

BROWN

Computer System Design

Lecture 13: Computer Arithmetic

Prof. R. Iris Bahar
EN164
February 26, 2007

Reading: Appendix I, I.1, I.2, I.8

BROWN

Chained 1-bit adders are slow

- A 32-bit ALU is much slower than a 1-bit ALU.
 - Carry must ripple through all 32 bits to determine result

$$C_1 = b_0c_0 + a_0c_0 + a_0b_0$$

$$C_2 = b_1c_1 + a_1c_1 + a_1b_1$$

$$C_3 = b_2c_2 + a_2c_2 + a_2b_2$$

$$C_4 = b_3c_3 + a_3c_3 + a_3b_3$$

- How do you get rid of the ripple effect?**
 - Generate the "carries" a different way

EN164
Lecture 12-2

BROWN

Fast Carry Chain Design

- A carry-out signal is "1" when it is propagated to generated by a single bit cell:
 - When would we always generate a carry? $g_i = a_i b_i$
 - When would we propagate the carry? $p_i = a_i + b_i$
 - When would we never propagate a carry? $k_i = a_i' * b_i'$

$$C_1 = g_0 + p_0c_0$$

$$C_2 = g_1 + p_1c_1$$

$$C_3 = g_2 + p_2c_2$$

$$C_4 = g_3 + p_3c_3$$

$$C_2 = g_1 + p_1g_0 + p_1p_0c_0$$

$$C_3 = g_2 + p_2g_1 + p_2p_1g_0 + p_2p_1p_0c_0$$

$$C_4 = \dots$$

- Note that c_i only depends on a_i , b_i , and c_0
- But there is a practical limit to the number of inputs used to generate c_i .

EN164
Lecture 12-3

BROWN

Carry Lookahead Adder

$$c_n = g_{n-1} + p_{n-1}g_{n-2} + \dots + p_{n-1}p_{n-2} \dots p_1g_0 + p_{n-1}p_{n-2} \dots p_0c_0$$

- What is the practical limit on the number of inputs a gate can have?
 - High fanin gates can be built, but in standard CMOS they would be very slow
 - Fanin of 4 is the practical limit

EN164
Lecture 12-5

BROWN

Carry Skip Adder

- Combines carry-lookahead with ripple carry ideas
- Carries begin rippling simultaneously through each block.
 - Grey boxes are all computing $G_{i:k-3}, P_{i:k-3}$ term simultaneously
- Assume the carry-in to each block is initially 0
- If the carryout and P signals are both true, then the carry *skips* the next block and is ready to feed into the following block.
 - What if it's not true?

$$P_{0:3} = p_0 p_1 p_2 p_3$$

$$G_{0:3} = g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 g_0$$

$$C_4 = G_{0:3} + P_{0:3}c_3$$

EN164
Lecture 12-7

BROWN

4-bit Block Carry-Skip Adder

bits 12 to 15 bits 8 to 11 bits 4 to 7 bits 0 to 3

Setup Setup Setup Setup

Carry Propagation Carry Propagation Carry Propagation Carry Propagation

Sum Sum Sum Sum

$C_{i,0}$

Worst-case delay \rightarrow carry from bit 0 to bit 15 = carry generated in bit 0, ripples through bits 1, 2, and 3, skips the middle two groups (k is the group size in bits), ripples in the last group from bit 12 to bit 15. Thus, for $N=16$

$$T_{add} = t_{setup} + k t_{carry} + ((N/k) - 1) t_{skip} + k t_{carry} + t_{sum}$$

EN164
Lecture 12-8

Carry Select Adder

- **Another way to speedup addition:** Don't wait for carry signal to be ready before computing the sum.
- Instead, two additions are performed in parallel, one assuming the carry-in is 0 and the other assuming the carry-in is 1.
- When the carry-in is finally known, the correct sum is selected.

What's the path through the worst case delay?

34
-9

Carry Select Adder: Critical Path

$$T_{add} = t_{setup} + k t_{carry} + N/k t_{mux} + t_{sum}$$

EN164
Lecture 12-10

Binary Multiplication

	1 0 1 0 1 0	Multiplicand
x	1 0 1 1	Multiplier
	1 0 1 0 1 0	Partial products (can be formed in parallel)
	0 0 0 0 0 0	
+	1 0 1 0 1 0	
	1 1 1 0 0 1 1 1 0	Double Precision Result

EN164
Lecture 12-12

Unsigned Multiplication: Version I

EN164
Lecture 12-13

Unsigned Multiplication, Version II

What is the main advantage here?

EN164
Lecture 12-14

Shift & Add Multiplication

- **Right shift and add**
 - Partial product rows accumulated from top to bottom on an N-bit adder
 - Time for N bits $T_{serial_mult} = O(N T_{adder}) = O(N^2)$ for a RCA
- **Making it faster**
 - Use a faster adder
 - Use higher radix (e.g., base 4) multiplication
 - Use **multiplier recoding** to simplify multiple formation
 - Form partial product array in parallel and add it in parallel
- **Making it smaller**
 - Use an array multiplier
 - Very regular structure with only short wires to nearest neighbor cells. Thus, very simple and efficient layout in VLSI
 - Can be easily and efficiently pipelined

EN164
Lecture 12-15