University of California, Berkeley
College of Engineering
Computer Science Division — EECS

Spring 2001 John Kubiatowicz

Homework Quiz (HW #3)
SOLUTIONS

February 20, 2001
CS152 Computer Architecture and Engineering

This quiz covers one of the problems from homework #3.
Good Luck!

Your Name:	
SID Number:	
Discussion Section:	

Total:	

Combinational Multipliers built with Carry Save Adders

In your homework, you learned about using Carry Save Adders to perform multiplication. Below we review using carry save adders to add four 3-bit numbers to produce a single 5-bit number:

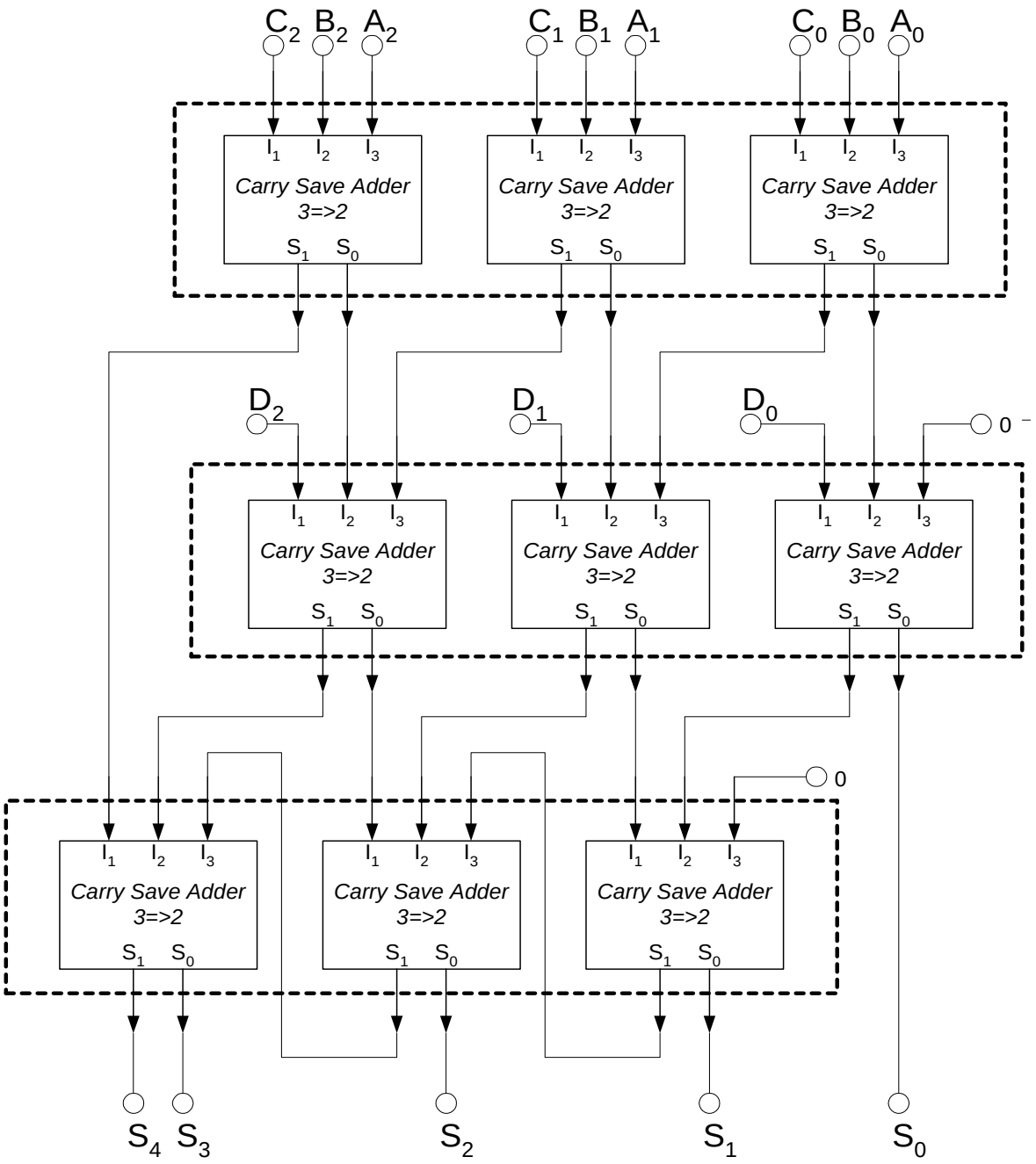

Figure #1: Adding four 3-bit numbers with carry-save adders.

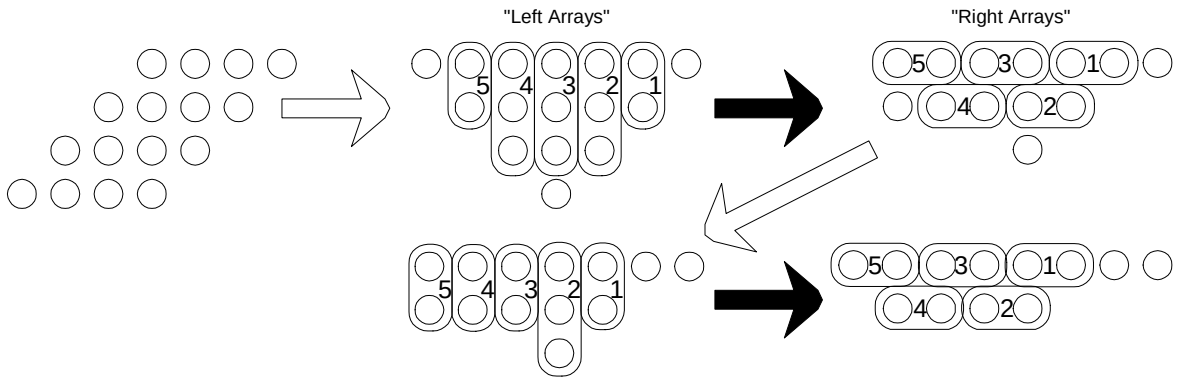

Figure #2: Carry Save Logic in a 4x4 multiplier

Figure #1 (previous page) shows a carry-save array for adding four 3-bit numbers together. You figured out how to use this type of addition to perform a 16x16 bit multiplier in your homework. One way to understand carry-save adders is that they add *columns* of bits together, just like people do on paper. Figure #2 shows this for a 4x4 multiplier. Each dot is a bit. The white arrows are renaming of bits, while the black arrows are carry-save logic.

The first array is the standard multiplication array. We rearrange the bits to get array 2. Each *pair* of arrays (called "left" and "right") represents one level of carry-save logic. The circled dots in a *column* of a left arrays are bits that are summed with a 3⇒2 adder. The circled dots in a *row* of a right array are the results. Thus, for instance, the two bits labeled "1" in array 3, are the result of adding the 2 bits labeled "1" in array 2. The long white arrow represents output of the first level going into inputs of the next. We continue until we have only two rows of bits, which will be added with a standard adder (possibly a carry-lookahead adder).

Representing each level by the maximum number of bits in any one column, we get a sequence that starts with the number of multiplier bits (4 in this case) and ends with the number 2:

$$4 \Rightarrow 3 \Rightarrow 2$$

1. Use a similar sequence to answer the following question: how many levels of carry-save logic would be needed for an 80x80 bit multiplier? The answer for 4x4 multiplier would be "2": Don't just write out a number. Show the actual sequence.

Answer: You must be careful to account for 2 and 3-element adders (which amounts to using the ceiling function where appropriate). Note that leftover single bits are not added at a given level.

$80 \Rightarrow 54 \Rightarrow 36 \Rightarrow 24 \Rightarrow 16 \Rightarrow 11 \Rightarrow 8 \Rightarrow 6 \Rightarrow 4 \Rightarrow 3 \Rightarrow 2$

Counting arrow (levels of logic), we get 10.

2. Use this number to *estimate* the maximum delay through this 80x80 multiplier, assuming that the last stage of addition uses a carry-lookahead adder. Assume that one level of logic (i.e. an AND or OR) requires 1 unit of delay. Assume that an adder requires 2 units of delay. Assume that you do the carry-lookahead in groups of 4 bits (exactly as in problem 4.45).

   Recall that the one-bit propagate and generate equations are: $p_i = a_i + b_i$ and $g_i = a_i \times b_i$. Also, the higher level propagate and generate signals are:

   $$P = p_3 \times p_2 \times p_1 \times p_0 \quad \text{and} \quad G = g_3 + (p_3 \times g_2) + (p_3 \times p_2 \times g_1) + (p_3 \times p_2 \times p_1 \times g_0)$$

   And the carry is:

   $$C_{out} = G + P \times C_{in}$$

   Don't assume gates wider than 5 bits.

   Hint: don't forget that you are doing a multiply.

   Express your result as a sum of numbers, with justifications for each of them. Try to make this as accurate as you can, but *remember that this is an estimate!*

   Note that the final adder is 2 x 80 = 160 bits. However, the critical path only includes the upper bits (since the most levels of logic start from the middle upward). To compute the critical path, consider 1 level (generate/propagate) + 2 levels (80⇒20) + 2 levels (20⇒5) + 2 levels(5⇒2) + 2 levels (final carry). So, we are talking 9 levels.

   *For the rest, 1 level for the AND gates (partial products) + 2 * 10 for the carry-lookahead logic.*

   Total = 9 + 1 + 20 = 30