Adder Circuits
Unsigned Integer Addition

Addition of 1 bit addends

Truth Table

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>$c$</th>
<th>$s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Half Adder Circuit

Circuit

Graphical Symbol
Adding Larger Numbers

To add numbers that have more than 1 bits, we carry out a pairwise addition of each bit position.

If the result of the addition is > 1, a **carry-out** bit is generated and is included in the addition of the subsequent bit position (**carry-in**).
Full Adder Circuit

<table>
<thead>
<tr>
<th>$c_i$</th>
<th>$x_i$</th>
<th>$y_i$</th>
<th>$c_{i+1}$</th>
<th>$s_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Truth Table

Full Adder Circuit
Ripple Carry Adder

4-Bit Ripple Carry Adder

\[ \begin{array}{cccccc}
\text{FA} & x & y & c_o & s_0 \\
\text{FA} & x & y & c_o & s_1 \\
\text{FA} & x & y & c_o & s_2 \\
\text{FA} & x & y & c_o & s_2 \\
\end{array} \]

\[ \begin{array}{cccc}
y_0 & x_0 & \quad & c_{in} \\
y_1 & x_1 & \quad & \quad \\
y_2 & x_2 & \quad & \quad \\
y_3 & x_3 & \quad & \quad \\
\end{array} \]
Ripple Carry Adder Demo
Critical Path Delays

In a Ripple-Carry Adder, the delay increases as more bits are added.

The overall speed of any circuit is limited by the longest delay path through the circuit (Critical Path Delay).

The critical path delay of adders can be reduced, by reducing the delays to generate carry bits.
Fast Adders

The delays to the carry bits can be reduced by calculating them directly from the inputs, rather than the outputs of the previous stage.

There are two ways to produce a carry-out bit from each adder stage, *carry generate* and *carry propagate*.

Fast adders that calculate carries in advance are commonly referred to as *Carry-Lookahead Adders*.
Carry Lookahead Addition

**Carry Generate** \( (g_n) \) – If both bits of a stage are one, a carry will be generated.

**Carry Propagate** \( (p_n) \) – If a carry comes in a given stage, we will propagate that carry to the next stage if either bit is 1.
Carry Lookahead Addition

To reduce delays to carry outputs, we can rewrite $c_{n+1}$ in terms of $c_0, g, p$. 
First Two Stages of a Carry Lookahead Adder

Design Tradeoff: Speed for Complexity / Increased Fan-in
Comparison to Ripple Carry Adder

First two stages of a ripple-carry adder (using expressions for g and p)