EE486A Introduction to Computer Architecture
Lab 3
Carry-Lookahead Adder

CDR Charles B. Cameron

January 25, 2007

1 Introduction

The purpose of this lab is to design and simulate a carry-lookahead adder/subtracter using VHDL. This will require modifying your earlier design of a ripple-carry adder/subtracter. In addition to adding the circuitry necessary to make the circuit add and subtract more quickly, you will also add the circuitry needed to make it perform excess-m addition and subtraction as well.

2 Equipment

Altera Quartus II software

3 Description

First recall that the carry function is defined by

$$\begin{aligned} c_{\text{out}} &= xy + xc_{\text{in}} + yc_{\text{in}} \\ &= xy + c_{\text{in}}(x + y) \end{aligned} \tag{1}$$

and that

$$s = x \oplus y \oplus c_{\text{in}}. \tag{2}$$

Furthermore, the generate and propagate functions are defined by

$$G = xy \tag{3}$$

$$P = x + y. \tag{4}$$

With these expressions we can rewrite (1) as

$$c_{\text{out}} = G + c_{\text{in}}P. \tag{5}$$

Applying (5) for the first three stages of the adder we have

$$
\begin{aligned}
c_1 &= G_0 + c_0 P_0 && (6)\\
c_2 &= G_1 + c_1 P_1 \\
&= G_1 + (G_0 + c_0 P_0) P_1 \\
&= (G_1 + G_0 P_1) + c_0 (P_0 P_1) && (7)\\
c_3 &= G_2 + c_2 P_2 \\
&= G_2 + \big((G_1 + G_0 P_1) + c_0 (P_0 P_1)\big) P_2 \\
&= (G_2 + G_1 P_2 + G_0 P_1 P_2) + c_0 (P_0 P_1 P_2) && (8)
\end{aligned}
$$

Note that c_1, c_2, and c_3 all are expressed here as functions of c_0 and of several generate and propagate functions. Find similar expressions for:

- c_5, c_6, and c_7 as functions of c_4 and of associated generate and propagate functions;
- c_9, c_{10}, and c_{11} as functions of c_8 and of associated generate and propagate functions;
- c_{13}, c_{14}, and c_{15} as functions of c_{12} and of associated generate and propagate functions.

Now we can also express c_4:

$$
\begin{aligned}
c_4 &= G_3 + c_3 P_3 \\
&= G_3 + \big((G_2 + G_1 P_2 + G_0 P_1 P_2) + c_0 (P_0 P_1 P_2)\big) P_3 \\
&= (G_3 + G_2 P_3 + G_1 P_2 P_3 + G_0 P_1 P_2 P_3) + c_0 (P_0 P_1 P_2 P_3) \\
c_4 &= G_0^1 + c_0 P_0^1 && (9)
\end{aligned}
$$

where

$$G_0^1 = G_3 + G_2 P_3 + G_1 P_2 P_3 + G_0 P_1 P_2 P_3 \tag{10}$$

and

$$P_0^1 = P_0 P_1 P_2 P_3. \tag{11}$$

Similarly, we can define

$$c_8 = G_1^1 + c_4 P_1^1 \tag{12}$$

$$c_{12} = G_2^1 + c_8 P_2^1 \tag{13}$$

$$c_{16} = G_3^1 + c_{12} P_3^1 \tag{14}$$

where

$$G_1^1 = G_7 + G_6P_7 + G_5P_6P_7 + G_4P_5P_6P_7 \tag{15}$$
$$G_2^1 = G_{11} + G_{10}P_{11} + G_9P_{10}P_{11} + G_8P_9P_{10}P_{11} \tag{16}$$
$$G_3^1 = G_{15} + G_{14}P_{15} + G_{13}P_{14}P_{15} + G_{12}P_{13}P_{14}P_{15} \tag{17}$$

and

$$P_1^1 = P_4P_5P_6P_7. \tag{18}$$
$$P_2^1 = P_8P_9P_{10}P_{11}. \tag{19}$$
$$P_3^1 = P_{12}P_{13}P_{14}P_{15}. \tag{20}$$

Note the similarity in:

1. the forms of the G_i^1 of (10), (15), (16), and (17);
2. the forms of the P_i^1 of (11), (18), (19), and (20);
3. expressions you should derive (and include in your report) for c_1, c_5, and c_9; c_2, c_6, and c_{10}; and c_3, c_7, and c_{11}.

Because of all these similarities, you should implement a single VHDL module to compute three carries, one propagate function, and one generate function. Instantiate four of these modules to compute each of these signals for a 16-bit adder.

Furthermore, another instance of the same module can be used to compute c_4, c_8, and c_{12} along with another generate and another propagate function by further developing (9), (12), and (13):

$$c_4 = G_0^1 + c_0P_0^1 \tag{21}$$
$$\begin{aligned} c_8 &= G_1^1 + c_4P_1^1 \\ &= G_1^1 + (G_0^1 + c_0P_0^1)P_1^1 \\ &= (G_1^1 + G_0^1P_1^1) + c_0(P_0^1P_1^1) \end{aligned} \tag{22}$$
$$\begin{aligned} c_{12} &= G_2^1 + c_8P_2^1 \\ &= G_2^1 + (G_1^1 + G_0^1P_1^1 + c_0P_0^1P_1^1)P_2^1 \\ &= (G_2^1 + G_1^1P_2^1 + G_0^1P_1^1P_2^1) + c_0(P_0^1P_1^1P_2^1) \end{aligned} \tag{23}$$

Finally,

$$\begin{aligned} c_{16} &= G_3^1 + c_{12}P_3^1 \\ &= G_3^1 + (G_2^1 + G_1^1P_2^1 + G_0^1P_1^1P_2^1 + c_0P_0^1P_1^1P_2^1)P_3^1 \\ &= (G_3^1 + G_2^1P_3^1 + G_1^1P_2^1P_3^1 + G_0^1P_1^1P_2^1P_3^1) + c_0(P_0^1P_1^1P_2^1P_3^1) \\ &= G_0^2 + c_0P_0^2 \end{aligned} \tag{24}$$

where

$$G_0^2 = G_3^1 + G_2^1 P_3^1 + G_1^1 P_2^1 P_3^1 + G_0^1 P_1^1 P_2^1 P_3^1 \tag{25}$$

and

$$P_0^2 = P_0^1 P_1^1 P_2^1 P_3^1. \tag{26}$$

With these circuits you can implement a 16-bit adder/subtracter. If the input signal $\overline{\mathsf{Add}}$/Subtract is low, the circuit should perform addition; otherwise it should perform subtraction.

An additional feature that did not enter into your design of the ripple-carry adder is the ability to do excess-m addition and subtraction as we discussed in class. Add another input signal to your design. If the input signal $\overline{\mathsf{TwosComplement}}$/Excess is low, the circuit should perform operations on two's-complement values; otherwise it should perform them on excess-m operations. Since you are designing a 16-bit adder/subtracter, this is an excess-$2^{16-1} - 1$ or excess-32767 adder/subtracter.

When you have fully simulated and debugged your design, take timing simulations on it and on a 16-bit adder/subtracter of the ripple-carry design you designed previously. Use these timing measurements to compute the speed-up provided by the carry-lookahead adder/subtracter circuit you have now designed.

4 Lab Report

Your lab report must contain all of your fully commented VHDL source files along with simulations convincingly showing the correct operation of the circuit for all four combinations of two's-complement and excess-32767 arguments, both for addition and subtraction. It should detail the timing simulations and the calculation of the speed-up.

The report itself should also provide commentary on the VHDL code and the simulations so that they all can be easily understood.