

FPGA Carry Chain Adder (1A)

-
-

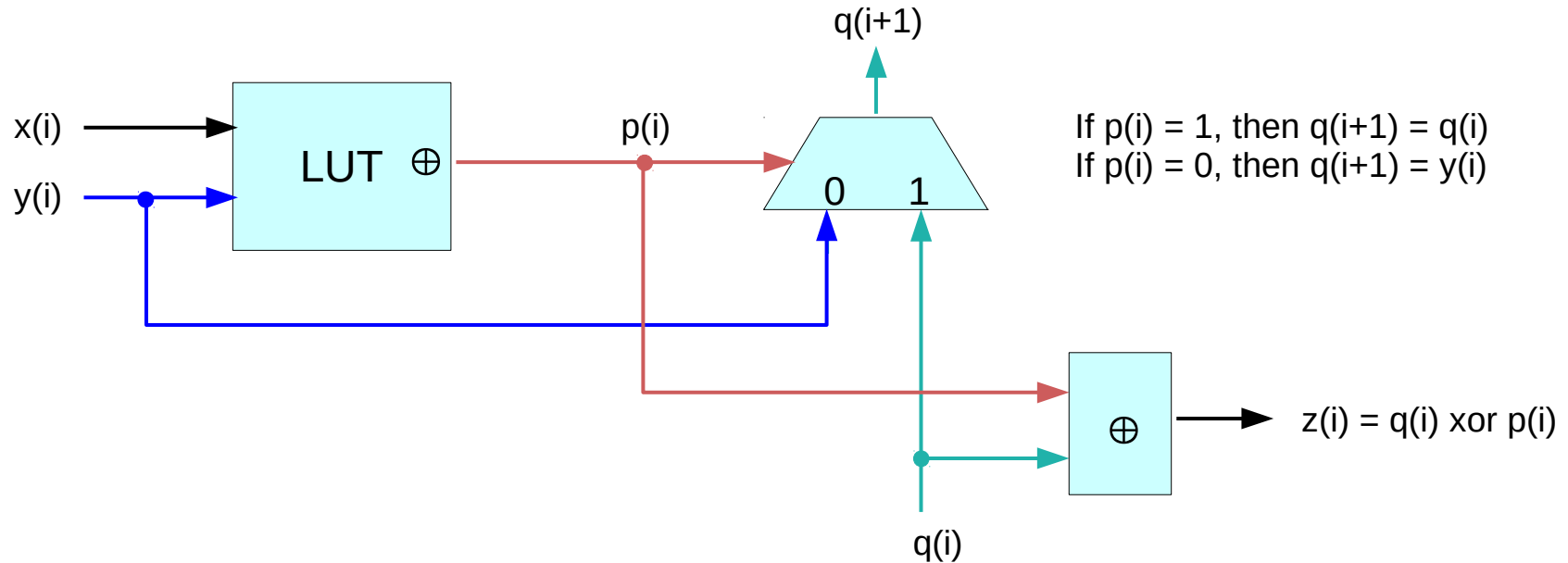
Copyright (c) 2010 -- 2020 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

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

This document was produced by using OpenOffice and Octave.

FPGA Carry Chain Cell



$$s_i = (a_i \oplus b_i) \oplus c_i = p_i \oplus c_i$$

$$c_{i+1} = (a_i \cdot b_i) + (a_i \oplus b_i) c_i = \bar{p}_i \cdot g_i + p_i \cdot c_i = \bar{p}_i \cdot a_i + p_i \cdot c_i = \bar{p}_i \cdot b_i + p_i \cdot c_i$$

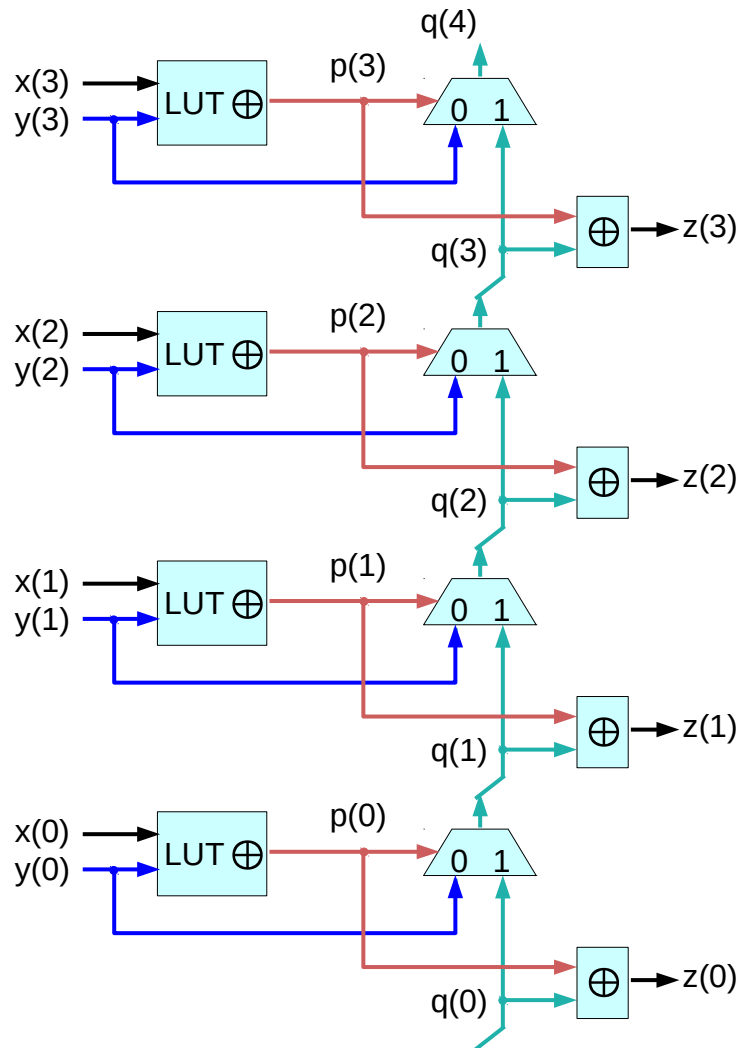
when $\bar{p}_i = 1$, then $a_i = b_i$

when $g_i = 1$, then $a_i = b_i = 1$

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

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

FPGA Carry Chain Cell



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

FPGA Carry Chain

FPGAs generally contain dedicated computation resources for generating fast adders

The Virtex family programmable arrays include logic gates (**XOR**) and **multiplexers** that along with the general purpose **lookup tables** allow one to build effective carry-chain adders

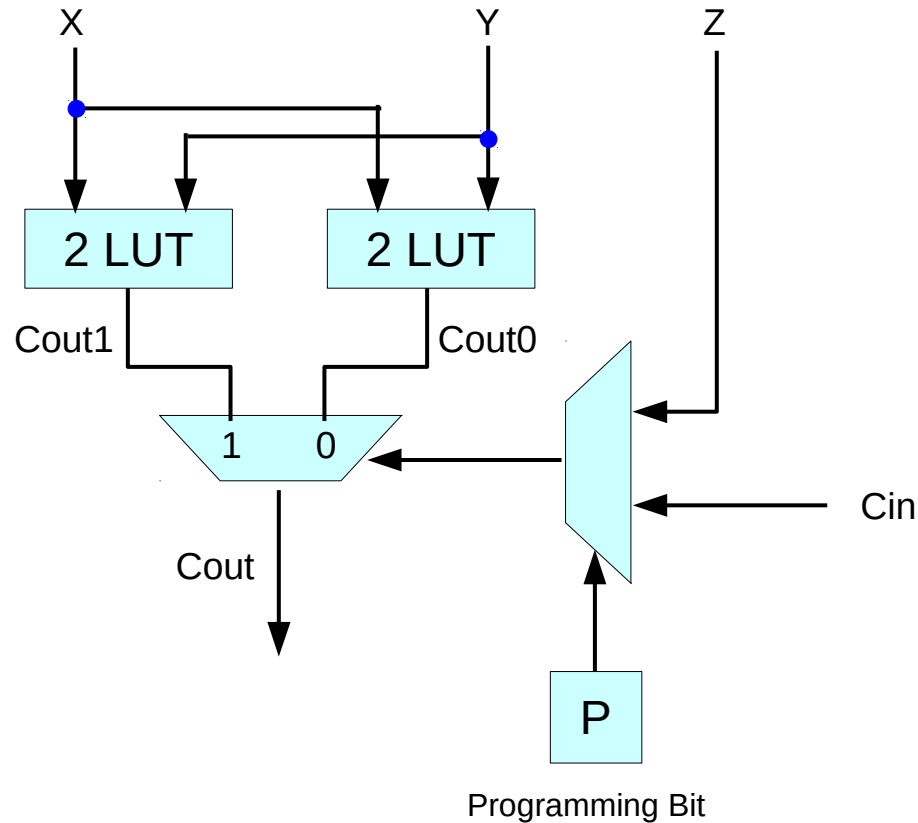
The carry chain is made up of multiplexers belonging to adjacent configurable blocks

the lookup table is used for implementing the exclusive or function

$$p(i) = x(i) \text{ xor } y(i)$$

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

FPGA Carry Chain Cell



Cout1, Cout2 : functions of X, Y, Cin

Cout1 = X+Y when Cin=1

Cout0 = X Y when Cin=0

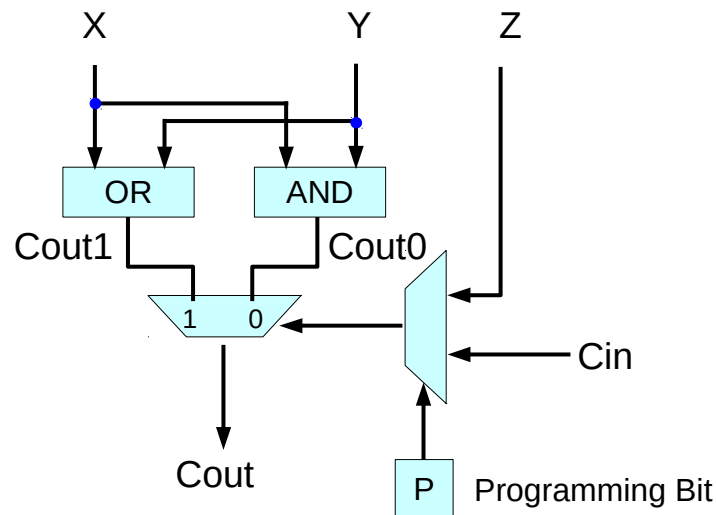
Cout = (X + Y) Cin + X Y $\overline{\text{Cin}}$

Cout = P' Cin + G $\overline{\text{Cin}}$... P' = relaxed P

Cout1	Cout0	Cout	Name
0	0	0	Kill
0	1	$\overline{\text{Cin}}$	Inverse Propagate
1	0	Cin	Propagate
1	1	1	Generate

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

FPGA Carry Chain Cell



X	Y	Cin	$\overline{\text{Cin}}$	$\overline{X} \overline{Y}$
		Cout1	Cout0	
0	0	0	0	$\overline{X} \overline{Y}$
0	1	1	0	$\overline{X} Y$
1	0	1	0	$X \overline{Y}$
1	1	1	1	$X Y$

Cout : functions of X, Y, Cin

$$\text{Cout}(X, Y, 1) = \text{Cout1} = X + Y$$

$$\text{Cout}(X, Y, 0) = \text{Cout0} = X Y$$

$$\text{Cout1} = X + Y \text{ when Cin}=1$$

$$\text{Cout0} = X Y \text{ when Cin}=0$$

$$\text{Cout1} = P' \text{Cin} \dots P' = \text{relaxed } P$$

$$\text{Cout0} = G \overline{\text{Cin}}$$

$$\text{If Cin, then Cout} = (\overline{X} Y + X \overline{Y} + X Y)$$

$$\text{If } \overline{\text{Cin}}, \text{ then Cout} = X Y$$

$$\text{Cin} (X + Y) + \overline{\text{Cin}} X Y$$

$$\text{Cin} (\overline{X} Y + X \overline{Y} + X Y) + \overline{\text{Cin}} X Y$$

$$\text{Cin} (\overline{X} Y + X \overline{Y}) + (\text{Cin} + \overline{\text{Cin}}) X Y$$

$$P \text{Cin} + G$$

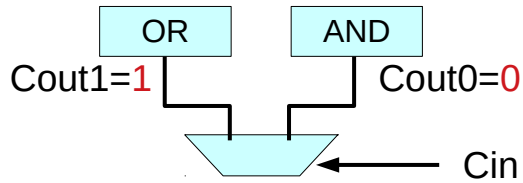
$$\text{Cin} (X + Y) + \overline{\text{Cin}} X Y$$

$$\text{Cin } P' + \overline{\text{Cin}} G$$

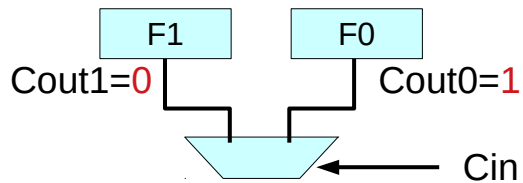
$$\dots P' : \text{relaxed } P$$

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

FPGA Carry Chain Cell



Cout1=1 when Cin=1
 Cout0=0 when Cin=0
 Cout = Cin



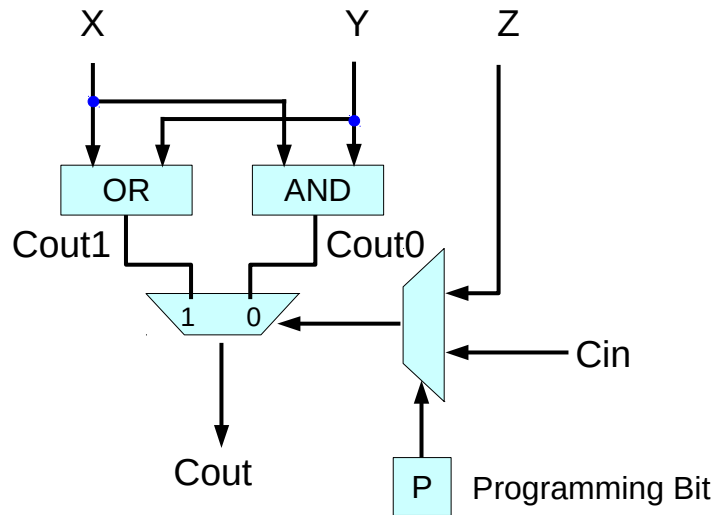
Cout1=0 when Cin=1
 Cout0=1 when Cin=0
 Cout = $\overline{\text{Cin}}$

Cout0	Cout1	Cout	Name
0	0	0	Kill
0	1	$\overline{\text{Cin}}$	Propagate
1	0	$\overline{\text{Cin}}$	Inverse Propagate
1	1	1	Generate

Cout1	Cout0	Cout	Name
0	0	0	Kill
0	1	$\overline{\text{Cin}}$	Inverse Propagate
1	0	$\overline{\text{Cin}}$	Propagate
1	1	1	Generate

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

Carry Chain



Carry Out

X	Y	Cin	Cout
0	0	Cin	$\overline{\text{Cin}}$
0	1	Cin	$\overline{\text{Cin}}$
1	0	Cin	$\overline{\text{Cin}}$
1	1	Cin	Cin

X	Y	Cin	$\overline{\text{Cin}}$	$\overline{\text{X}} \overline{\text{Y}}$
		Cout1	Cout0	
0	0	0	0	$\overline{\text{X}} \overline{\text{Y}}$
0	1	1	0	$\overline{\text{X}} \text{Y}$
1	0	1	0	$\text{X} \overline{\text{Y}}$
1	1	1	1	$\text{X} \text{Y}$

Cout1	Cout0	Cout	Name
0	0	0	Kill
0	1	$\overline{\text{Cin}}$	Inverse Propagate
1	0	Cin	Propagate
1	1	1	Generate

Cout1=1 when Cin=1

Cout0=0 when Cin=0

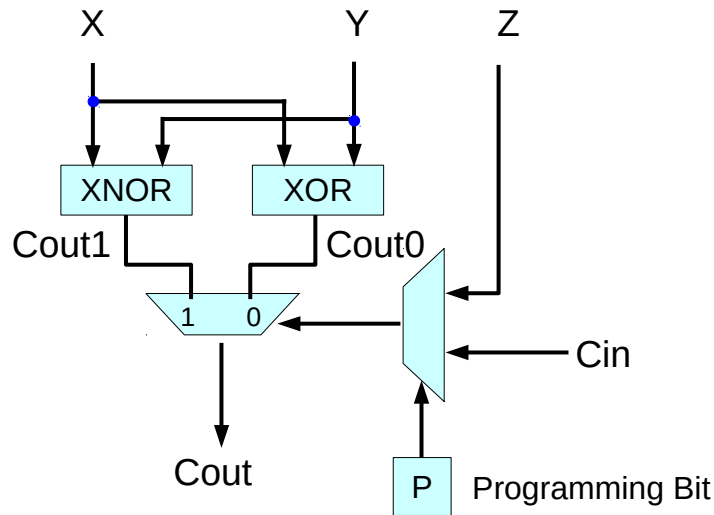
Cout = Cin propagate

Cout1=0 when Cin=1

Cout0=1 when Cin=0

Cout = $\overline{\text{Cin}}$ inverse propagate

Parity Checker



X	Y	Cin	$\overline{\text{Cin}}$	
		Cout1	Cout0	
0	0	1	0	$\overline{X} \overline{Y}$
0	1	0	1	$\overline{X} Y$
1	0	0	1	$X \overline{Y}$
1	1	1	0	$X Y$

Cout1	Cout0	Cout	Name
0	0	0	Kill
0	1	$\overline{\text{Cin}}$	Inverse Propagate
1	0	Cin	Propagate
1	1	1	Generate

Computing Parity

$X \oplus Y \oplus \text{Cin}$	
$0 \oplus 0 \oplus \text{Cin}$	$\overline{\text{Cin}}$
$0 \oplus 1 \oplus \text{Cin}$	$\overline{\text{Cin}}$
$1 \oplus 0 \oplus \text{Cin}$	$\overline{\text{Cin}}$
$1 \oplus 1 \oplus \text{Cin}$	Cin

Cout1=1 when Cin=1

Cout0=0 when Cin=0

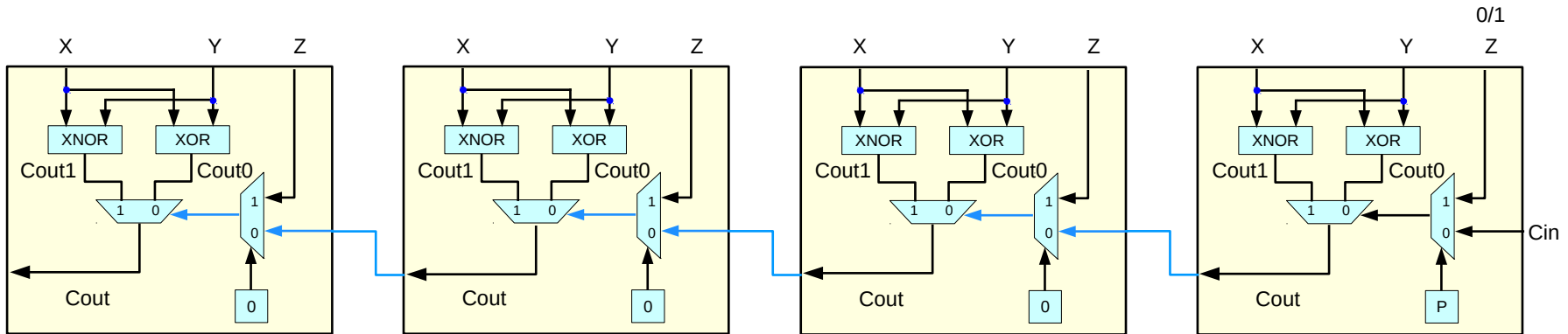
Cout = Cin propagate

Cout1=0 when Cin=1

Cout0=1 when Cin=0

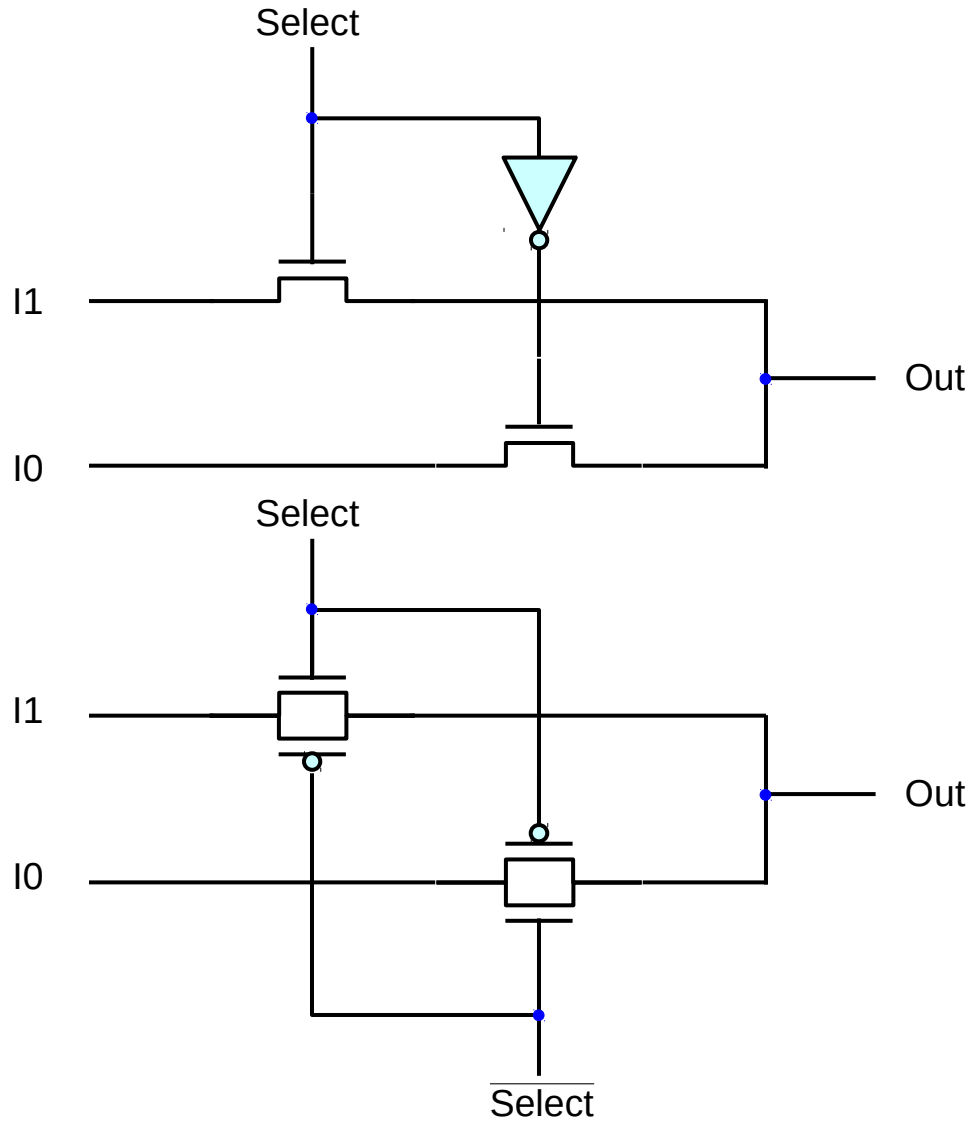
Cout = $\overline{\text{Cin}}$ inverse propagate

Ripple Carry Chain



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

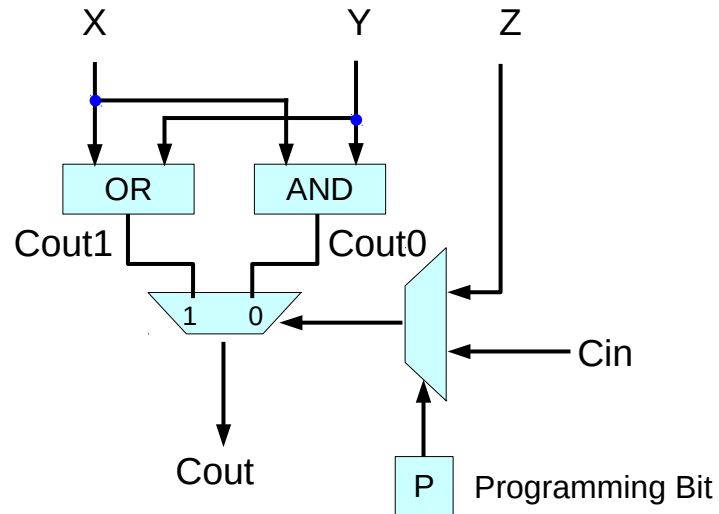
Ripple Carry Chain



P

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

FPGA Carry Chain Cell



Cout1, Cout2 : functions of X, Y, Cin

Cout1 = X+Y when Cin=1

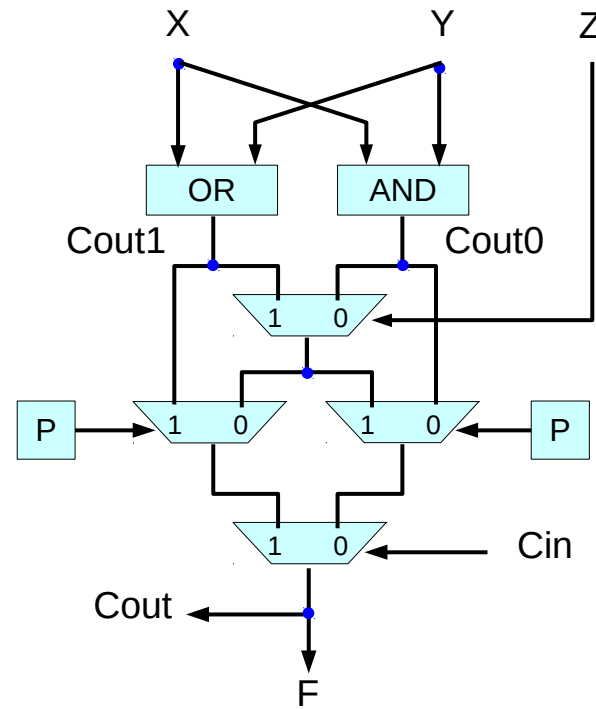
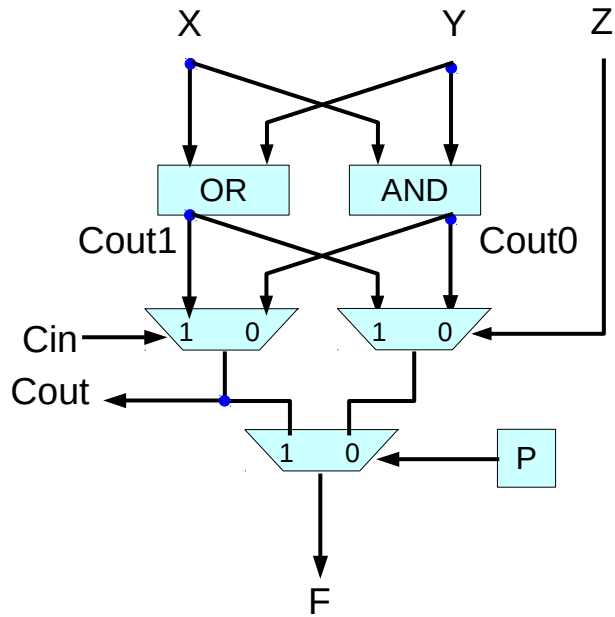
Cout0 = X Y when Cin=0

$Cout1 = \overline{Cin} (X + Y)$

$Cout0 = Cin X Y$

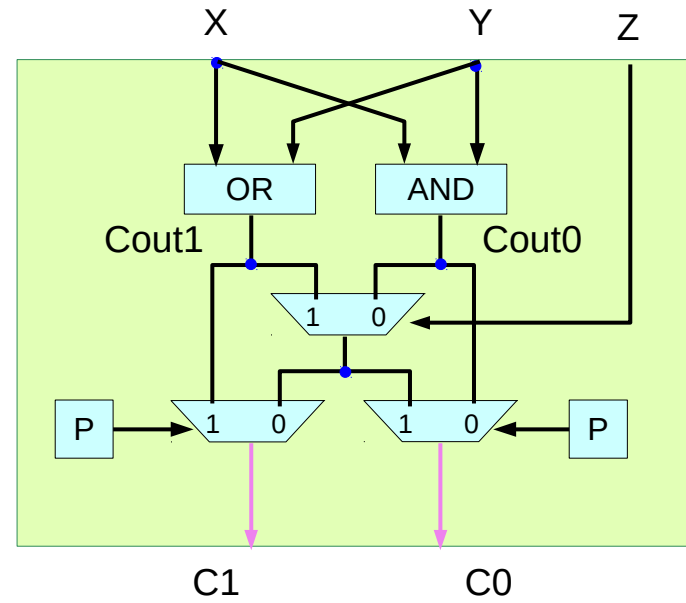
High Performance Carry Chains for FPGAs, S. Hauck, M. M. Hosler, T. W. Fry

FPGA Carry Chain Cell



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

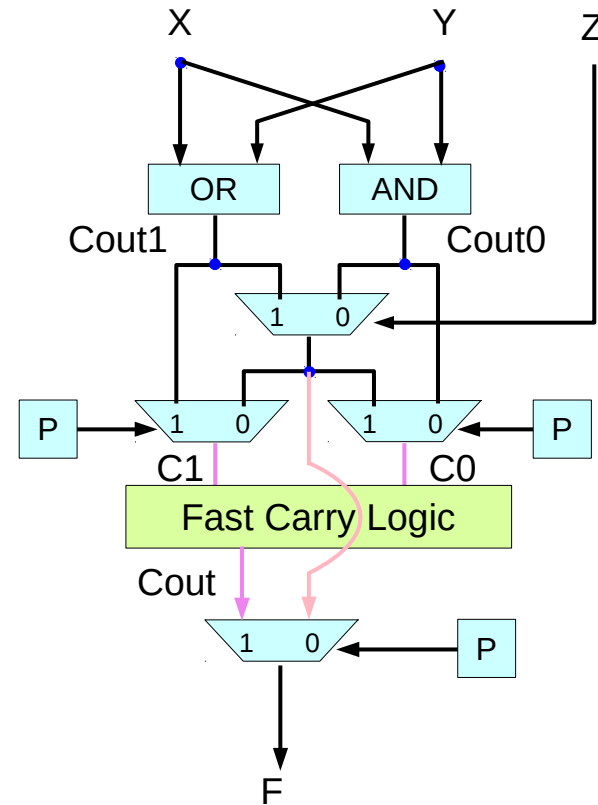
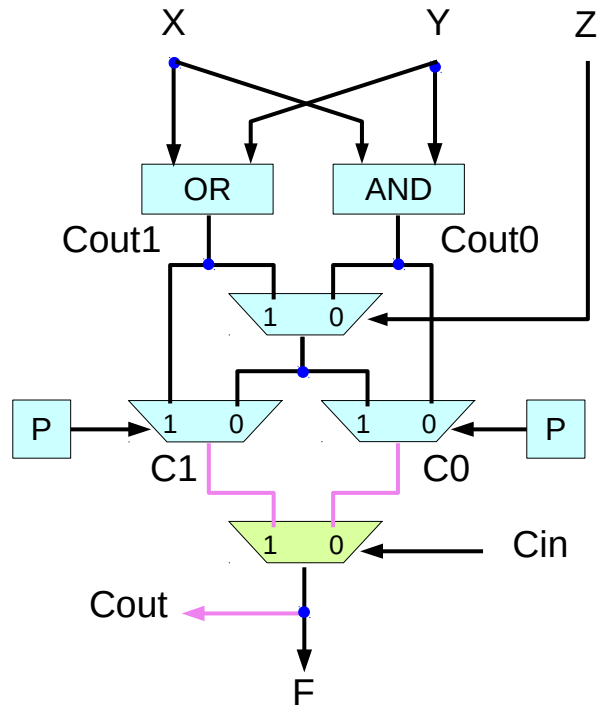
FPGA Carry Chain Cell



$$Cout_i = (Cout_{i-1} \cdot C1_i) + (\overline{Cout_{i-1}} \cdot C0_i)$$

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

FPGA Carry Chain Cell



$$Cout_i = (Cout_{i-1} \cdot C1_i) + (\overline{Cout_{i-1}} \cdot C0_i)$$

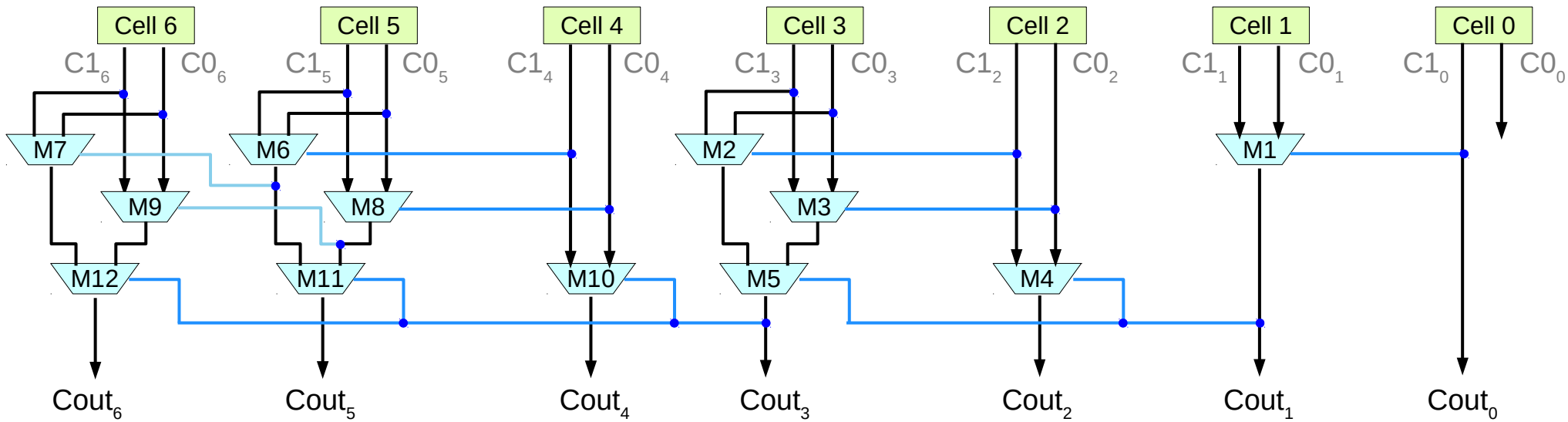
High Performance Carry Chains for FPGAs, S. Hauck, M. M. Hosler, T. W. Fry

Fast Carry Logc

Carry Select Adder
Carry Lookahead Adder
 Brent-Kung
Variable Block
Ripple Carry Adder

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

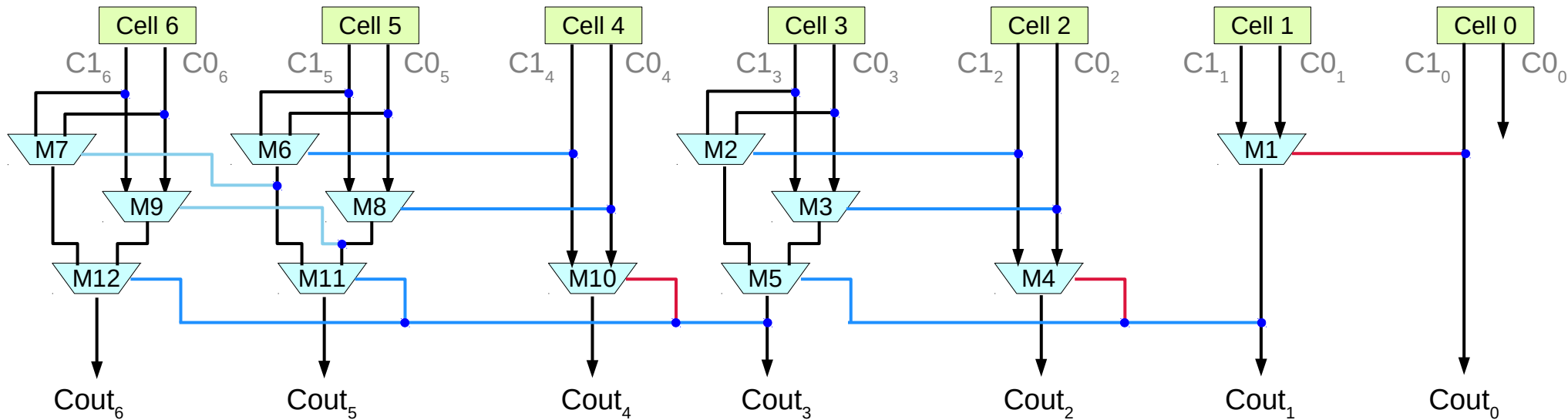
FPGA Carry Chain Cell



$$Cout_i = (Cout_{i-1} \cdot C1_i) + (\overline{Cout_{i-1}} \cdot C0_i)$$

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

FPGA Carry Chain Cell



$$Cout_i = (Cout_{i-1} \cdot C1_i) + (\overline{Cout_{i-1}} \cdot C0_i)$$

$$Cout_1 = (Cout_0 \cdot C1_1) + (\overline{Cout_0} \cdot C0_1)$$

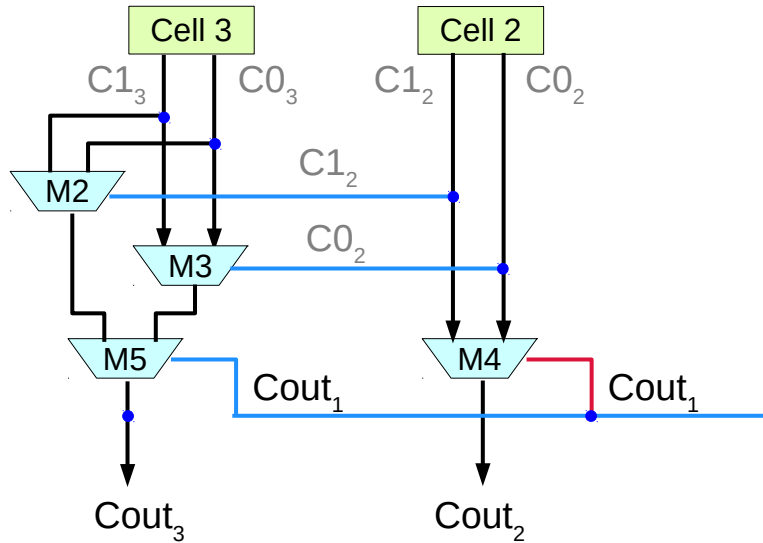
$$Cout_{i+1} = (Cout_i \cdot C1_{i+1}) + (\overline{Cout_i} \cdot C0_{i+1})$$

$$Cout_1 = (C1_0 \cdot C1_1) + (\overline{C1_0} \cdot C0_1)$$

$$Cout_{i+1} = (((Cout_{i-1} \cdot C1_i) + (\overline{Cout_{i-1}} \cdot C0_i)) \cdot C1_{i+1}) + (\overline{((Cout_{i-1} \cdot C1_i) + (\overline{Cout_{i-1}} \cdot C0_i))} \cdot C0_{i+1})$$

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

FPGA Carry Chain Cell



$$Cout_i = (Cout_{i-1} \cdot C1_i) + (\overline{Cout_{i-1}} \cdot C0_i)$$

$$Cout_{i+1} = (Cout_i \cdot C1_{i+1}) + (\overline{Cout_i} \cdot C0_{i+1})$$

$$Cout_2 = (Cout_1 \cdot C1_2) + (\overline{Cout_1} \cdot C0_2)$$

$$Cout_3 = (Cout_2 \cdot C1_3) + (\overline{Cout_2} \cdot C0_3)$$

$$= (((Cout_1 \cdot C1_2) + (\overline{Cout_1} \cdot C0_2)) \cdot C1_3)$$

$$+ (((Cout_1 \cdot C1_2) + (\overline{Cout_1} \cdot C0_2)) \cdot C0_3)$$

$$(((Cout_1 \cdot C1_2) + (\overline{Cout_1} \cdot C0_2)) \cdot C1_3)$$

$$(C1_3 C1_2 Cout_1 + C1_3 C0_2 \overline{Cout_1})$$

$$(C1_3 C1_2 + C0_3 \overline{C1_2}) Cout_1 + (C1_3 C0_2 + C0_3 \overline{C0_2}) \overline{Cout_1}$$

$$(((\overline{Cout_1} \cdot C1_2) + (\overline{Cout_1} \cdot C0_2)) \cdot C0_3)$$

$$= (((\overline{Cout_1} + \overline{C1_2}) \cdot (\overline{Cout_1} + \overline{C0_2})) \cdot C0_3)$$

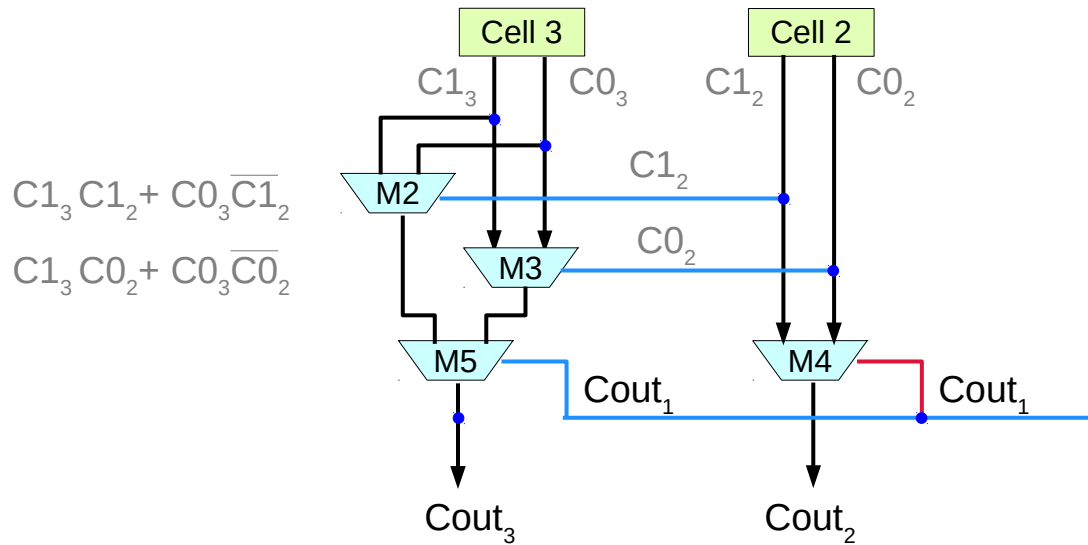
$$= (\overline{Cout_1} \overline{Cout_1} + \overline{C1_2} \overline{Cout_1} + \overline{Cout_1} \overline{C0_2} + \overline{C1_2} \overline{C0_2}) \cdot C0_3$$

$$= (\overline{C1_2} \overline{Cout_1} + \overline{C0_2} \overline{Cout_1}) \cdot C0_3$$

$$= (C0_3 \overline{C1_2} \overline{Cout_1} + C0_3 \overline{C0_2} \overline{Cout_1})$$

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

FPGA Carry Chain Cell

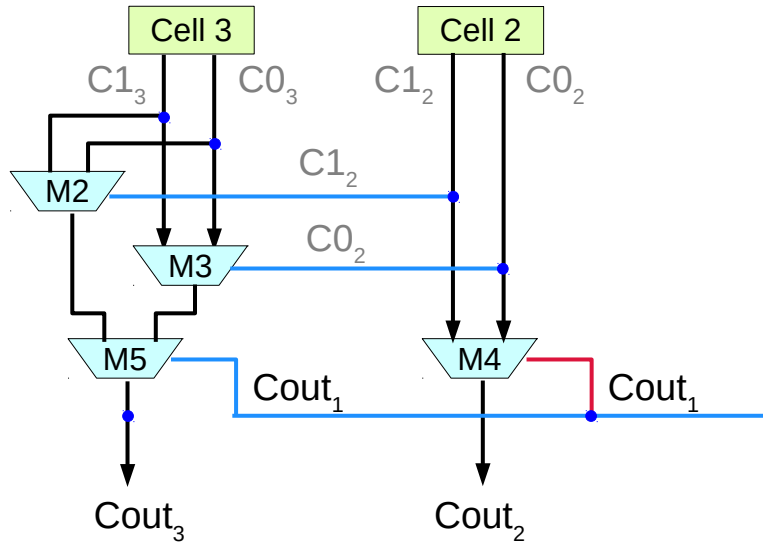


$$(C1_3 C1_2 + C0_3 \overline{C1_2})Cout_1 + (C1_3 C0_2 + C0_3 \overline{C0_2})\overline{Cout_1}$$

$$= C1_3 \cdot (C1_2 Cout_1 + C0_2 \overline{Cout_1}) + C0_3 \cdot (\overline{C1_2} Cout_1 + \overline{C0_2} \overline{Cout_1})$$

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

FPGA Carry Chain Cell



$$Cout_i = (Cout_{i-1} \cdot C1_i) + (\overline{Cout_{i-1}} \cdot C0_i)$$

$$Cout_{i+1} = (Cout_i \cdot C1_{i+1}) + (\overline{Cout_i} \cdot C0_{i+1})$$

$$Cout_{i+1} = \left(\left[(Cout_{i-1} \cdot C1_i) + (\overline{Cout_{i-1}} \cdot C0_i) \right] \cdot C1_{i+1} \right) + \left(\left[(Cout_{i-1} \cdot C1_i) + (\overline{Cout_{i-1}} \cdot C0_i) \right] \cdot C0_{i+1} \right)$$

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

References

[1] <http://en.wikipedia.org/>

[2] J-P Deschamps, et. al., “Sunthesis of Arithmetic Circuits”, 2006