3.3 ARITHMETIC PIPELINE

Some functions of the arithmetic logic unit of a processor can be pipelined to maximize performance. An arithmetic pipeline is used for implementing complex arithmetic functions like floating-point addition, multiplication, and division. These functions can be decomposed into consecutive subfunctions. For example Figure 3.13 presents a pipeline architecture for floating-point addition of two numbers. (A nonpipelined architecture of such an adder is described in Chapter 2.) The floating-point addition can be divided into three stages: mantissas alignment, mantissas addition, and result normalization [MAN 82, HAY 78].

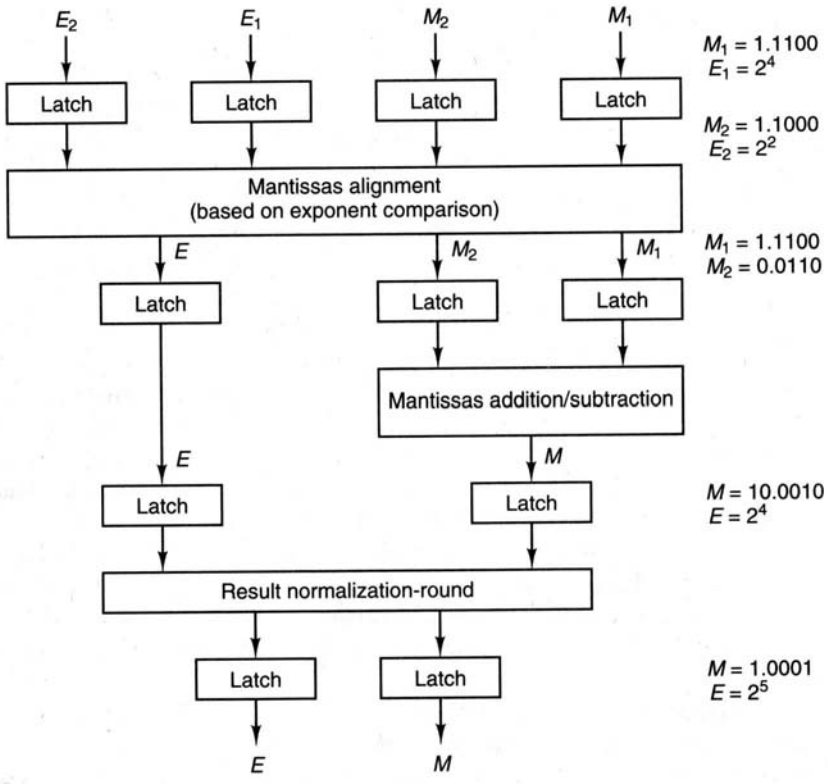

Figure 3.13 A pipelined floating-point adder.

In the first stage, the mantissas M_1 and M_2 are aligned based on the difference in the exponents E_1 and E_2. If $| E_1 - E_2 | = k > 0$, then the mantissa with the smaller exponent is right shifted by k digit positions. In the second stage, the mantissas are added (or subtracted). In the third stage, the result is normalized so that the final mantissa has a nonzero digit after the fraction point. When necessary, this normalized adjustment is done by shifting the result mantissa and the exponent.

Another example of an arithmetic pipeline is shown in Figure 3.14. This figure presents a pipelined architecture for multiplying two unsigned 4-bit numbers using carry save adders. The first stage generates the partial products M_1, M_2, M_3, and M_4. Figure 3.14 represents how M_1 is generated; the rest of partial products can be generated in the same way. The M_1, M_2, M_3, and M_4, are added together through the two stages of carry save adders and the final stage of carry lookahead adder. (A nonpipelined architecture of such a multiplier is described in Chapter 2.)

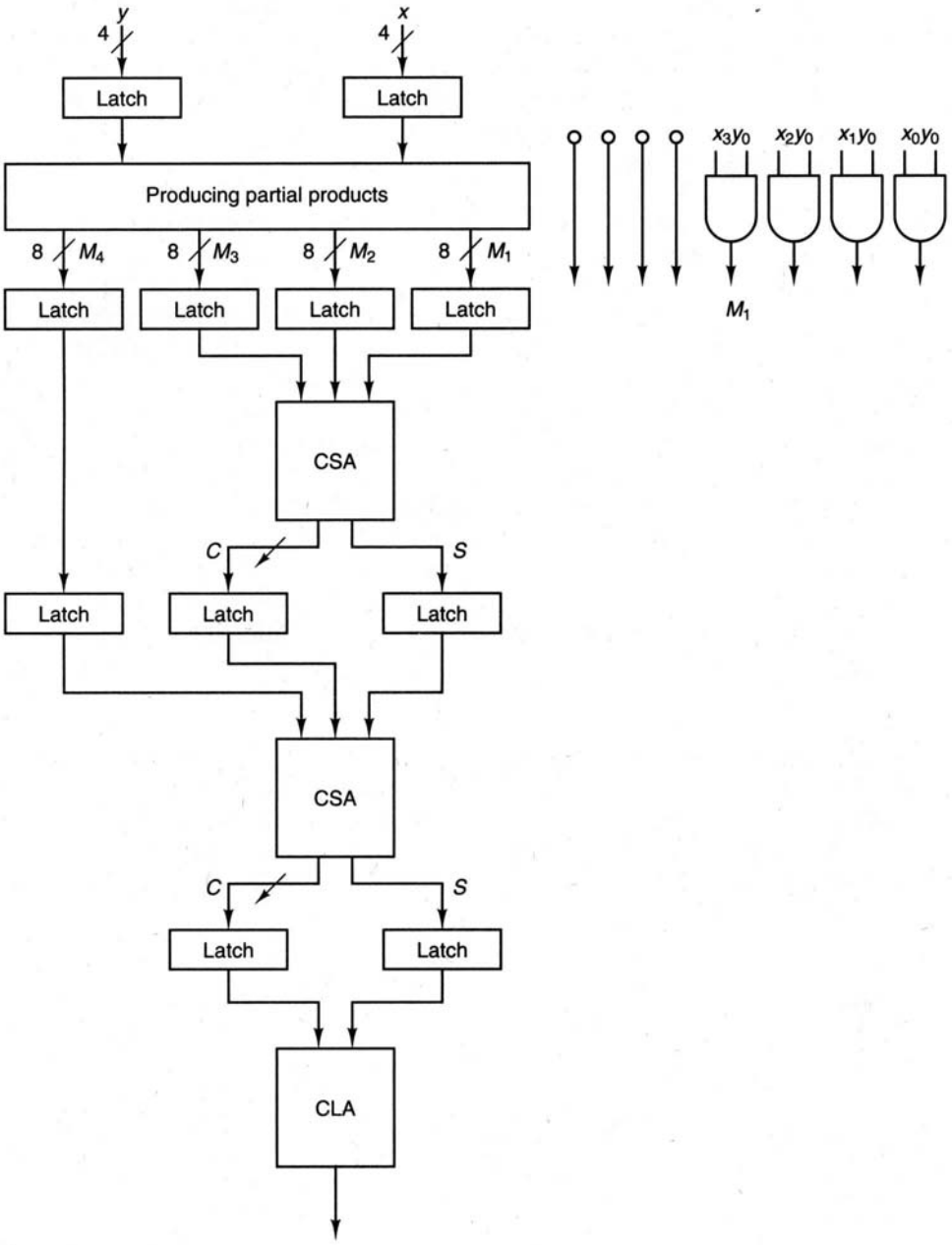

Figure 3.14 A pipelined carry save multiplier.