

# Computer Architecture: Adders

Kasper Engelen

Sam Pieters

Viktor Hura

# Addition

$$\begin{array}{r} 00101110 \\ + 00100111 \\ \hline \end{array}$$




# Addition

$$\begin{array}{r} 0101110 \\ 00101110 \\ + 00100111 \\ \hline 01010101 \end{array}$$



# Addition

$$\begin{array}{r} 0101110 \\ 00101110 \\ + 00100111 \\ \hline 01010101 \end{array}$$


# Addition

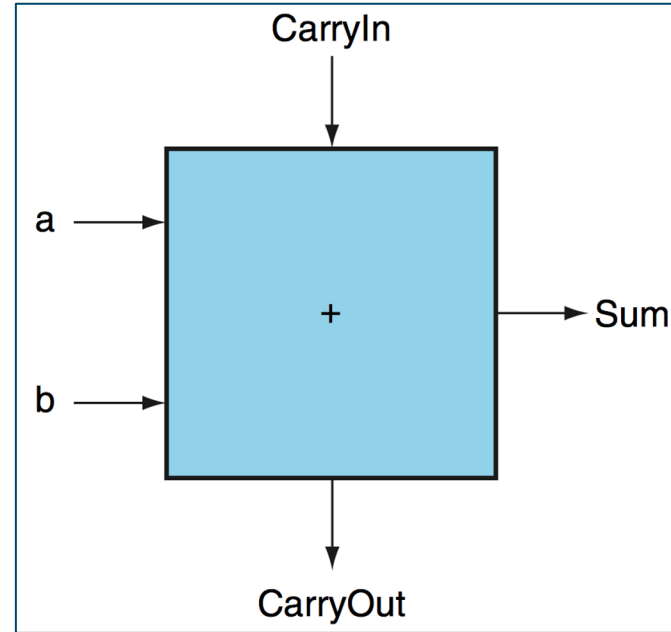
Inputs			Outputs		Comments
a	b	CarryIn	CarryOut	Sum	
0	0	0	0	0	$0 + 0 + 0 = 00_{\text{two}}$
0	0	1	0	1	$0 + 0 + 1 = 01_{\text{two}}$
0	1	0	0	1	$0 + 1 + 0 = 01_{\text{two}}$
0	1	1	1	0	$0 + 1 + 1 = 10_{\text{two}}$
1	0	0	0	1	$1 + 0 + 0 = 01_{\text{two}}$
1	0	1	1	0	$1 + 0 + 1 = 10_{\text{two}}$
1	1	0	1	0	$1 + 1 + 0 = 10_{\text{two}}$
1	1	1	1	1	$1 + 1 + 1 = 11_{\text{two}}$



# 1-Bit Full Adder

1-bit addition:

- CarryOut output is 1 when at least two inputs are 1
- Sum output is 1 when exactly one input is 1 or all three inputs are 1



# 1-Bit Full Adder

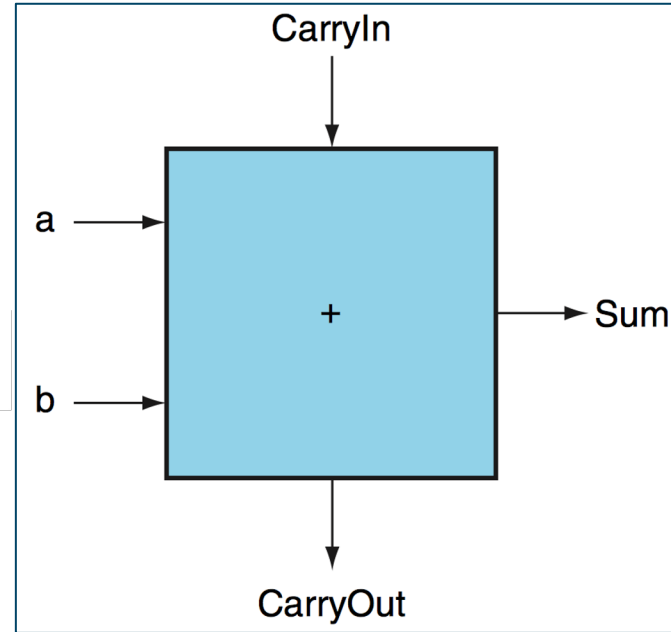
1-bit addition:

- CarryOut output is 1 when at least two inputs are 1

$$\text{CarryOut} = (b \cdot \text{CarryIn}) + (a \cdot \text{CarryIn}) + (a \cdot b)$$

- Sum output is 1 when exactly one input is 1 or all three inputs are 1

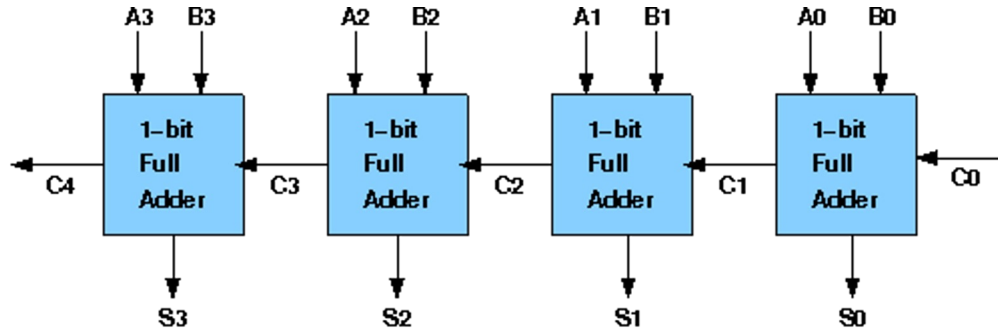
$$\text{Sum} = (a \cdot \bar{b} \cdot \overline{\text{CarryIn}}) + (\bar{a} \cdot b \cdot \overline{\text{CarryIn}}) + (\bar{a} \cdot \bar{b} \cdot \text{CarryIn}) + (a \cdot b \cdot \text{CarryIn})$$



# Ripple Carry Adder

n-bit addition:

- Series of 1-bit full adders
- Carry ripples through addition = Slow!





# Calculate Carry

$$\begin{array}{r} \text{????} \quad \text{????} \\ 0010 \quad 1110 \\ + 0010 \quad 0111 \\ \hline \end{array}$$

# Propagate and Generate

Generate: “When do  $a_i$  and  $b_i$  generate a carry?”

$$g_i = a_i \cdot b_i$$

Propagate: “When do  $a_i$  and  $b_i$  propagate a carry?”

$$p_i = a_i + b_i$$

CarryIn:

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

$$\begin{array}{r} c_3 \ c_2 \ c_1 \ c_0 \\ a_3 \ a_2 \ a_1 \ a_0 \\ + \ b_3 \ b_2 \ b_1 \ b_0 \\ \hline s_3 \ s_2 \ s_1 \ s_0 \end{array}$$



# Carry Lookahead Adder

$$c_1 = g_0 + (p_0 \cdot c_0)$$

$$c_2 = g_1 + (p_1 \cdot g_0) \\ + (p_1 \cdot p_0 \cdot c_0)$$

$$c_3 = g_2 + (p_2 \cdot g_1) \\ + (p_2 \cdot p_1 \cdot g_0) \\ + (p_2 \cdot p_1 \cdot p_0 \cdot c_0)$$

$$c_4 = g_3 + (p_3 \cdot g_2) \\ + (p_3 \cdot p_2 \cdot g_1) \\ + (p_3 \cdot p_2 \cdot p_1 \cdot g_0) \\ + (p_3 \cdot p_2 \cdot p_1 \cdot p_0 \cdot c_0)$$

$$\begin{array}{rcccc} & c_3 & c_2 & c_1 & c_0 \\ & a_3 & a_2 & a_1 & a_0 \\ + & b_3 & b_2 & b_1 & b_0 \\ \hline s_3 & s_2 & s_1 & s_0 & \end{array}$$

# Carry Lookahead Adder

$$\begin{array}{r} \textcircled{????} \\ 0010 \\ +0010 \\ \hline \end{array} \quad \begin{array}{r} ????? \\ 1110 \\ + 0111 \\ \hline \end{array}$$

# Super Propagates and -Generates

Calculate CarryIn of each 4-bit carry-lookahead adder.

Superpropagate  $P_i$  and supergenerate  $G_i$  :

$$P_0 = p_3 \cdot p_2 \cdot p_1 \cdot p_0$$

$$G_0 = g_3 + (p_3 \cdot g_2) + (p_3 \cdot p_2 \cdot g_1) + (p_3 \cdot p_2 \cdot p_1 \cdot g_0)$$

Calculate  $C_i$  :

$$C_1 = G_0 + (P_0 \cdot c_0)$$

$$C_2 = G_1 + (P_1 \cdot G_0) + (P_1 \cdot P_0 \cdot c_0)$$

$$C_3 = \dots$$



# 8-bit Carry Lookahead Adder

