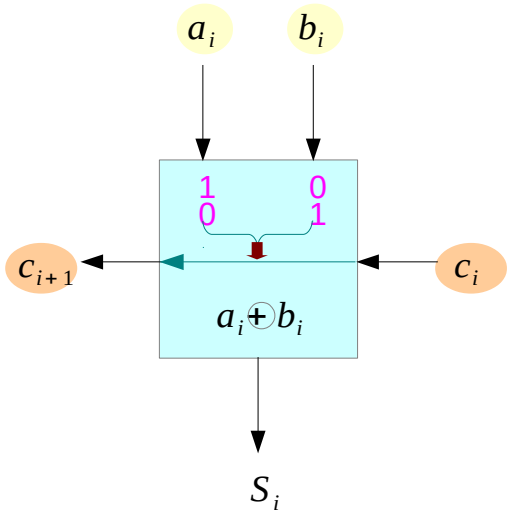
Principles of CMOS VLSI design – A Systems Perspective, N Weste, K Eshraghian

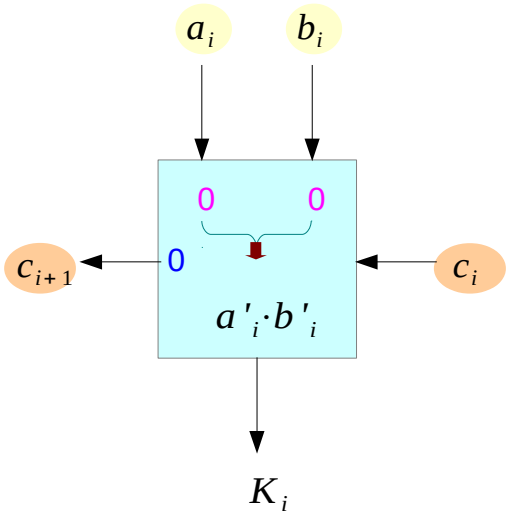
G, P, and K

Generate	$G_i = a_i \cdot b_i$
Propagate	$P_i = a_i \oplus b_i$
Kill	$K_i = \bar{a}_i \cdot \bar{b}_i$

$$c_{out} = G_i + P_i c_i$$



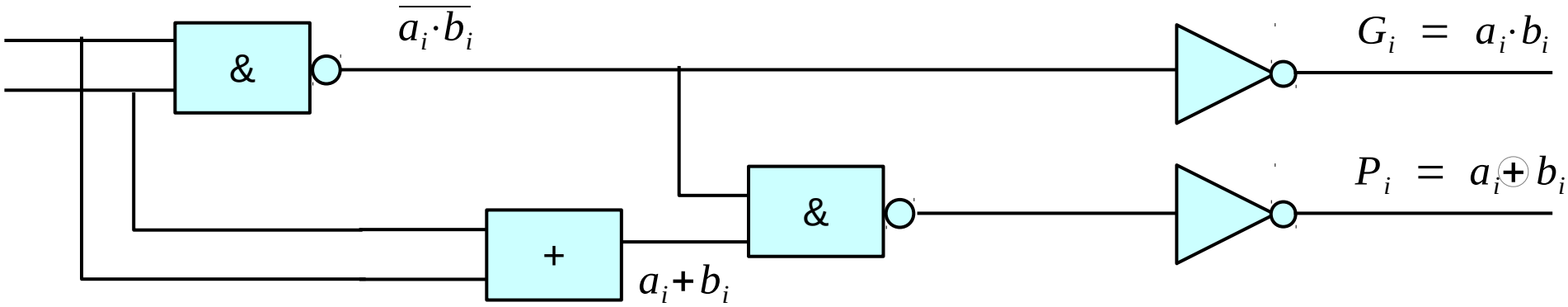




PG Circuits

Generate $G_i = a_i \cdot b_i$

Propagate $P_i = a_i \oplus b_i$



$\overline{(a_i+b_i)}\,\overline{(a_i \cdot b_i)}$

$\overline{(a_i+b_i)}+(a_i \cdot b_i)$

$\overline{a_i} \cdot \overline{b_i} + a_i \cdot b_i$

$(a_i+b_i)\,\overline{(a_i \cdot b_i)}$

$(a_i+b_i) \cdot (\overline{a_i}+\overline{b_i})$

$\overline{a_i} \cdot b_i + a_i \cdot \overline{b_i}$

References

[1] http://en.wikipedia.org/
[2] J-P Deschamps,et. al., “Sunthesis of Arithmetic Circuits”, 2006