Fast Adders

See: P&H Chapter 3.1-3, C.5-6
Goals:

serial to parallel conversion
time vs. space tradeoffs
design choices
- First full adder, 2 gate delay
- Second full adder, 2 gate delay
- ...

4-bit Ripple Carry Adder

Carry ripples from lsb to msb
Every bit needs to wait for carry in.

Q: Can we compute Cin earlier?

A: carry look-ahead adder (CLA)
For each bit, analyze situation independent of Cin

- Just based on (A,B) only

Q: When is Cout == 1, irrespective of Cin?
A: When A == 1 and B == 1
   (this bit *generates* a carry, irrespective of Cin)

Q: When else might Cout == 1?
A: When A == 1 or B == 1, and Cin == 1
   (this bit *propagates* carry from Cin to Cout)
Invent two new terms: propagator, generator

- $g = 1$ means this bit generates carry, irrespective of Cin
  - $g = AB$
- $p = 1$ means this bit propagates Cin to Cout, but doesn’t generate carry
  - $p = A \text{xor} B$

**Performance?**

- $p$ and $g$ generated in 1 gate delay after $A$ and $B$ arrive
- $R$ is 2 gate delay after Cin arrives
CLL inputs: \( p, g \) from all 4 bits, \( C_0 \)

CLL outputs: all carry bits (using just 2 gate delays)

\[
\begin{align*}
C_1 &= g_0 + p_0 C_0 \\
C_2 &= g_1 + p_1 g_0 + p_1 p_0 C_0 \\
C_3 &= g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 p_0 C_0 \\
C_4 &= g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 g_0 + p_3 p_2 p_1 p_0 C_0
\end{align*}
\]
4-bit CLA

CLL (carry look-ahead logic)
<table>
<thead>
<tr>
<th></th>
<th>Space</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 Bit Ripple</td>
<td>$4 \times 9 = 36$ gates (4 input)</td>
<td>$4 \times 2 = 8$ gate delays</td>
</tr>
<tr>
<td>4 Bit Look-Ahead</td>
<td>$4 \times 11 + 14 = 58$ gates (5 input)</td>
<td>5 gate delays</td>
</tr>
<tr>
<td>16 Bit Ripple</td>
<td>$16 \times 9 = 144$ gates (4 input)</td>
<td>$16 \times 2 = 32$ gate delays</td>
</tr>
<tr>
<td>16 Bit Look-Ahead</td>
<td>$16 \times 11 + 152 = 328$ gates (17 input)</td>
<td>5 gate delays</td>
</tr>
<tr>
<td>64 Bit Ripple</td>
<td>$64 \times 9 = 576$ gates (4 input)</td>
<td>$64 \times 2 = 128$ gate delays</td>
</tr>
<tr>
<td>64 Bit Look-Ahead</td>
<td>$64 \times 11 + 2144 = 2848$ gates (65 input)</td>
<td>5 gate delays</td>
</tr>
</tbody>
</table>
Only compute some fast carry signals
Carry-Skip Adder

Only compute some fast carry signals

Time: $\sim 2x$ faster than ripple
Space: $O(N)$ extra gates, $O(N)$ gate inputs
Hybrid Approach

Hybrid Adders

4-bit CLA

C16

4-bit CLA

C12

4-bit CLA

C8

4-bit CLA

C4

R15..8

R11..8

R7..4

R3..0

A15..8 B15..8

A11..8 B11..8

A7..4 B7..4

A3..0 B3..0

Carry ripples from lsb to msb
Hierarchical Approach

\[ A_{15..8} \quad B_{15..8} \]

\[ \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \]

\[ 4\text{-bit} \quad 4\text{-bit} \quad 4\text{-bit} \quad 4\text{-bit} \]

\[ g \quad g \quad g \quad g \]

\[ \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \]

\[ p \quad p \quad p \quad p \]

\[ \]

\[ C_{16} \quad \text{CLL (carry look-ahead logic)} \quad C_0 \]

\[ \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \]

\[ 4\text{-bit} \quad 4\text{-bit} \quad 4\text{-bit} \quad 4\text{-bit} \]

\[ \]

\[ R_{15..8} \quad R_{11..8} \quad R_{7..4} \quad R_{3..0} \]
\[ r4 = (r1 + r2) \mid r3 \]

\[ r8 = 4r3 + r4 - 1 \]

\[ r9 = 9 \]

\[ \text{ADDU} \ rd, \ rs, \ rt \]
\[ \text{SUBU} \ rd, \ rs, \ rt \]
\[ \text{OR} \ rd, \ rs, \ rt \]
\[ \text{XOR} \ rd, \ rs, \ rt \]
\[ \text{NOR} \ rd, \ rs \ rt \]
R-type instruction
R-type instruction

- Prog. Mem
- Inst
- PC
- +4
- Shamt
- Reg. File
- 5 5 5
- ALU
- Control
I-type instruction

- Program Memory (Prog. Mem)
- Instruction (inst)
- Register File (Reg. File)
- ALU
- Immediate (imm)
- Shamt (shamt)
- Control (control)
- Extend (extend)
- Program Counter (PC)