Single-Precision Floating Point Matrix Multiplier Using Low-Power Arithmetic Circuits

Soumya Gargave, Yash Agrawal and Rutu Parekh

Abstract This paper presents a single-precision floating point (IEEE 754 standard) matrix multiplier module. This is constructed using subblocks, which include floating point adder and floating point multiplier. These subblocks are designed to achieve the goal of low power consumption. Different architectures of subblocks are compared on the basis of energy-delay product. Design and simulations have been performed for 180 and 45 nm technology node. Simulation results show that design of floating point matrix multiplier is better at 45 nm than 180 nm technology node in terms of lesser delay by 43% and energy-delay product by 97.86% at 1 V. Also, 45 nm technology cells occupy only 6.25% of the area as compared to 180 nm cells.

Keywords Arithmetic and logic circuits · Energy-delay product (EDP)
Floating point multiplication · Low-voltage low-power design
Matrix multiplier · Simulation

1 Introduction

Technology advances brought together large number of devices on a small silicon area, thereby increasing power density on a chip. Low-power designs of these devices are therefore needed to prolong lifetime of a chip as well as the battery. One way to reduce power of a bigger module is to reduce power dissipation at subblocks

S. Gargave · Y. Agrawal · R. Parekh (✉)
VLSI & Embedded Systems Group, Dhirubhai Ambani Institute
of Information and Communication Technology,
Ganghinagar 382007, Gujarat, India
e-mail: rutu_parekh@daiict.ac.in

S. Gargave
e-mail: soumyagargave@gmail.com

Y. Agrawal
e-mail: yash_agrawal@daiict.ac.in

A. Garg et al. (eds.), *Advances in Power Systems and Energy Management*, Lecture Notes in Electrical Engineering 436,
https://doi.org/10.1007/978-981-10-4394-9_67

level. This requires implementation of judicial design strategies and architectures to obtain the best result for any application. Matrix multiplication unit is used in digital signal processing applications such as, digital imaging, signal processing, computer graphics, and multimedia. So, it is very crucial that it occupies less area, works fast, and consumes low power. IEEE standard 754 floating point (FP) is a representation for real numbers on computers. The goal of this paper is to present a low-power design for FP matrix multiplier. Paper publications in this area have been limited to building algorithm [1, 2] and subblocks required for floating point matrix multipliers, which include floating point multiplier [3] and floating point adder [4]. The circuit level simulation of designs at 180 and 45 nm technology nodes is performed using low-voltage and low-power design techniques in Virtuoso-Cadence environment.

The rest of the paper is organized as follows. Section 2 presents architecture of FP matrix multiplier along with its subblocks like multiplexer, FP multiplier, FP adder, and register. Section 3 presents the simulation results and comparison of delay, power, and energy-delay product (EDP) at 180 and 45 nm node for low-power design. Finally, conclusion is drawn in Sect. 4.

2 Matrix Multiplier Implementation

FP matrix multiplication is performed basically in two steps, FP multiplication and FP addition. In this paper, power is reduced by reusing the adder architecture to perform addition one after the other [1, 2]. This is an efficient technique to reduce power consumption by breaking the main circuit in partially sequential and partially parallel parts. Matrix multiplication operation is performed as follows:

$$\begin{bmatrix} X_{11} & X_{12} & X_{13} & X_{14} \\ X_{21} & X_{22} & X_{23} & X_{24} \\ X_{31} & X_{32} & X_{33} & X_{34} \\ X_{41} & X_{42} & X_{43} & X_{44} \end{bmatrix} \times \begin{bmatrix} Y_{11} & Y_{12} & Y_{13} & Y_{14} \\ Y_{21} & Y_{22} & Y_{23} & Y_{24} \\ Y_{31} & Y_{32} & Y_{33} & Y_{34} \\ Y_{41} & Y_{42} & Y_{43} & Y_{44} \end{bmatrix} = \begin{bmatrix} Z_{11} & Z_{12} & Z_{13} & Z_{14} \\ Z_{21} & Z_{22} & Z_{23} & Z_{24} \\ Z_{31} & Z_{32} & Z_{33} & Z_{34} \\ Z_{41} & Z_{42} & Z_{43} & Z_{44} \end{bmatrix}, \tag{1}$$

where

$$Z_{11} = X_{11}Y_{11} + X_{12}Y_{21} + X_{13}Y_{31} + X_{14}Y_{41}, \tag{2}$$

$$Z_{12} = X_{11}Y_{12} + X_{12}Y_{22} + X_{13}Y_{32} + X_{14}Y_{42}, \tag{3}$$

and so on…

The block diagram of FP matrix multiplier along with its subblocks is shown in Fig. 1a–e. Figure 1a depicts the 4 × 4 matrix multiplier. Figure 1b shows operation of each cell of the matrix multiplier. It comprises of 4:1 MUX, FP multiplier, FP adder, and memory register. The circuit diagram of 4:1 MUX is presented in

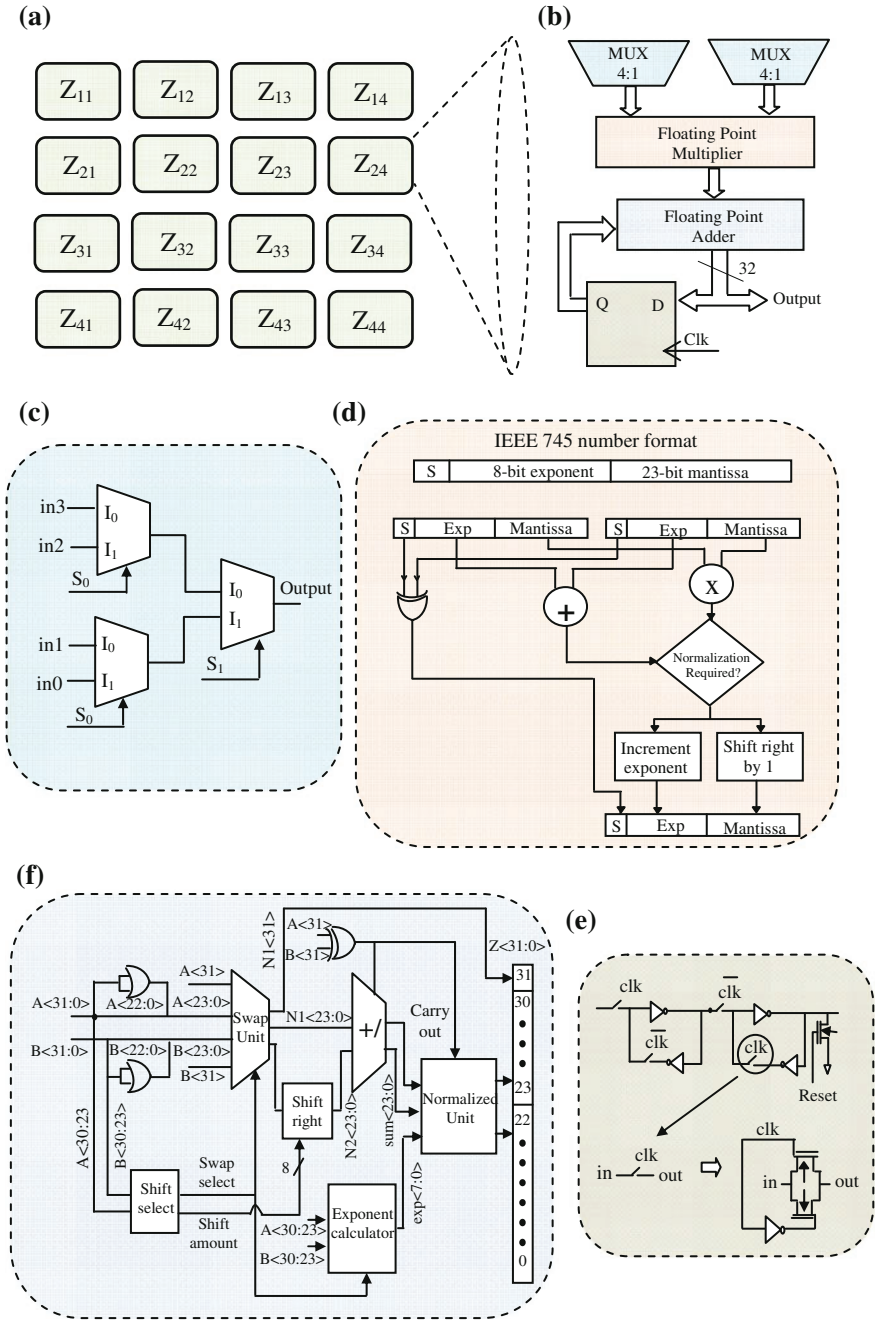

Fig. 1 Representation of FP matrix multiplier and its subblocks. **a** Architecture of 4×4 FP matrix multiplier. **b** FP matrix multiplier block diagram. **c** MUX schematic. **d** FP multiplier block diagram. **e** FP adder block diagram. **f** 1-bit register circuit

Fig. 1c. MUX selects the inputs of the two matrices to be multiplied. These are 4:1 MUX for matrices of size 4×4. Each MUX gives output as a 32-bit single-precision floating point number. These are inputs to the floating point multiplier unit.

The FP multiplier unit has three subparts as shown in Fig. 1d. Floating point multiplication [3] is done in three steps: sign calculation, exponent calculation, and mantissa calculation. The multiplier unit used for mantissa calculation is the slowest path as it has to perform operation on 24 bits (23 bits mantissa and 1 bit normalizing bit). So, in order to make this subblock fast, different architectures of multipliers have been studied. FP multiplier can be implemented using several designs as divide and conquer multiplier [5], Wallace tree multiplier [6], and Barun multiplier [5]. Divide and conquer multiplier is used mostly in FPGAs, as this leads to fast calculation. However, this comes at the expense of more hardware. Wallace tree multiplier has less number of bits in each level, but delay is not evenly distributed. So, it may cause glitches. Also, the tree structure uses a lot of interconnection; therefore uses a lot more area. Barun multiplier shown in Fig. 2 offers evenly distributed delay for the inputs of a full adder and it can be extended to multiply higher number of bits. Based on the simulation results presented in Sect. 3 for all the above three multipliers, it has been inferred that Barun multiplier offers lesser EDP. Consequently, this is incorporated in the present low-power FP matrix multiplier design.

The basic block in FP matrix multiplier is a full adder. Different architectures of full adder like 17-T, 14-T, 10-T have been discussed in [7]. The 17-T adder proves to be the best architecture in terms of power, delay, and power-delay product. A 17-T adder was made using pass transistor logic, but there is a probability of getting wrong logic at the output. For example, say when $a = b = 0$ the output at

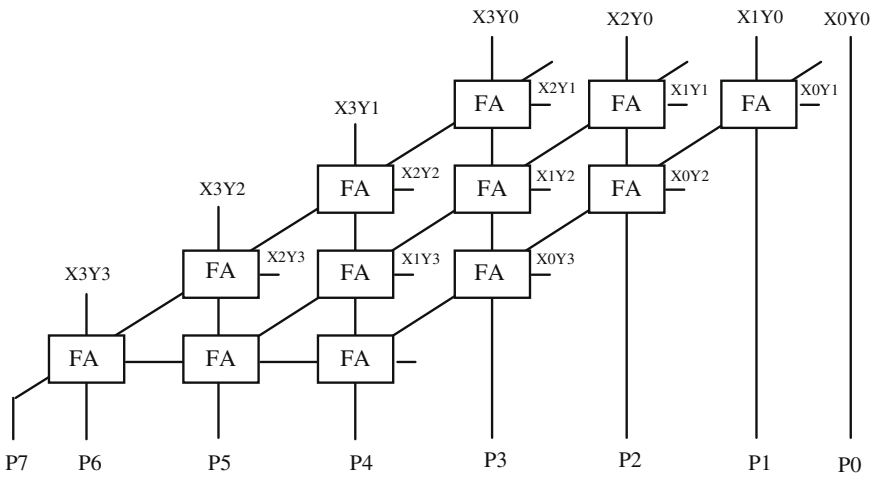

Fig. 2 Barun multiplier

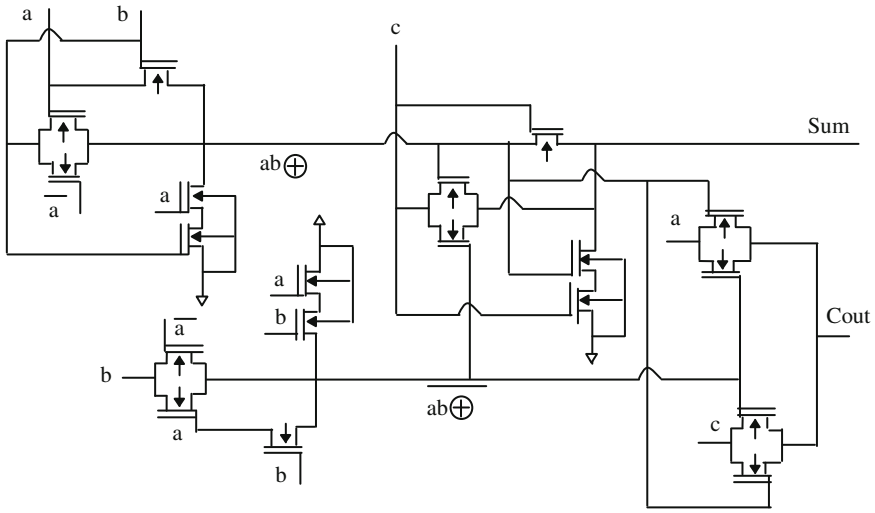

Fig. 3 Proposed full adder

$a \oplus b$ is V_{tp} and that at Ex-NOR is $V_{dd} - V_{tn}$. This problem increases when adders are cascaded as inputs a and b can be output of any previous adder. So, modified 17T architecture is proposed here and used in this work. This is shown in Fig. 3. In this, transmission gate is used instead of pass transistor which transfers full logic.

The addition of two FP numbers is the trickiest part. The algorithm for FP addition is taken from [4] and block diagram for the same is shown in Fig. 1e. FP adder performs addition of partial products, as shown in Eqs. (2) and (3). First, there is a need to make exponent of both the operands equal in order to perform addition. According to Fig. 1e, if $|B_{exp}| > |A_{exp}|$ ($B_{exp} = B\langle 30:23\rangle$ and $A_{exp} = A\langle 30:23\rangle$), then swapping of the numbers is required, which implies that the signal swap select = 1. The mantissa of smaller number (A) is shifted right by the shift amount to make exponent of both equal and then added to or subtracted from other number, as per the signs of the two numbers. Normalizing unit normalizes the output to IEEE 754 format again.

A 32-bit register stores the value of the adder block, such that it can be added again to the new multiplied value. Figure 1f shows the circuit diagram of 1-bit register.

Power of the FP matrix multiplier in the present design is reduced by,

(1) Using multiplier module with sleep transistor logic.
(2) Reducing the bit width for the floating point multiplier input from 24 to 12 bits [8]. This method is called *bit truncation*.

The inputs to the multiplier in Fig. 1d are of 24 bits, so after multiplication output is of 48 bits. But as this output needs to be fitted in FP format again, rounding has to be done. This requires more power. Consequently to reduce power,

bit truncation is used in which inputs are reduced to 12 bits, giving output of 24 bits. This can be effectively used in most of the applications like sensors, image processing, etc., where limited accuracy can be tolerated.

3 Simulation Results

Simulations are performed on circuits made using UMC 180 nm and gpdk 45 nm technology node library in Cadence Virtuoso, so as to compare low-power designs of both technologies in terms of EDP. Figure 4a shows the comparison of EDP for different multiplier architectures. It is evident from the figure that Braun multiplier is best in terms of lower EDP and minimum energy-delay point is observed at 1 V. So, this multiplier is built again using 45 nm technology cells and compared under same conditions with 180 nm circuit. It is clear from Fig. 4b that the best case (1 V) value of 45 nm circuit is lower than the best case (1 V) value of 180 nm circuit.

The functionality of FP multiplier and adder units have been analyzed. It has been investigated that FP multiplier discussed here has an improvement of 61.9% in EDP with respect to [9] (904.2E−21 J) and FP adder has an improvement of 94.8% in power with respect to [10] (5.6 mW). For logic and functionality verification, single element of the FP matrix multiplier is implemented and presented below. Using Eq. (2), the first element of the output matrix is computed as:

$$Z_{11} = X_{11}Y_{11} + X_{12}Y_{21} + X_{13}Y_{31} + X_{14}Y_{41},$$

Let,

$$X_{11} = -16.23998,\ X_{12} = -12.55799,\ X_{13} = 1.26839,\ X_{14} = 1.26839,$$
$$Y_{11} = -12.2398,\ Y_{21} = 0.01246,\ Y_{31} = 124.8939,\ \text{and}\ Y_{41} = 0.01246.$$

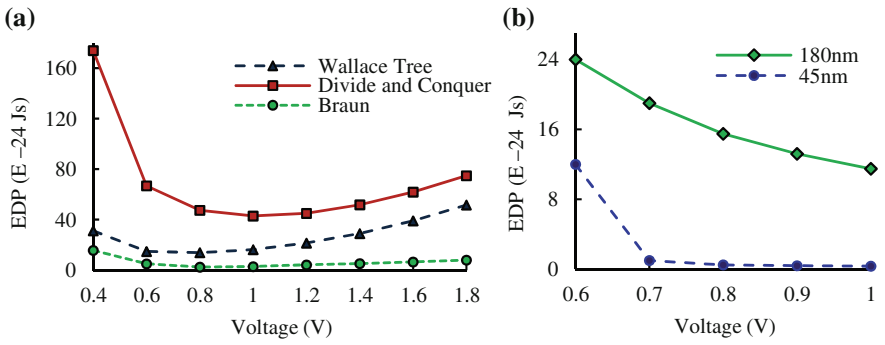

Fig. 4 **a** EDP of different multipliers at 180 nm technology node. **b** EDP comparison of Barun multiplier at 180 and 45 nm technology nodes

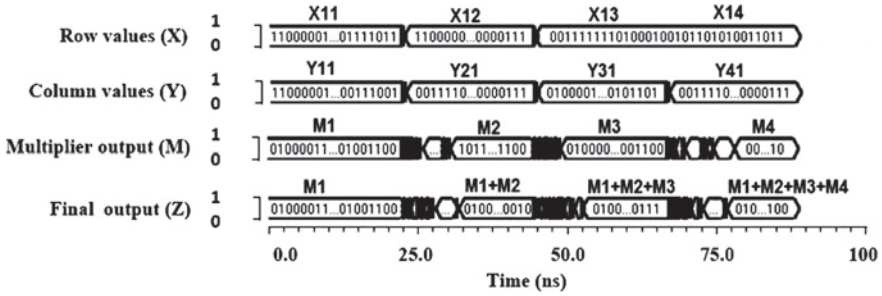

Fig. 5 Simulation table for floating point matrix multiplier

Here X_{11}, X_{12}, X_{13}, and X_{14} are the inputs to MUX1, which is first element in Fig. 1b. These comprise row vector. While Y_{11}, Y_{21}, Y_{31}, and Y_{41} are the inputs to MUX2, these comprise column vector. $M1$ ($X_{11} \times Y_{11}$), $M2$ ($X_{12} \times Y_{21}$), $M3$ ($X_{13} \times Y_{31}$), and $M4$ ($X_{14} \times Y_{41}$) are corresponding products of the row and column values. Z is the final output, which is $M1 + M2 + M3 + M4$. The output from the simulator ADE-XL of virtuoso is shown in Fig. 5. Table 1 shows the simulated result of a single element Z (equals to $M1 + M2 + M3 + M4$) for floating point matrix multiplier. It is analyzed that the error between desired and simulated output is 0.044% which is because of rounding the product of multiplication to 24 bits as in FP multiplication algorithm. Table 2 presents power, delay, and EDP obtained from the simulation results of FP matrix multiplier at 180 nm for normal voltage of 1.8 V and low-power design at 1 V. The same FP matrix multiplier circuit is implemented for 45 nm technology node. Table 3 compares the results for 180 and 45 nm technology at 1 V.

Table 1 Output of first element of floating point matrix multiplier (180 nm)

Quantity	Decimal value	32-bit floating point binary value
Desired output	357.0476	0 10000111 01100101000011000011000
Simulated output	356.89	0 10000111 01100100111000111101100

Table 2 Simulation results of floating point matrix multiplier at 180 nm

Parameters	At 1.8 V	At 1 V
Total power (mW)	5.006	0.805
Static power (μW)	145.2	48.09
Dynamic power (mW)	4.8608	0.757
Delay (ns)	55.6478	91.8
EDP (Js)	15.5E−18	6.783E−18

Table 3 Comparison of logic circuits at 180 and 45 nm at supply of 1 V

Logic circuit	Technology (nm)	Delay (ns)	Static power (μW)	Dynamic power (μW)	Total power (μW)	EDP (Js)
Braun multiplier	180	2.4249	0.798	1.286	2.084	12.25E−24
	45	0.7651	0.001	0.752	0.0762	0.045E−24
FP multiplier	180	7.9722	9.619	68.711	78.33	4.96E−21
	45	2.0788	0.0126	3.8904	3.903	0.017E−21
FP adder	180	25.323	17.7	401.8	419.5	269E−21
	45	14.084	0.7347	9.4553	10.19	2.02E−21
FP matrix multiplier	180	91.8	48.0932	757.2068	805.3	6.786E−18
	45	52.32	0.0672	52.7728	52.32	0.145E−18

4 Conclusion

The floating point matrix multiplier operating at low voltages is designed and simulated at 180 and 45 nm technology nodes. Various parameters like propagation delay, power dissipation, and energy-delay product have been studied. Based on the study, it is concluded that best performance (in terms of EDP) is achieved by fixing supply voltage to 1 V at 180 nm node. The FP matrix multiplier circuits made by using 45 nm technology reduce delay by 43%, power by 93.44%, and EDP by 97.86% as compared to circuits made at 180 nm technology cells at 1 V. In addition, the 45 nm technology cell occupies only 6.25% of the area as compared to 180 nm cells.

References

1. Lin, R.: A reconfigurable low-power high-performance matrix multiplier design. In: Proceedings of IEEE Symposium Quality Electronic Design, pp. 1–8 (2000)
2. Saha, P., Banerjee, A., Bhattacharyya, P., Dandapat, A.: Improved matrix multiplier design for high-speed digital signal processing applications. IET Circuits Devices Syst. **8**, 27–37 (2014)
3. Al-Ashrafy, M., Salem, A., Anis, W.: An efficient implementation of floating point multiplier. In: Proceedings of Saudi International Conference Electronics, Communications and Photonics, pp. 1–5 (2011)
4. Loucas, L., Cook, T.A., Johnson, W.H.: Implementation of IEEE single precision floating point addition and multiplication on FPGAs. In: Proceedings of IEEE Symposium FPGAs for Custom Computing Machines, pp. 107–116 (1996)