

## Homework 2

Prof. Loh  
 CS3220 - Processor Design - Spring 2005  
 Handed Out: 01 Feb 2005  
 Due: 08 Feb 2005

### 1. Carry-Save Addition

In class we saw a carry-save adder that takes three inputs  $x, y, z$  and produces two outputs  $c, s$  such that  $x+y+z = c+s$ . Consider a generalized carry-save adder that takes inputs  $x_0, x_1, x_2, \dots, x_{n-1}$  and produces output  $y_0, y_1, \dots, y_{m-1}$  such that  $n > m$  and  $\sum_{i=0}^{n-1} x_i = \sum_{j=0}^{m-1} y_j$ .

- (a) Consider a 5-to-3 carry-save adder ( $n = 5, m = 3$ ), with inputs  $x_4, x_3, x_2, x_1, x_0$  and outputs  $y_2, y_1, y_0$ . Fill in the truth table for the  $y$ 's, given the 32 possible values of the  $x$ 's:

$x_4$	$x_3$	$x_2$	$x_1$	$x_0$	$y_2$	$y_1$	$y_0$
0	0	0	0	0	0	0	0
$\vdots$			$\vdots$		$\vdots$		

- (b) Draw a Wallace tree using 5-to-3 carry-save adders to add 32 numbers together. After the last 5-to-3 CSA, you will need to use a 3-to-2 CSA, followed by a final look-ahead carry adder (LCA/parallel-prefix adder) to complete the carry.
- (c) For  $N$  inputs each  $k$  bits in width, what is the asymptotic gate delay to add all  $N$  numbers together using 5-to-3 carry-save adders (plus the final 3-to-2 and LCA)?
- (d) For three outputs ( $m = 3$ ), what is the maximum number of inputs (largest value of  $n$ )?
- (e) Write the asymptotic delay to add  $N$  inputs each of  $k$  bits in width assuming  $X$ -to-3 carry save adders, where  $X$  is the value from problem 1d.
2. **Booth-Multiplication** In class, we saw Booth Multiplication algorithms for processing 1 or 2 bits per step. The algorithm can be extended to processes as many bits per step as you wish, although the logic gets more and more complicated. Consider a Radix-8 Booth Multiplier (processes 3 bits per step). Fill in the following table to determine the 3-bit Booth encoding, where  $M$  is the multiplier:

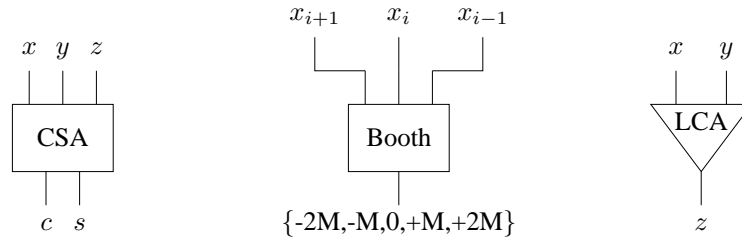
Current Bits			Previous Bit	Operation	Reason
$x_{i+2}$	$x_{i+1}$	$x_i$	$x_{i-1}$	—	—
0	0	0	0	Nothing	In middle of run of zeros
0	0	0	1	$+M$	Ended a run of ones
0	0	1	0	$+M$	Ended a run of zeros ( $-M$ ), Ended a run of ones ( $+2M$ )
$\vdots$	$\vdots$	$\vdots$	$\vdots$		

3. **Hybrid-Multiplier** With regular multiplication using an adder tree (e.g. a Wallace tree), the multiplication problem basically boils down to adding  $k$  numbers together (assuming inputs are  $k$ -bit values). A 2-bit Booth encoding allows

you to group two bits together and only have to perform one addition (or subtraction) for those two bits. Similarly, a 3-bit Booth encoding allows you to only perform one addition per three bits.

16-bit multiplication would normally require adding together 16 numbers. Show how to use a 2-bit Booth encoding to reduce a 16-bit multiplication to adding together only 8 numbers. Complete the multiplication by summing the 8 numbers with a Wallace tree (using 3-to-2 CSAs and a final LCA).

You may use regular logic gates, MUXes, and the following basic blocks:



To get you started:

