ELSEVIER

Microelectronics Journal 31 (2000) 161–168

Microelectronics Journal

www.elsevier.com/locate/mejo

An FPGA implementation guide for some different types of serial–parallel multiplier structures

M.A. Ashour*, H.I. Saleh

Atomic Energy Authority, NCRRT, P.O. Box 29, Nasr City, Cairo, Egypt

Accepted 2 July 1999

Abstract

The multiplier is one of the most important components in the computing and reconfigurable computing systems, especially in the field of digital signal processing (DSP). Hence, in this paper, a performance evaluation and comparison (efficient area and moderate speed) for different serial–parallel multiplier structures have been carried out for the case of their implementation by one of the programmable logic devices, such as a field programmable gate array (FPGA). The implementation of these structures for 8-bit parallel operands has been executed by utilizing the XC4010E chip and Foundation software package V1.3 from Xilinx. The implementation results illustrate the progress in the design area, saving and speeding up the design performance. © 2000 Elsevier Science Ltd. All rights reserved.

Keywords: Digital signal processing (DSP); Serial–parallel multiplier; Field programmable gate array (FPGA)

1. Introduction

Traditionally, digital signal processing (DSP) algorithms are implemented using general-purpose (programmable) DSP chips for low-rate applications. For higher rates, special-purpose (fixed function) DSP chip-sets and application-specific integrated circuits (ASICs) are used to implement DSP algorithms. The progress in field programmable gate arrays (FPGAs) provides new options for DSP design engineers. The FPGA maintains the advantages of custom functionality, like an ASIC, while avoiding the high development costs and the inability to make design modifications after production. The FPGA also adds design flexibility and adaptability with optimal device utilization while conserving both board space and system power, which is often not the case with conventional DSP chips. When a design requires the use of a DSP, or time-to-market is critical, or design adaptability is crucial, then FPGA may offer a better solution.

FPGAs offer a rapid design cycle of programmable DSPs with the flexibility and raw performance of gate array products. Also, the main advantages of FPGAs include:

1. Parts may be reprogrammed over and over. If you want to upgrade your design, you need not replace FPGAs, just reprogram them.
2. FPGAs are pre-tested. Traditional gate array design methodology requires that you also develop costly manufacturing test suites. This task is not required with FPGAs.
3. FPGAs can be dynamically reconfigured within the system. Sophisticated designers can build systems that adapt to changing conditions by altering the circuit configured within the FPGA. This re-configurable design approach is becoming more and more popular since many systems need to perform several different functions, but never all of them at the same time.

Multiplication is an essential function and it is the most currently used arithmetic operation in DSP algorithms. Multiplication of a given number A with n bits by a coefficient B with m bits provides a product P which has, at most, $n + m$ bits. The AND of each multiplier bit and each multiplicand bit $(a_i \times b_j)$ is formed to generate the partial product bits [1]. The partial product bits are then summed up to produce the product. It can be represented by the following equation:

$$P = A \times B = \left(\sum_{i=0}^{n-1} a_i 2^i\right)\left(\sum_{j=0}^{m-1} b_j 2^j\right) = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} a_i b_j 2^{i+j} \quad (1)$$

There are many multiplication algorithms for unsigned, signed, two's complement [1,2], diminished-1 [3,4], Booth [5], modified Booth [6,24], and multi-bit recoding representations [7,8]. Multipliers may be generally classified into

* Corresponding author.
E-mail address: ma_ashour@hotmail.com (M.A. Ashour)

PII: S0026-2692(99)00110-X

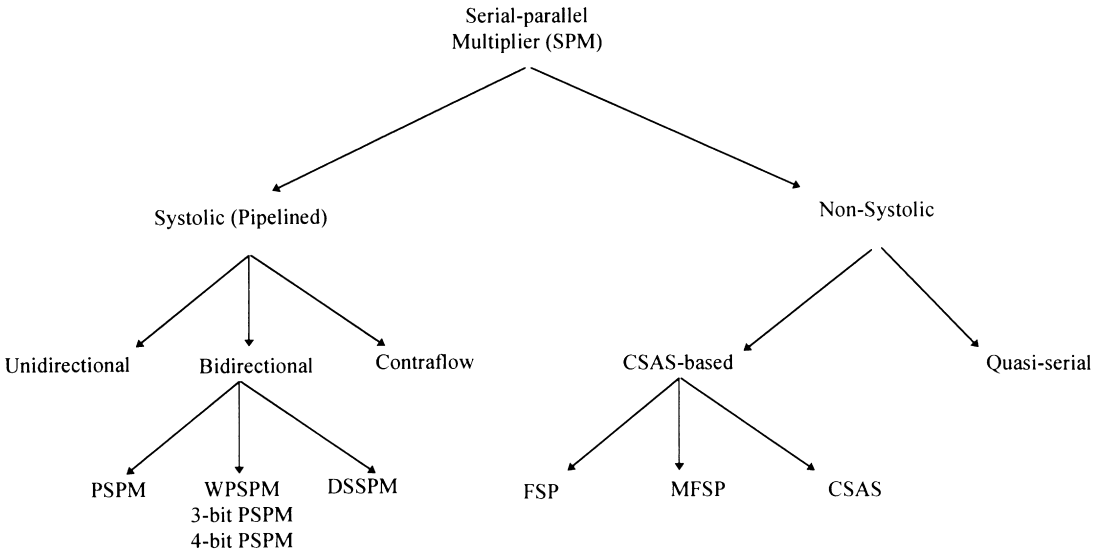

Fig. 1. Classification tree of the serial–parallel multiplier.

array (parallel) [9–11], serial [9,12–15], and serial–parallel multipliers (SPM). SPMs, which are used for hardware simplicity and moderate speed, have one of the two operands (the multiplicand, *A*) loaded in parallel, and the other (the multiplier, *B*) fed serially to the multiplier.

The SPMs are classified, as shown in Fig. 1, into systolic and non-systolic structures. Both, quasi-serial [14,16] and carry-save add-shift (CSAS)-based structures are non-systolic structures. The quasi-serial structure is based on adding one column of the partial products through an *n*-bit parallel counter to obtain one bit of the result at each step [16]. The CSAS-based structures [17,18], perform addition of one of the raw partial products at each iteration and save the carry to the next step. Systolic SPMs are divided into unidirectional [20,21], bidirectional [20,22], and contraflow [19] structures, according to the flow directions of the operands and the partial product.

2. SPM architectures

The conventional CSAS multiplier is shown in Fig. 2 for $m = n = 4$. The structure of this multiplier has four unit cells, where the least three units are identical. Each unit cell has one AND gate to produce the partial product bit of the serial operand bit with the multiplicand (*A*) bit, and one adder with carry save. The most significant unit requires neither adder nor carry save, since there is no addition operation. Multiplication of two 4-bit numbers requires a total of eight clock cycles; four clocks are used for 4-raw carry-save additions, and the other four clocks for the remaining carries propagation. The architecture can be extended for signed operands, if the sign extension is considered. The sign extension can be implemented by the feedback of the sign bit of the previous partial product word to be added with the sign bit of the existing one, as shown in Fig. 2.

Gnanasekaran [17] proposed an $(n-1)$-bit carry-ripple or carry look-ahead parallel adder to perform the addition of sum and carry words, that exist in the CSAS structure after *n* clock cycles, instead of using *n* more clock cycles to propagate the carries. Eliminating the delay caused by the storage elements during the last *n* clock cycles, the structure (fast serial parallel multiplier (FSPM)) operates as a CSAS unit for the first *n* clocks and reconfigures itself as an *n* bit ripple carry adder at the $(n+1)$st clock. This structure is shown in Fig. 3 for the two's complement operands. During the first $(n-1)$ clock pulses, *Q* is zero and the structure acts like a

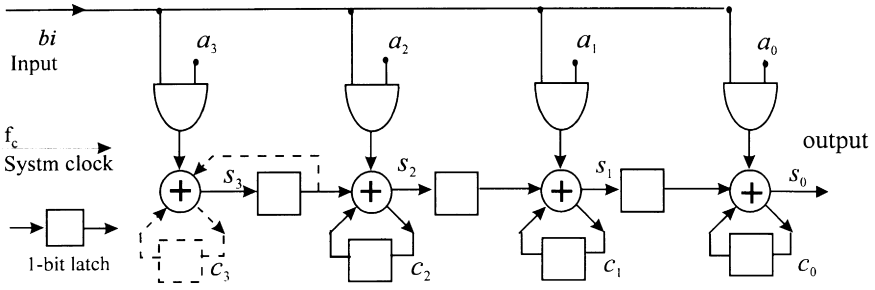

Fig. 2. A 4-bit CSAS multiplier.

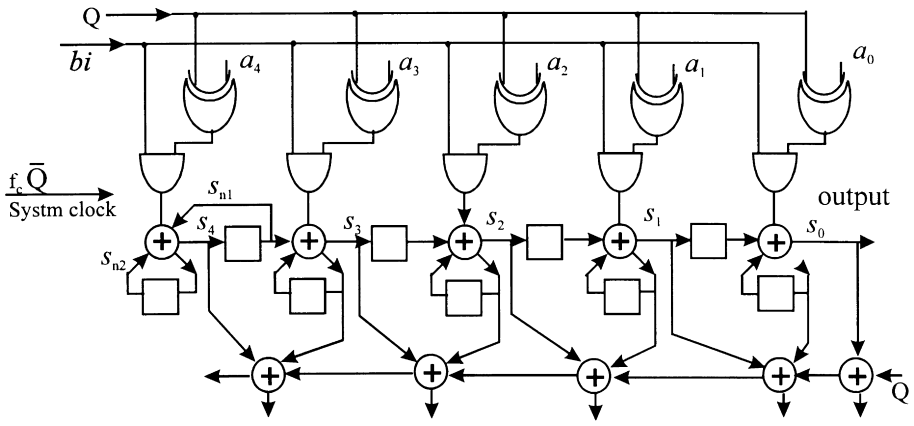

Fig. 3. Two's complement 4-bit FSP multiplier.

CSAS unit. The first $(n-1)$ least significant bits (LSBs) are shifted out serially at the output. Q is set to one by the nth clock, which in turn stops the system clock. For the two's complement FSP multiplier, an n-bit operand, multiplicand or multiplier, is represented as $n+1$-bit number, where the extra one bit is being used as the sign extension of the operand [17].

Avoiding the sign extension of the sum or carry bits, Sunder [18] proposed a modified structure by using a form which uses only positive bit products. Fig. 4 shows the 4×4 structure of the modified FSPM (MFSPM). It can be noted in Fig. 3, that in each of the first three rows of the bit products, one NAND and three AND operations are performed, while in the fourth row, one AND and three NAND operations are performed.

The word "systolic" implies a clocked and pipelined system. Pipelined serial–parallel multipliers (PSPM), which are composed of pipelined unit cells, use either unidirectional or contraflow data streams. No buses are allowed, which means that the only global signals are ground, power and clock signals [20]. Most of these structures use a unidirectional movement of data and partial product. They introduce an initial delay before obtaining the first bit of the product. This delay (latency) is n cycles for an n bit operation ($n=$ size of parallel operand). However, to reduce the latency, data must move in one direction and partial product in the opposite one.

Depending on the systolic architecture, a structure with a higher pipelining degree was proposed [22]. The cell is divided into two different stages working in a pipeline fashion. The first stage performs the addition of the two-logic AND outputs with the previous carry. Once this addition is done, the multiplier bits is shifted to the next left cell, and the product will pass to the next stage. The second stage is used to add the result of the first stage to the partial product shifted from the next left cell. The output of the second stage is shifted to the input of the next right cell and the process is repeated.

Fig. 5 shows a PSPM, which is composed of three kinds of cells: MC (middle cell), RC (rightmost cell) and LC (leftmost cell). The internal control signal Q becomes b_{n-1}, at the last step, otherwise Q is 0. The MC is the same as the basic cell of the unsigned scheme except for two XOR gates, that generate the inverted values of the multiplicand bits when $Q=1$. A conventional unidirectional PSPM was described in Ref. [20]. The multiplier bits are serially fed

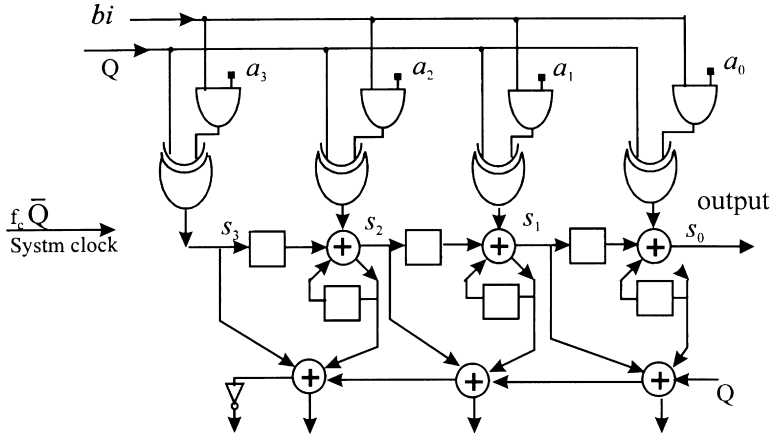

Fig. 4. Modified two's complement 4-bit FSP multiplier.

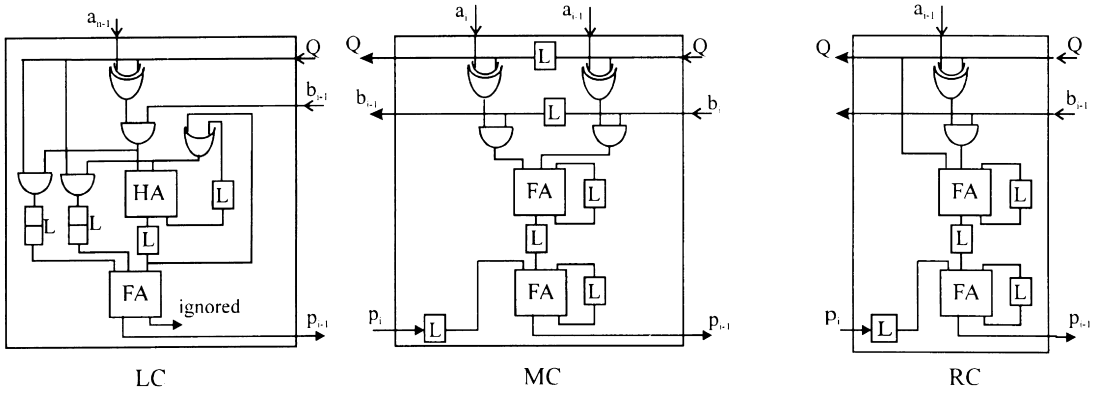

Fig. 5. Detailed diagram of the PSPM.

from one end of the structure. The output bits are obtained at the opposite end, one bit at each cycle. The initial delay can be calculated as nT_C, where T_C is the clock cycle, and n is the number of multiplicand bits.

Based on the modified Booth's algorithm, in which a triplet of bits is treated in every iteration [6], a serial–parallel systolic multiplier was designed by Wu [23]. Adjacent triplets share one bit so that, in every iteration, there is one bit that retires. The multiplicand was recorded by modified Booth to reduce the number of multiplier cells to $n/2$. The cells of the multiplier is connected in a bidirectional PSPM structure, as shown in Fig. 6. It is mentioned as WPSPM.

The contraflow data streams PSPM structure accepts two operands serially, and the result can be received in a parallel

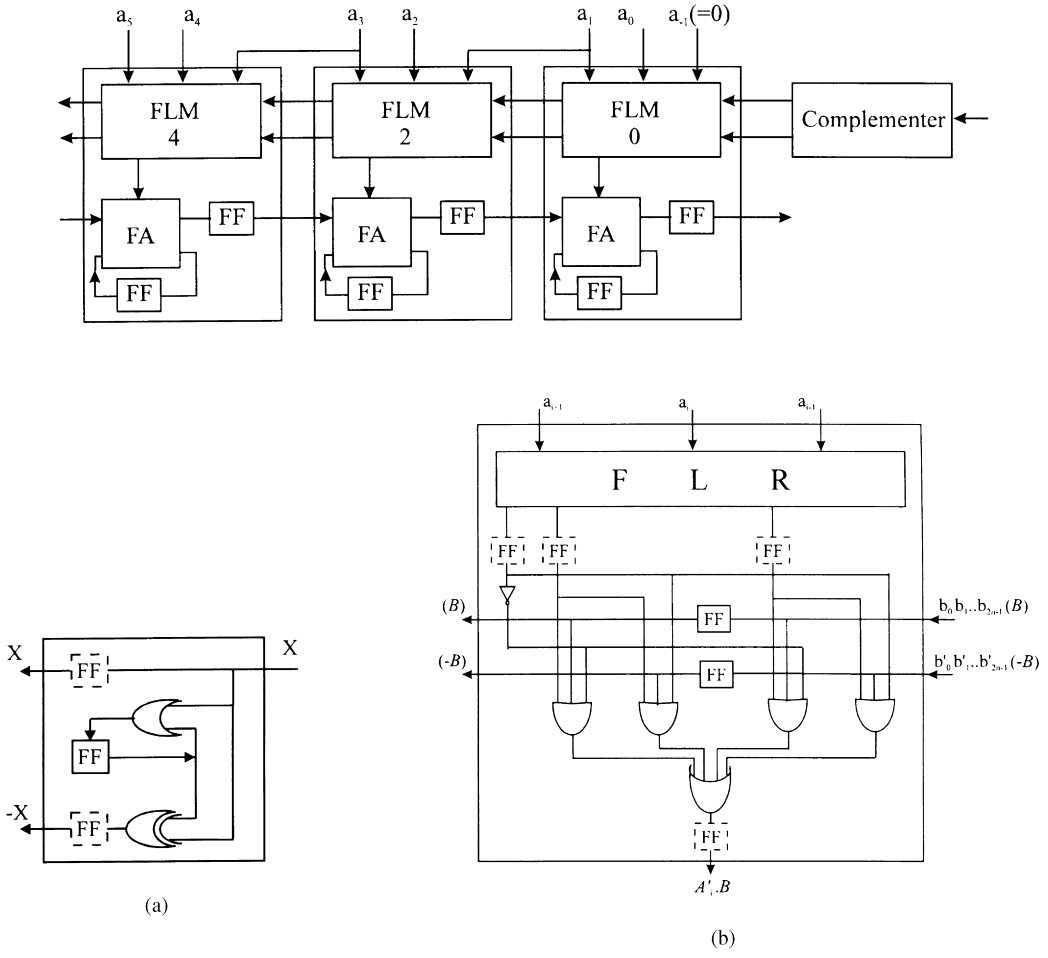

Fig. 6. 4-Bit WPAPM: (a) serial complementer and (b) five-level multiplier.

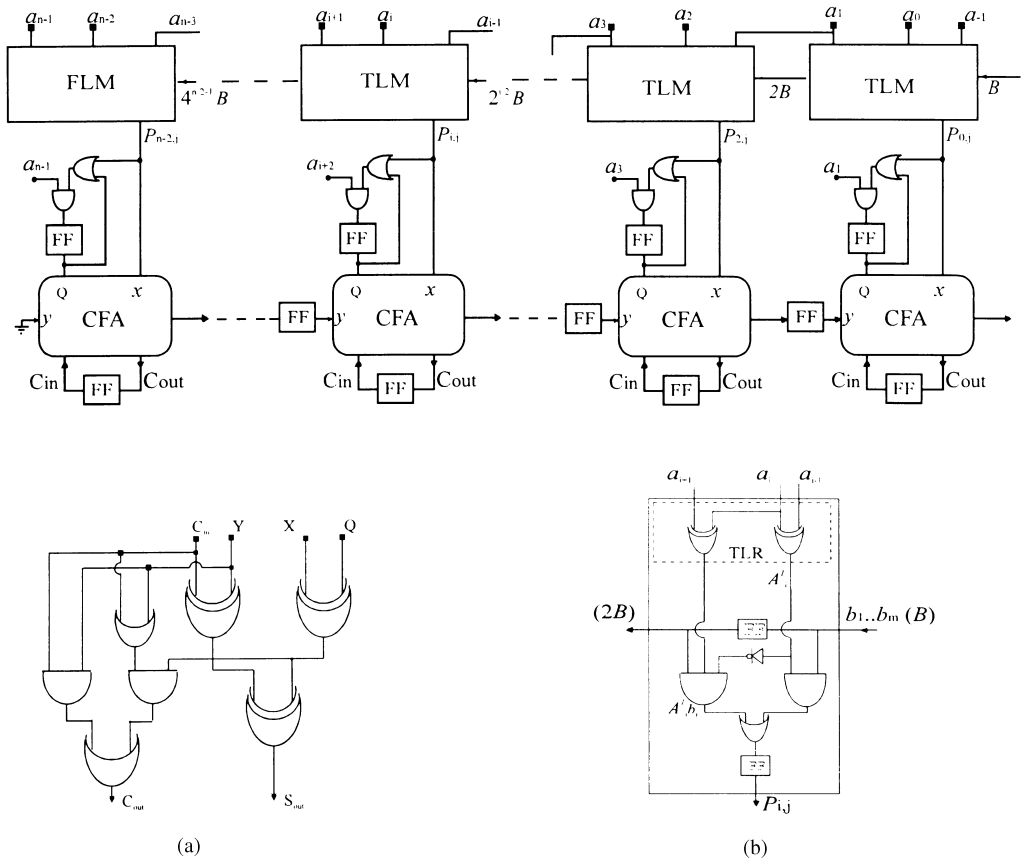

Fig. 7. The proposed 3-bit recoded systolic multiplier architecture: (a) the proposed CFA and (b) the proposed TLM.

form as serial–serial–parallel structure (SSP). The Muller multiplier (MM) and its modified architecture [19] are contraflow data stream PSPMs, where both multiplier and multiplicand are serially fed from the opposite sides of the linear cell array.

The modified recoded-multiplicand systolic multiplier (3b-PSPM), which is similar to the WPSPM [23], is shown in Fig. 7. This modification is to recode the magnitude of the three bits by using a three level recoder (TLR) instead of a five-level recoder (FLR) of WPSPM, and the sign is treated inside the adder by using a controlled full adder (CFA), which acts as adder/subtractor. Recoding the 4-bit of the parallel operand into one cell is used in the 4b-PSPM, which is shown in Fig. 8. FLR is the 4-bit recoder of the parallel operand.

The double speed serial–parallel multiplier (DSSPM) was proposed in Ref. [27]. The DSSPM, in which the multiplier operand is recoded serially into modified Booth codes, is shown in Fig. 9. The multiplier speed is increased, since two bits of the serial operand are multiplied through one clock cycle. The serial-recoder uses a clock (clk1) to feed the serial input to the circuit and outputs recoded signals at each clock cycle of clk2. The frequency of clk1 is twice the frequency of clk2. A multmux subunit is proposed to get the partial products (M_i and M_{i+1}), corresponding to two bits of the multiplicand operand (A) multiplied by the recoded

Table 1
Resources and estimated speed for different 8-bit SPM structure

Multiplier	Maximum estimated CLK	FG	FGH	DFF	CLB	10 pins	CLB O(n)
WPSPM	79.5 MHz	25	4	35	18	10	$2n+2$
PSPM	54.1	33		25	17	11	$2n+1$
CSAS	48.2	32	8	14	16	10	$2n$
MFSP	48.3	44		13	22	19	$3n-2$
FSP	48.3	48	2	15	24	19	$3n$
DSSP	55	39		30	20	10	$2n+4$
4b-PSPM	69	24	1	26	13	10	$1.5n+1$
3b-PSPM	75.5	23	0	32	16	10	$2n$

Table 2
The time, area and complexity of the serial–parallel multipliers in terms of T_N and A_N

Multiplier	Time	Area	Area × time^2 complexity
CSAS-based multipliers			
FSP	$(12.2m + 4.6n + 16.8)T_N$	$(24.5n + 21.5)A_N$	$O(n^3)$
MFSP[a]	$(16.8n - 4.6)T_N$ [a]	$(24.5n - 21.5)A_N$	$O(n^3)$
Unsigned CSAS	$9.9(m + n)T_N$	$(15.5n - 6.5)A_N$	$O(n^3)$
DS-SPM[b]	$(4.4(m + n) + 17.8)T_N$	$(22.125n + 42.5)A_N$	$O(n^3)$
Systolic (pipelined multipliers (PSPM)			
2's-Bi	$12.2((m + n)/2 + 2)T_N$	$(20.25n + 26.75)A_N$	$O(n^3)$
WSPM	$8.4((m + n + 2)T_N$ [c]	$(25.25n + 12.75)A_N$	$O(n^3)$
3b-PSPM[b]	$8.4((m + n + 1)T_N$ [c]	$17.125nA_N$	$O(n^3)$
4b-PSPM[b]	$8.4((m + n + 2)T_N$ [c]	$16.5n + 14)A_N$	$O(n^3)$

[a] This structure was proposed for $m = n$, otherwise the shorter operand is extended to be equal to the other.
[b] The proposed structures.
[c] Fully pipelined structure, the structure with lower degree of pipelined is given in Section 3.

value of the multiplier operand. The proposed structure consists of $n/2$ cells, the tripler and the sign extension unit. Each cell is composed of a multmux and 5-3 counter.

3. FPGA implementation

The previous differently demonstrated structures of 8-bit multipliers are implemented using an FPGA-based peripheral component interconnect (PCI) interface card [26]. The target device in this board is XC4010E, which is used to implement the PCI interface core and the multiplier structure. The Foundation package V1.3 from Xilinx is used for design entry and implementation. Also, a graphical user interface has been developed by using Labview 5.0 to access the implemented multipliers via PCI-bus. From the placement step, the usage resources, such as the number of configurable logic block (CLB), are calculated for each

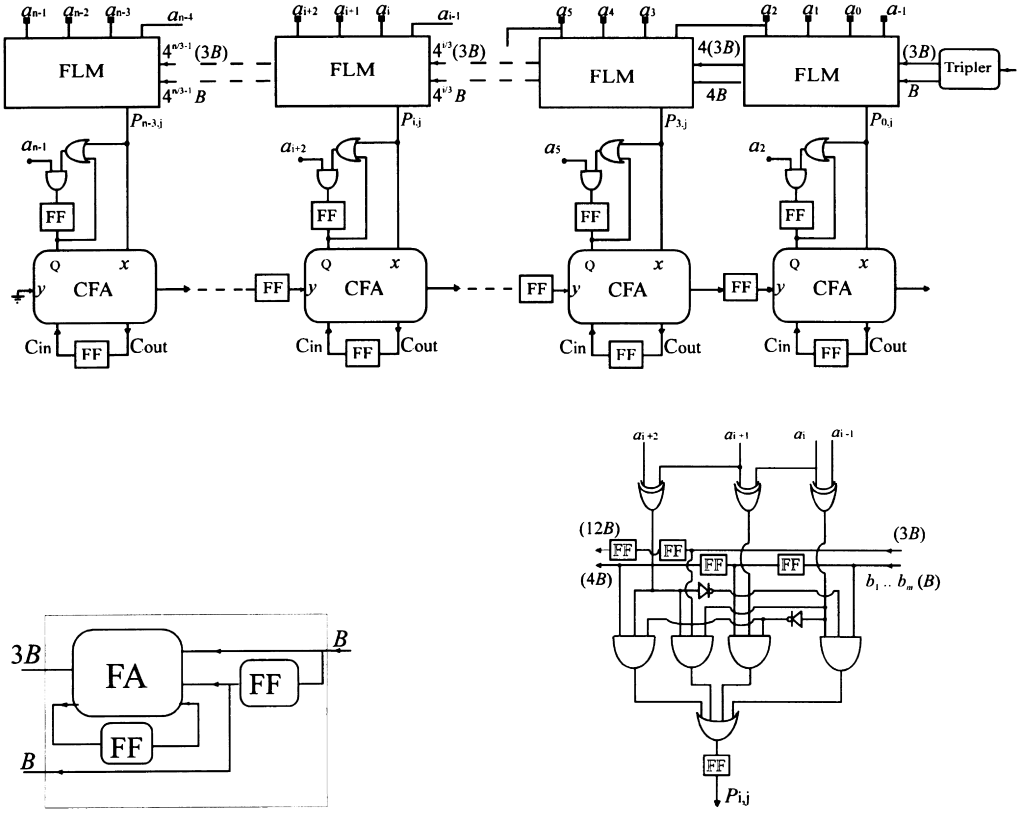

Fig. 8. 4-Bit recoded systolic multiplier architecture: (a) the proposed tripler and (b) the proposed FLM.

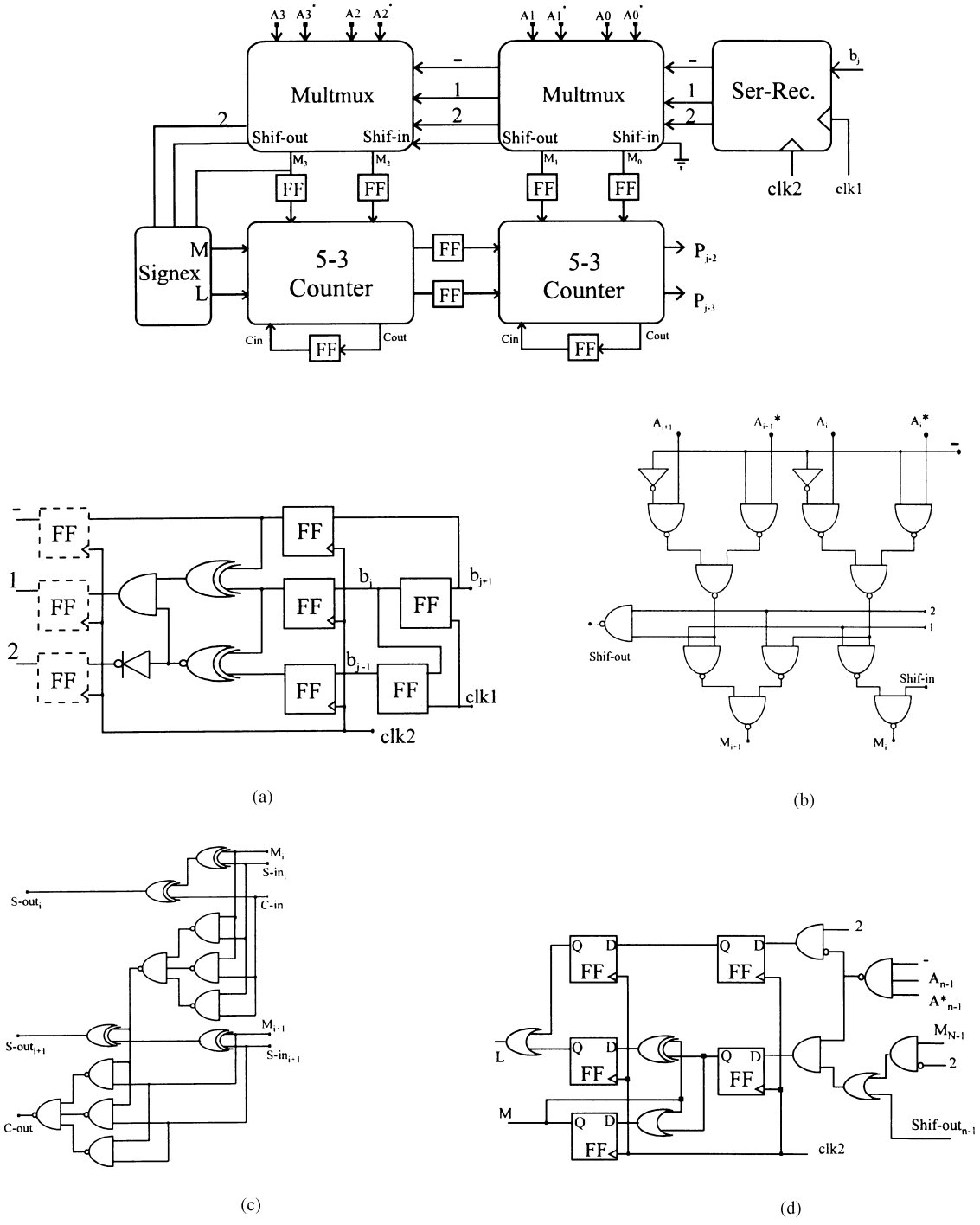

Fig. 9. Double speed SPM: (a) the proposed serial recoder; (b) the proposed multmux; (c) the 5-3 counter; and (d) the proposed sign-extension.

structure. From the routing step, the estimated delays and the maximum clock are calculated. The results of these processes (placement and routing) are illustrated in Table 1 and a comparison between the different SPM structures is carried out. The demonstrated structures were compared based on the delay and area of the two-input NAND gate (TN, AN) [25], to be independent of the implementation technology [27]. According to Table 2, which summarize the two-input NAND-based comparison, the fastest two's complement structure is the DSSPM. The smallest area is the 4-bit recoded PSPM.

4. Conclusion

We can conclude that the 4-bit recoded PSPM has the

minimum area for implementation, but the fastest structures are the WPSPM and 3-bit recoded. Since the implemented multipliers are systolic structures, the implementation resources are in order of the parallel-operand size (n). Therefore, a relation between the number of CLB and n can be estimated as shown in the last column of Table 1. Also, from the systolic structure features, the clock of the multipliers are independent of the multiplier size n.

References

[1] C. Baugh, B.A. Wooly, A two's complement parallel array multiplication algorithm, IEEE Transactions on Computer C-22 (12) (1973) 1045–1047.

[2] P.E. Blankenship, Comments on a two's complement parallel array multiplication algorithm, IEEE Transactions on Computer (1974) 1327.

[3] S. Sunder, F. EL-Guibaly, A. Antoniou, Area-efficient diminished-1 multiplier for Fermat number theoretic transform, IEE Proceedings G-140 (3) (1993) 211–215.

[4] M. Benaissa, A. Pajayakrit, S. Dlay, A. Holt, VLSI design for diminished-1 multiplication of integers modulo a Fermat number, IEE Proceedings 135 (3) (1988) 161–164.

[5] A.D. Booth, Signed binary multiplication technique, Quarterly Journal of Mechanical and Applied Mathematics 4 (1951) 20.

[6] O.L. MacSorley, High-speed arithmetic in binary computers, Proceedings of IRE 49 (1961) 67–91.

[7] H. Sam, A. Gupta, A generalized multibit recoding of two's complement binary numbers and its proof with application in multiplier implementations, IEEE Transactions on Computers C-39 (8) (1990) 1006–1015.

[8] S. Vassiliadis, E. Schwarz, D.J. Hanrahan, A general proof for overlapped multiple-bit scanning multiplications, IEEE Transactions on Computers C-38 (2) (1989) 172–183.

[9] M.K. Ibrahim, Radin-2n multiplier structures: a structured design methodology, IEE Proceedings 140 (4) (1993) 185–190.

[10] S. Sunder, F. EL-Guibaly, A. Antoniou, Area-efficient multipliers for digital signal processing applications, IEEE Transactions on Circuits and Systems—II: Analog and Digital Signal Processing 43 (2) (1996) 90–95.

[11] K.M. Elleithy, M.A. Bayoumi, Systolic architecture for modulo multiplication, IEEE Transactions on Circuits and Systems—II: Analog and Digital Signal Processing 42 (11) (1995) 725–729.

[12] I. Chen, R. Willoner, An O(n) parallel multiplier with bit-sequential input and output, IEEE Transactions on Computers C-28 (10) (1979) 721–727.

[13] R. Gnanasekaran, On a bit-serial input and bit-serial output multiplier, IEEE Transactions on Computer C-32 (9) (1983) 878–881.

[14] T.G. McDonald, R.K. Guha, The two's complement quasi-serial multiplier, IEEE Transactions on Computers (1975) 1233–1235.

[15] W. Marwood, Generalized systolic ring serial floating point multiplier, Electronics Letters 26 (11) (1990) 753–754.

[16] E.J. Swartzlander, The quasi-serial multiplier, IEEE Transactions on Computers C-22 (1973) 317–321.

[17] R. Gnanasekaran, A fast serial–parallel binary multiplier, IEEE Transactions on Computer C-34 (8) (1985) 741–744.

[18] S. Sunder, F. EL-Guibaly, A. Antoniou, Two's complement fast serial–parallel multiplier, IEE Proceedings of Circuits Devices System 142 (1) (1995) 41–44.

[19] M.B. Tosic, M.K. Stojcev, Pipelined serial/parallel multiplier with contraflowing data streams, Electronics Letters 27 (25) (1991) 2361–2363.

[20] P.E. Danielson, Serial–parallel convolvers, IEEE Transactions on Computer C-33 (7) (1984) 652–667.

[21] D. Ait-Boudaoud, M.K. Ibrahim, B.R. Hays-Gill, Novel pipelined serial/parallel multiplier, Electronics Letters 26 (9) (1991) 582–583.

[22] D. Ait-Boudaoud, M.K. Ibrahim, B.R. Hays-Gill, Novel cell architecture for bit level systolic arrays multiplication, IEE Proceedings E 138 (1) (1991) 21–26.

[23] I. Wu, A fast 1-D serial–parallel systolic multiplier, IEEE Transactions on Computer C-36 (10) (1987) 1243–1247.

[24] L.P. Rubinfield, A proof of the modified Booth algorithm for multiplication, IEEE Transactions on Computer (1975) 1014–1015.

[25] Hall, The CMOS Standard Cell Library, Report ICI-021r0, Queen's University, Kingston, 1990.

[26] H. Saleh, P. Reinhard, R. Engels, R. Reinartz, F. Rongen, A flexible compatible PCI interface for nuclear experiments, IEEE Nuclear Science Symposium in Albuquerque, USA November (1997) 9–15.

[27] H. Saleh, A. Khalil, M.A. Ashour, A. Ezzat, Novel serial–parallel multiplier and dividers, submitted for publication.