Logarithmic Lookahead Adders

$t_p \sim N$

$t_p \sim \log_2(N)$
Tree Adders

\[ P_G = p_m \cdot p_l \quad \text{m – more significant} \]
\[ G_G = g_m + p_m \cdot g_l \quad \text{l – less significant} \]

Start from the input P, G, and continue up the tree
2-bit groups, then 4-bit groups, …

\[ (g, p) = (g_m, p_m) \cdot (g_l, p_l) = (g_m + p_m \cdot g_l, p_m \cdot p_l) \]

Kogge, Stone, Trans on Comp.’73

---

Tree Adders

Associativity:
\[ i \geq k > j \]
\[ (G_{i:j}, P_{i:j}) = (G_{i:k}, P_{i:k}) \cdot (G_{k-1:j}, P_{k-1:j}) \]

Idempotency:
\[ (G, P) \cdot (G, P) = (G + P \cdot G, P \cdot P) = (P, G) \]

Associativity + Idempotency:
\[ i \geq k > j \quad v > k \]
\[ (G_{i:j}, P_{i:j}) = (G_{i:k}, P_{i:k}) \cdot (G_{v:j}, P_{v:j}) \]
Examples

16-bit ripple carry

\[(G_{15}, P_{15}) \cdot (G_{14}, P_{14}) \cdot \ldots \cdot (G_0, P_0) \cdot (1, c_0)\]

16-bit carry lookahead with 4-bit groups

\[(G_{15:12}, P_{15:12}) \cdot \{ (G_{11:8}, P_{11:8}) \cdot \{ (G_{7:4}, P_{7:4}) \cdot \{ (G_{3:0}, P_{3:0}) \cdot (1, c_0) \}\}\}\]

Brent-Kung Adder

\[t_{\text{add}} \sim \log_2(N)\]

Brent, Kung,
Trans on Comp, 3/82
Tree Adders

Ling Adder

Variation of CLA

\[ p_i = a_i \oplus b_i \]
\[ g_i = a_i \cdot b_i \]
\[ G_i = g_i + p_i \cdot G_{i-1} \]
\[ S_i = p_i \oplus G_{i-1} \]

Ling’s equations

\[ t_i = a_i + b_i \]
\[ g_i = a_i \cdot b_i \]
\[ H_i = g_i + t_{i-1} \cdot H_{i-1} \]
\[ S_i = t_i \oplus H_i + g_i t_{i-1} H_{i-1} \]

Ling, IBM J. Res. Dev, 5/81
Ling Adder

\[ G_i = g_i + p_i \cdot G_{i-1} \]
\[ G_i = g_i + g_i G_{i-1} + p_i \cdot G_{i-1} \]
\[ = g_i + (g_i + p_i) \cdot G_{i-1} \]

Ling's equation

\[ G_i = g_i + t_i \cdot G_{i-1} \]
\[ H_i = g_i + t_{i-1} \cdot G_{i-1} \]

Doran, Trans on Comp 9/88

---

**Ling Adder**

Conventional

\[ G_3 = g_3 + t_3 g_2 + t_3 t_2 g_1 + t_3 t_2 t_1 g_0 \]

Ling

\[ H_3 = g_3 + t_2 g_2 + t_2 t_1 g_1 + t_2 t_1 t_0 g_0 \]
\[ = g_3 + g_2 + t_2 g_1 + t_2 t_1 g_0 \]
HP Adder

\[ i_4 = p_3 p_2 p_1 p_0 \]

Naffziger, ISSCC'96

UC Berkeley EE241

B. Nikolic

HP Adder

 Carry ripple

 Sum select

 UC Berkeley EE241

B. Nikolic
Hybrid Adders

- 8-bit tapered pre-discharged Manchester carry chains, with $C_{in} = 0$ and $C_{in} = 1$
- 32-bit LSB carry-lookahead
- 32-bit MSB conditional sum adder
- Carry-select on most significant bits
- Latch-based timing
Binary Multiplication

\[
z = x \cdot y = \sum_{k=0}^{M \cdot N-1} z_k 2^k
\]

\[
- \left( \sum_{i=0}^{M-1} x_i 2^i \right) \left( \sum_{j=0}^{N-1} y_j 2^j \right)
\]

\[
- \sum_{i=0}^{M-1/N-1} \left( \sum_{j=0}^{i+j} x_i y_j 2^i j \right)
\]

with

\[
x = \sum_{i=0}^{M-1} x_i 2^i
\]

\[
y = \sum_{j=0}^{N-1} y_j 2^j
\]

---

Binary Multiplication

\[
\begin{array}{cccccc}
1 & 0 & 1 & 0 & 1 & 0 \\
\times & 1 & 0 & 0 & 1 & 1 \\
\hline
1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2N & \text{bits in the final sum}
\end{array}
\]

AND operation

Partial Products
Array Multiplier

Shift-and-Add Multiplier

- Standard adder and shift-in the multiplicand
- Shift the result as well and add
- N cycles
- Parallel adders add more hardware (adders) instead.
**MxN Array Multiplier**

---

**Critical Path**

---

Critical Path 1

Critical Path 2

Critical Path 1 & 2

---

\[ t_{\text{mult}} = (N-1) t_{\text{carry}} + (N-1) t_{\text{sum}} + (N-1) t_{\text{and}} \]

---

**Carry-Save Multiplier**

---

Vector Merging Adder

---

\[ t_{\text{mult}} = |N-1| t_{\text{carry}} + |N-1| t_{\text{and}} + t_{\text{merge}} \]
Adder Cells in Array Multiplier

Identical Delays for Carry and Sum

Multiplier Floorplan

X and Y signals are broadcasted through the complete array.

(—→ )
Wallace-Tree Multiplier

Wallace, Trans on Comp. 2/64
Wallace-Tree Multiplier

Tree Multipliers

- Time is proportional to log N
- Wiring is complicated
- Different wire lengths
- Optional pipelining