Addition

Basic component: Full adder - combinational

\[
x \rightarrow FA \rightarrow c = \text{MSB}(x+y+z) = \text{majority}
\]
\[
y \rightarrow c \rightarrow s = \text{LSB}(x+y+z) = \text{parity}
\]

Problem:
Add 2 N-bit numbers

Ripple-carry adder

\[
\begin{array}{c}
 s_4 \\
 c_4 \\
 a_4 b_4 \\
 s_3 \\
 c_3 \\
 a_3 b_3 \\
 s_2 \\
 c_2 \\
 a_2 b_2 \\
 s_1 \\
 c_1 \\
 a_1 b_1 \\
 s_0 \\
 c_0
\end{array}
\]

N-bit #’s \(\Rightarrow\) \(\Theta(N)\) time, \(\Theta(N)\) HW, combinational

Serial adder

\(\Theta(N)\) time, \(\Theta(1)\) hardware; sequential (clocked)
Fast addition

Idea: carries are the hard part.
Know carries \( \Rightarrow \) compute sum in \( \Theta(n) \) time

Now? Array of full adders. <<Show on ripple-carry adder>>

<table>
<thead>
<tr>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

\[ g \quad p \quad g \quad k \quad p \quad g \quad (k) \]

Classify stages:

\[ \text{kill}: 0 \Rightarrow \text{carry-out} = 0 \]

\[ \text{propagate}: 0 \text{ or } 1 \Rightarrow \text{carry-out} = \text{carry-in} \]

\[ \text{generate}: 1 \Rightarrow \text{carry-out} = 1 \]

\( \text{Carry into stage} = \begin{cases} 1 & \text{if most recent non-} p \text{ is } k \\ 0 & \text{otherwise} \end{cases} \)

When do 2 consecutive stages kill, prop, gen?

\[
\begin{array}{c|ccc}
\otimes & k & p & g \\
\hline
k & k & k & g \\
\otimes & x_i & x_{i+1} \\
p & k & p & g \\
g & k & g & g \\
x_i & & & \\
\end{array}
\]

Associative!
Theorem. Let $x_i$ = carry status of stage $i$, where $x_0 = k$. Define $y_i = x_0 \otimes x_1 \otimes \ldots \otimes x_i$.

Then $y_i = k \Rightarrow c_i = 0$,
$y_i = g \Rightarrow c_i = 1$,
$y_i = p$ does not occur.

Proof. Induction on $i$. \(\Box\)

Log-time circuit:

$$
egin{align*}
Y_0 &= x_0 \\
Y_1 &= x_0 \otimes x_1 \\
Y_2 &= x_0 \otimes x_1 \otimes x_2 \\
Y_N &= x_0 \otimes x_1 \otimes \ldots \otimes x_N
\end{align*}
$$

Use tree for each calculation:

Use tree to broadcast inputs (bounded-degree network):

Time $= \Theta(\lg N)$, $HW = \Theta(N^2)$. 
Carry-lookahead addition

$\Theta(\lg N)$ time, $\Theta(N)$ HW.

"Parallel prefix"

Let $[i, j]$ denote $x_i \otimes x_{i+1} \otimes \ldots \otimes x_j$.

**Lemma.** $[i, j] \otimes [j+1, k] = [i', k'] \otimes$

$x_i = [0, i]$

$y_i = [0, i]$

Build tree:

Globally:

Left child values are passed up.
Similar method:

Left child values are passed up and right

Postscript: Kill, propagate, generate first used in standard relay calculator circa mid-1940's.

O(1)-time addition (in their model).