Supplement to

Logic and Computer Design Fundamentals
4th Edition[1]

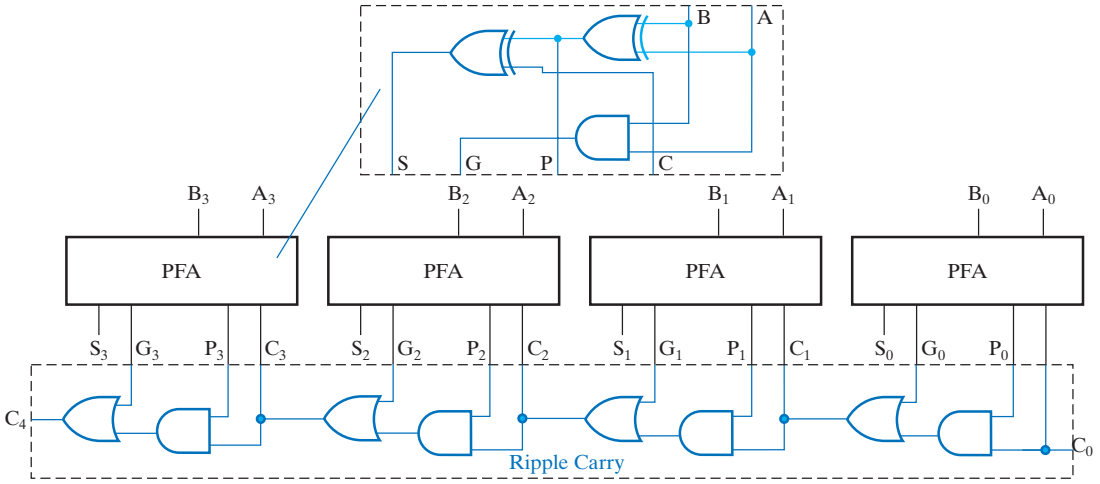
Carry-Lookahead Adders

Selected topics not covered in the fourth edition of *Logic and Computer Design Fundamentals* are provided here for optional coverage and for self-study. This material fits well with the desired coverage in some programs but not may not fit within others due to time constraints or local preferences. This supplement based on material from the third edition of *Logic and Computer Design Fundamentals* and is intended to be used at the end of section 4-2 of the fourth edition. It introduces the carry lookahead adder concept and gives a rough measure for predicting the performance improvement achievable by use of carry lookahead techniques. This material is appropriate for students in programs not covering carry lookahead in subsequent computer architecture courses.

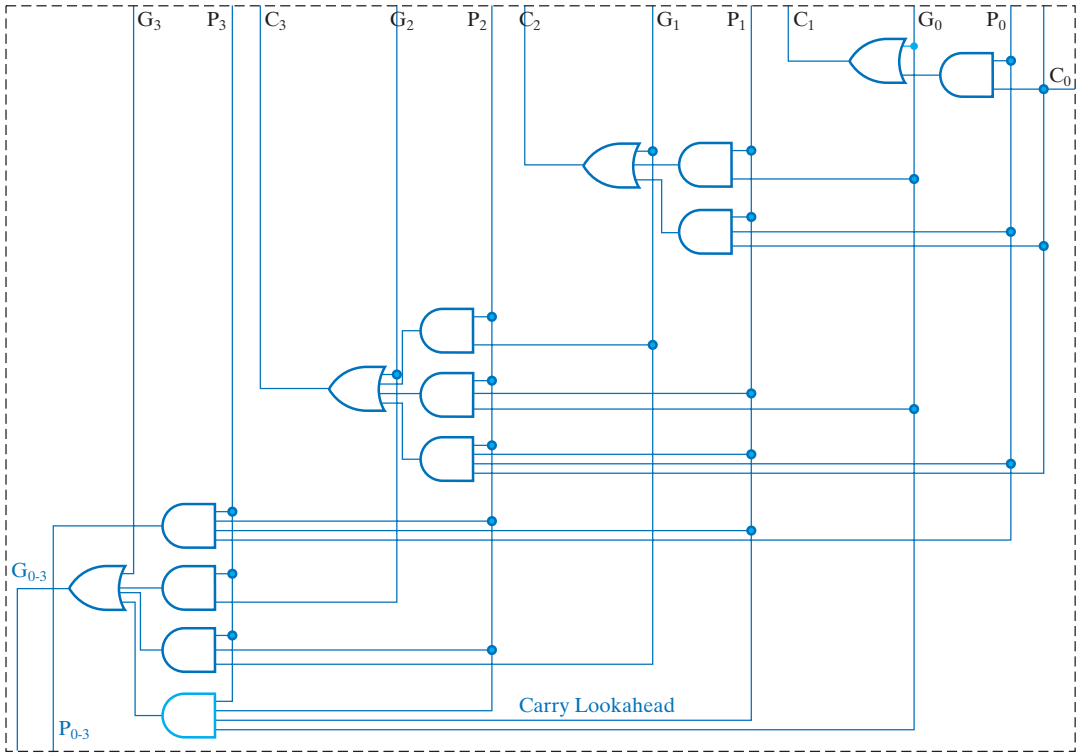
Carry Lookahead Adder

The ripple carry adder, although simple in concept, has a long circuit delay due to the many gates in the carry path from the least significant bit to the most significant bit. For a typical design, the longest delay path through an n-bit ripple carry adder is approximately $2n + 2$ gate delays. Thus, for a 16-bit ripple carry adder, the delay is 34 gate delays. This delay tends to be one of the largest in a typical computer design. Accordingly, we find an alternative design, the *carry lookahead adder*, attractive. This adder is a practical design with reduced delay at the price of more complex hardware. The carry lookahead design can be obtained by a transformation of the ripple carry design into a design in which the carry logic over fixed groups of bits of the adder is reduced to two-level logic. The transformation is shown for a 4-bit adder group in Figure 1.

First, we construct a new logic hierarchy, separating the parts of the full adders not involving the carry propagation path from those containing the path.



(a)



(b)

FIGURE 5-1
Development of a Carry Lookahead Adder

We call the first part of each full adder a *partial full adder* (PFA). This separation is shown 1(a), which presents a diagram of a PFA and a diagram of four PFAs connected to the carry path. We have removed the OR gate and one of the AND gates from each of the full adders to form the ripple carry path.

There are two outputs, P_i and G_i, from each PFA to the ripple carry path and one input C_i, the carry input, from the carry path to each PFA. The function $P_i = A_i \oplus B_i$ is called the *propagate* function. Whenever P_i is equal to 1, an incoming carry is propagated through the bit position from C_i to C_{i11}. For P_i equal to 0, carry propagation through the bit position is blocked. The function $G_i = A_i \cdot B_i$ and is called the *generate* function. Whenever G_i is equal to 1, the carry output from the position is 1, regardless of the value of P_i, so a carry has been generated in the position. When G_i is 0, a carry is not generated, so that C_{i11} is 0 if the carry propagated through the position from C_i is also 0. The generate and propagate functions correspond exactly to the half adder and are essential in controlling the values in the ripple carry path. Also, as in the full adder, the PFA generates the sum function by the exclusive-OR of the incoming carry C_i and the propagate function P_i.

The carry path remaining in the 4-bit ripple carry adder has a total of eight gates in cascade, so the circuit has a delay of eight gate delays. Since only AND and OR gates are involved in the carry path, ideally, the delay for each of the four carry signals produced, C_1 through C_4, would be just two gate delays. The basic carry lookahead circuit is simply a circuit in which functions C_1 through C_3 have a delay of only two gate delays. The implementation of C_4 is more complicated in order to allow the 4-bit carry lookahead adder to be extended to multiples of 4 bits, such as 16 bits. The 4-bit carry lookahead circuit is shown in Figure 1(b). It is designed to directly replace the ripple carry path in Figure 1(a). Since the logic generating C_1 is already two-level, it remains unchanged. The logic for C_2, however, has four levels. So to find the carry lookahead logic for C_2, we must reduce the logic to two levels. The equation for C_2 is found from Figure 1(a), and the distributive law is applied to obtain

$$\begin{aligned} C_2 &= G_1 + P_1(G_0 + P_0C_0) \\ &= G_1 + P_1G_0 + P_1P_0C_0 \end{aligned}$$

This equation is implemented by the logic with output C_2 in Figure 5-6(b). We obtain the two-level logic for C_3 by finding its equation from the carry path in Figure 1(a) and applying the distributive law:

$$\begin{aligned} C_3 &= G_2 + P_2(G_1 + P_1(G_0 + P_0C_0)) \\ &= G_2 + P_2(G_1 + P_1G_0 + P_0C_0) \\ &= G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_0 \end{aligned}$$

The two-level logic with output C_3 in Figure 1(b) implements this function.

We could implement C_4 using the same method. But some of the gates would have a fan-in of five, which may increase the delay. Also, we are interested in reusing this same circuit for higher numbered bits (e.g., 4 through 7, 8 through 11, and 12 through 15 of a 16-bit adder). For this adder, in positions 4, 8, and 12 we would like the carry to be produced as fast as possible without using excessive fan-in. Accordingly, we want to repeat the same carry lookahead trick for *4-bit groups* that we used to handle the 4 bits. This will allow us to reuse the carry lookahead circuit design for each group of 4 bits, and also to use the same circuit for four 4-bit groups as if they were individual bits. So instead of generating C_4, we produce generate and propagate functions that apply to 4-bit groups instead of a single bit to act as the inputs for the group carry lookahead circuit. To propagate a carry from C_0 to C_4, we need to have all four of the propagate functions equal to 1, giving the *group propagate* function

$$P_{0-3} = P_3P_2P_1P_0$$

To represent the generation of a carry in positions 0, 1, 2, and 3, and its propagation to C_4, we need to consider the generation of a carry in each of the positions, as represented by G_0 through G_3, and the propagation of each of these four generated carries to position 4. This gives the *group generate* function

$$G_{0-3} = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0$$

The group propagate and group generate equations are implemented by the logic in the lower part of Figure 1(b). If there are only four bits in the adder, then the logic circuit used for C_1 can be used to generate C_4 from these two outputs; we will later refer to the C_1 logic block as OC (Output Carry) for generating the output carry from an adder, in this case, C_4.

In a longer adder, for $4n$ bits, where n = 2, 3, and so on, one or more carry lookahead circuits identical to that in the figure, except for labeling, are placed at the second level to generate C_4, C_8, C_{12} and so on. As the number of adder bits crosses values equal to 4^m for m = 2, 3, and so on, additional carry-lookahead circuit (CLC) levels are added.

Assuming that an exclusive OR contributes 2 gate delays, the longest delay in the 4-bit carry lookahead adder is 6 gate delays, compared with 10 gate delays in the ripple carry adder. The improvement is very modest and perhaps not worth all the extra logic.

Two examples of larger carry-lookahead adders (CLAs) are shown in Figure 2. In Figure 2(a), a 16-bit CLA is shown. This adder requires two CLC levels and five carry-lookahead circuits (CLCs). Note how the second-level CLC uses the group P and G outputs from the four first level CLCs as inputs and provides the carry outputs C_4, C_8, and C_{12}. Also, the P and G group outputs from the second-level CLC cover carry generation and propagation for all 16 bits and, by using an OC circuit, can combine these two outputs with C_0 to produce carry C_{16}.r.

In Figure 2(a), a blue (or gray) path represents one of many longest delay paths through the circuit. Assuming that each passage through an FPA or CLC block requires two gate delays, the delay of this circuit can be estimated as 5 x 2 =

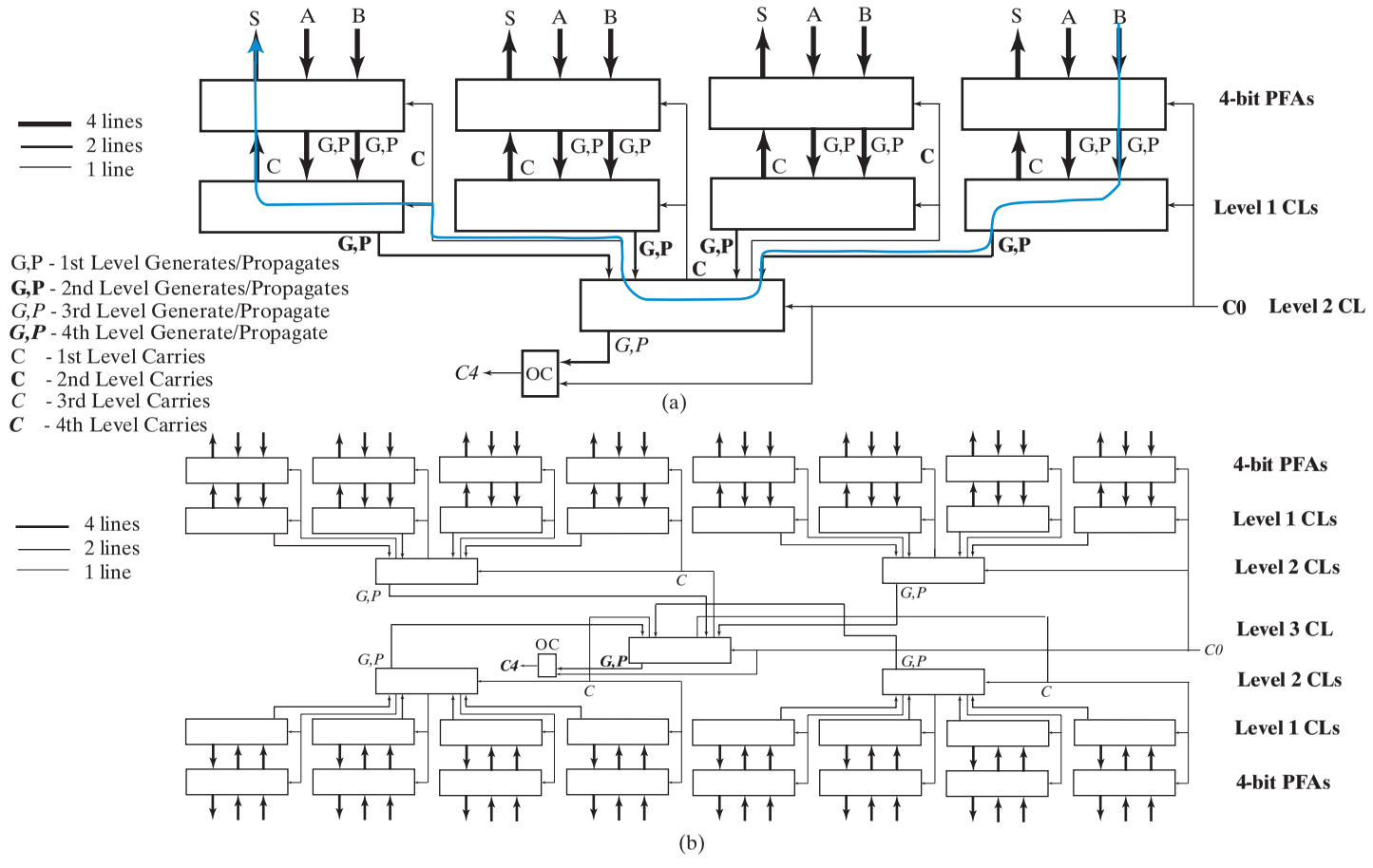


FIGURE 5-2
Carry Lookahead Adders: (a) 16-bit and (b) 64 b-bit

10 gate delays compared to 2 x 16 + 2 = 34 gate delays for the 16-bit ripple carry adder. The performance improves by a factor greater than three.

Based on the results for the 4-bit and 16-bit CLAs, we now attempt to deduce a formula for the maximum delay of an a CLA using L CLC levels. For the 4-bit CLA, there are 6 gate delays for L = 1 and, for the 16-bit CLA there are 10 gate delays for L = 2. In the 4-bit CLA, there is one pass through a CLC and, and in the 16-bit CLA, there are three passes through CLCs. Based on this information, we can conclude that the number of passes through CLCs = $2L - 1$ for L levels. Based on the numerical values of 6 and 10 gate delays just presented, the Estimated Maximum Path Delay is

$$2(2L - 1) + 4 = 4L - 2 + 4 = 4L + 2 \text{ Gate Delays}$$

In Figure 2(b), a 64-bit CLA with 21 CLCs is shown. This adder was designed by adding a single third-level CLC and one OC circuit to four 16-bit CLAs minus their respective OC circuits. Note that the third-level CLC uses the group P and G outputs from the four second-level CLCs as inputs and provides the carry outputs C_{16}, C_{32}, and C_{48}. Also, the P and G group outputs from the third-level CLC circuit cover carry generation and propagation for all 64 bits and, by using an OC circuit, can combine these two outputs with C_0 to produce carry output C_{64}. The delay analysis of this circuit is treated in Problem 2.

References

1. MANO, M. M. AND C. R. KIME. *Logic and Computer Design Fundamentals, 4th* ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2008.
2. MANO, M. M. AND C. R. KIME. *Logic and Computer Design Fundamentals*, 3rd ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2004.

Problems

1. Suppose that a 16-bit adder is to be designed using a mixture of carry-lookahead and ripple carry. This design requires that: (1) The carry output C_4 is required instead of $G_{0\text{-}3}$ and $P_{0\text{-}3}$ from the carry lookahead logic, and (2) a ripple carry to be used between the four 1st-level carry-lookahead circuits in place of the 2nd-level lookahead circuit.

(a) Find the logic diagram for the 2-level implementation of output C_4 for a modified lookahead circuit. What is the maximum gate fan-in required?

(b) Compare the length of a maximum path in terms of number of gate delays for this 16-bit adder compared to that for the 16-bit adder with 2nd-level lookahead in Figure 2(a).

2. **(a)** Draw a maximum delay path through the 64-bit adder in Figure 2(b) and determine the number of gate delays assuming that each block of logic passed through requires two gate delays. **Hint:** A maximum delay path goes from the upper right to the lower left of the diagram passing through all levels of the circuit once.

(b) Compare your result in part a to that obtained from the maximum delay path formula for L = 3 levels. Do the results agree?

(c) Compare your result in part a to the maximum path delay for a 64-bit ripple-carry adder. What is the delay improvement factor?

(d) Find the estimated maximum delay through the 16-bit and the 64-bit carry-lookahead adders in Figure 2 assuming that each gate delay is 0.20 ns.

3. **(a)** What is the number of carry-lookahead levels L in a 128-bit carry-lookahead adder using the same component types are in Figure 2?

(b) How many carry-lookahead circuits are required?

(c) What is the estimated maximum delay through the 128-bit adder assuming that each gate delay is 0.20 ns?