Design of a 16-by-16-bit Unsigned Serial-parallel Multiplier using Retime Technique

Amirhossien Vafi¹, Ziaddin Daie Kokehkanani¹, Jafar Sobhi¹, Mousa Yousefi²*

1- Department of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran.
   Email:amir_y65@yahoo.com
2- Department of Engineering, Azarbaijan Shahid Madani University, Tabriz, Iran.
   Email: m.yousefi@azaruniv.ac.ir (Corresponding author)

Received: February 2019   Revised: May 2019   Accepted: August 2019

ABSTRACT:
In this paper, the structure of a 16-by-16 unsigned hybrid (serial-parallel) multiplier has been proposed. Parallel multipliers, in comparison with serial multipliers, have higher speed and higher power consumption. In hybrid structures, to reduce power and increase speed, both serial and parallel techniques are used. The proposed structure improves propagation delay and reduces power consumption using pipeline and retime techniques. Simulation results show that it has 5.7 ns propagation delay and 2.65 mW power consumption. The figure of merit for energy consumption is 15.2 PJ. The proposed multiplier has been designed using 0.18 μm TSMC process at 1.8 V supply and simulated using Cadence tools. The layout of the multiplier occupies 52414 μm².

KEYWORDS: Multiplier, Serial-Parallel, Unsigned, Pipeline, Retime.

1. INTRODUCTION
The main reason for the widespread presence of electronic systems in daily human life is impressive advances in the manufacturing of electronic circuits. Especially, the possibility of integration of different parts like analog, digital, radio frequency on a single chip and shrinking of transistor size is an advantage. An essential issue in the design of integrated system-on-chip is the reduction of consumed silicon area. On the other hand, another challenge in many electronic systems like wireless sensors, communication systems, control systems and human vital signal monitoring is the limited energy source. So, all electronic systems must have low power consumption [1-2].

In a mixed-signal electronic system, a processor is needed for digital signal processing. A processor needs a strong arithmetic unit to perform these calculations. Strength of a processor depends on its Arithmetic and Logic Unit (ALU) in which multiplier is a very crucial part.

In general, an electronic system must have low power consumption. In wireless systems, it is the main challenge. On the other hand, to increase the performance of system-on-chips, speed must be increased, which is in contrast with a low power requirement. Considering all these discussions, a multiplier must have high speed, low power and small area [3-5].

Multipliers are generally two types: serial and parallel. Parallel multipliers have higher speed and higher power consumption in comparison with serial counterparts. Multiplier implementation includes three stages: partial product generation unit, reduction of partial products and carry addition unit. Considering the application of multipliers and data access method, partial products can be generated serially or in parallel. Final adder regarding high speed or low power requirement can be Carry Look Ahead (CLA) or Ripple Carry Adder (RCA). Booth algorithm can reduce partial products by increasing the occupied area [6-7].

In the design of multipliers, a significant challenge is carry propagation. For digital integrated circuit designer, design of high speed and low power multiplier is an important issue. Many algorithms have been proposed. For example, Braun, Baugh-Wooley, Wallace tree and Booth algorithms can be named [7-10].

As mention previously, parallel multipliers, including array and tree multipliers, have better speed performance. Array structure has higher speed, but it needs large silicon real estate. Tree structure, also, because of asymmetric connections occupies the large area [11-12].

In contrast, serial multipliers have lower speed and power consumption and need a smaller area. Since system-on-chips contain various units, in these systems to reduce output pins and in other words, to reduce wiring costs, serial input-output ports are widely used. In addition, this problem arises in multipliers with higher
number of bits. So, the design of high-speed serial multipliers in these systems is crucial [13-15].

Considering the points mention above, designers aim to use the advantages of both serial and parallel multipliers. By combining different techniques, optimized structures from speed, power and area point of view can be designed, which are called hybrid structures [16-19]. In following, several hybrid structures are described.

In [19], a hybrid radix-4/8 multiplier has been proposed for Graphics Processing Unit (GPU) applications which combine advantages of radix-4 (high speed) and radix-8 (low-power). Another design combines Complementary Metal Oxide Semiconductor (CMOS) and Pass Transistor Logic (PTL) logic for D and T flip flop and logic gate implementation to reduce power and area and increase speed [20]. Elimination and reduction of partial bits reduces the number of full adders and half adders, which would result in reduced area and increased speed [21]. Hybrid tree structure uses DADDA and Wallace methods to reduce partial products. For a group of them, DADDA and for another group Wallace has been used. The final stage is the adder.

In this paper, a serial-parallel multiplier structure has been proposed. One input is serial, and the other one is parallel. Speed performance is boosted using pipeline and Wallace techniques. Moreover, using the retiming technique reduces power consumption.

The rest of this report is organized as follows: In Section 2, the structure of the proposed multiplier is presented. In addition, the description of the primary serial structure and different part like retiming, pipeline and Wallace implementation method are located in this section. In Section 3, transistor-level circuits of different parts with transistor sizing are presented. Section 4 includes simulation results for the proposed multiplier.

2. THE STRUCTURE OF THE PROPOSED MULTIPLIER

Typically, in serial multiplication 2n clock cycle is needed to multiply two n-bit numbers. To get final result, n clock cycle is used for the carry-save row, and the next n cycle is used for the remaining tasks. The structure of the proposed multiplier has been shown in Fig. 1 which contains one serial and one parallel input. It is similar to the Carry Save Add Shift (CASA) structure in which input is fed serially from the least significant bit and outputs in a similar manner. In the proposed multiplier, by applying the retimed technique and the Wallace tree structure, the cycle of clock, for generating product of multiplier, is reduced to n cycle.

2.1. The Proposed Multiplier with the Pipeline Technique

This structure is designed in a way that a CASA unit operates for n clock signal and resets as an n-bit parallel number in (n+1)’th clock cycle. Using this technique, the delay resulting from storage elements is eliminated. In other words, every stage gets serial input with parallel and after n clock signal, multiplication is done.

As we know, the serial structure has low power, but it needs more time to get the final result. So, the most important challenge in their design is to apply methods to increase multiplication speed. It must be noted that increasing speed must not increase power consumption very much. One common speed boosting method is circuit synchronization by injection of a clock signal, which can be implemented by pipeline technique. This method has been utilized in the proposed multiplier. However, is should be noted that in the pipeline method, system changes periodically with clock changes, which increases power consumption. In other words, in pipeline technique, clock signal is located in output path by register it causes that part of the circuit stays on path even when it has produced partial products and generated Partial Product Summation (PPS) result. Implementation method for this structure has been shown in Fig. 2. The proposed structure has been implemented by flip flops, full adder and other logic AND gates. Registers are for the implementation of the pipeline technique.

Retiming is a continuous optimization technique. In synchronous circuits by replacing some registers while the performance of the circuit as a whole and critical path are not changed, it helps to increase the speed. Another advantage of retiming is the elimination of clock skew. Furthermore, changes in clock signal between registers, which is widely separated can cause clock signal error and can affect circuit performance substantially.
When the clock signal path is drawn on layout, by increasing the length of signal path synchronization is practically lost because the path for the signal of the last stage is not the same. So, the clock signal is not injected at the same time, and synchronization is lost, and the final result is not correct. Clock skew is a serious problem in layout which is eliminated by the retiming technique. When register is replaced from output signal path, in transistor level layout, clock signal path is drawn on layout which does not include register affecting the output of the multiplier, so clock distribution problem does not exist anymore. In other words, outputs of all adder which produce partial products do not wait for the clock signal.

As it is shown in Fig. 3, using the retiming technique, the flip flop which had been placed at the output of the serial circuit to implement pipeline, is moved cross upper to multiplication input. There is no register on the signal path. Since registers are removed from the signal path, the circuit does not depend on the clock signal and power will be reduced. As a result, transistor level layout problem is solved. In addition, it was possible that a noise accompanied by a clock signal enters the circuit and affects the final result, which, is not the case by using the retime technique.

2.2. Wallace Structure in the Proposed Multiplier
Another challenge in the design of multipliers is increasing the speed which is done typically using parallel structures. One of these structures is a Wallace tree. Considering design rules, the Wallace tree has been designed. Finally, a fast Carry save Adder (CSA) is used to generate final products. In the following, the operation of the Wallace tree is described.

1. Partial products must be grouped into three categories:
2-Partial product which is higher than 3 bits and in special positions must be added to carry of the previous stage in next level.

3-Half adder must only be used in input-output stages, and the number of half adder in each tree must be higher than one.

4-Sign format bits are added in the last stage or the stage before it, to avoid unnecessary changes in the tree.

Design of Wallace tree is such that each bit passed through D flip flop has weight and every time result of the multiplier pass through D flip flop that it is multiplied by 2. Weight bits are entering together to a half adder or full adder to generate final result at the end of Wallace tree.

In Fig. 4, same order bits entering half adder are colored in red and same order bits entering full adder are colored in black, and finally, the results enter half adder.

For example, 4 low order bits of multiplier a0–a4 are generated from multiplying 4 low order bits of serial input to 16 parallel bits.

B0 and a4 are generated respectively, a4 is generated by multiplying 4 low order bits of serial input to the 16-parallel number after passing associated shift registers shown in Fig. 3 which have higher weight than a3 places after a3. B0 is the product of second 4 bits of low weight serial multiplied by 16-bit parallel. Since a4 and B0 have the same significance, both enter the same half adder.

For example, a8, which is a product of 4 low order bit of serial multiplied by 16 bit parallel, after 8 clock
signal passed through the associated register and generated as Fig. 4. Moreover, B₄, which is the product of the second 4 bits of low weight serial multiplied by 16-bit parallel after passing shift register is generated after 4 clock signals. Noting that C₀ is the product of third 4 bits of lower weight serial multiplied by 16-bit parallel and all three numbers are of the same weight that enter the same full adder. To generate the final result, the Wallace tree is used. Fig. 4 shows the implementation of this multiplier. As said before, bits with the same weight are added and generate the output of the Wallace tree.

![Wallace Tree Diagram]

**Fig. 4.** Implementation of the Wallace tree.

Carry and Sum outputs of the previous step are summed by half adder with Carry and Sum generated by the previous stage because they have the same weight to produce final Wallace tree output. This process continues like this, and it is same for outputs resulting from multiplying 4 high order bit of serial by the 16-bit parallel number which is shown in the figure by D₀ that directly enter full adder with the result of all previous addition which generated Sum and Carry. Considering abovementioned rules and described model for Wallace tree, its block diagram has been shown in Fig. 5.

**3. TRANSISTOR LEVEL IMPLEMENTATION**

By investigating all multiplier structures, it can be concluded that all of them have a fundamental block called full adder. Its performance has a strong effect on multiplier the figure of merits. On the other hand, to reduce delay and power consumption in the proposed technique, blocks and structures are used, which as finally led to reduction of propagation delay and power consumption of the multiplier.

**3.1. Full Adder Structure**

Many circuits have been developed for full adders considering overall multiplier systems. In the design process, different full adder structures are implemented in transistor level using TSMC 180nm CMOS process.

![Full Adder Circuit Diagram]

Post layout results are summarized in Table 1. In all simulation, a 1 pf load capacitor is connected to the output of all full adders. Considering all results, finally, the transistor level structure shown in Fig. 6 has been selected. This structure includes 10 transistors with sizing presented in Table 2.

**Table 1** Simulation Results for all investigated full adders.

<table>
<thead>
<tr>
<th>Adders</th>
<th>Number of Transistors</th>
<th>Pdc (mW)</th>
<th>Propagation Time (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TG</td>
<td>20</td>
<td>0.031</td>
<td>78</td>
</tr>
<tr>
<td>Pseudo-NMOS CMOS</td>
<td>14</td>
<td>0.036</td>
<td>90</td>
</tr>
<tr>
<td>Double Pass Transistor</td>
<td>28</td>
<td>0.051</td>
<td>150</td>
</tr>
<tr>
<td>Complementary pass Transistor [22]</td>
<td>48</td>
<td>0.088</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>0.1</td>
<td>62</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>0.016</td>
<td>58</td>
</tr>
</tbody>
</table>

**Table 2.** Sizes of the transistors for full adder depicted in Fig. 6.

<table>
<thead>
<tr>
<th>L(µm)</th>
<th>W(µm)</th>
<th>Transistors</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.18</td>
<td>1.9</td>
<td>M₂₅,₂₆,₂₇,₂₈,₂₉</td>
</tr>
<tr>
<td>0.18</td>
<td>0.43</td>
<td>M₂₅,₂₆,₂₇,₂₈,₂₉</td>
</tr>
</tbody>
</table>
Fig. 5. Block diagram of Wallace tree used in the proposed multiplier.
3.1. Flip flop structure

Rising edge triggered D flip flop has been used in this multiplier. Fig. 7 shows the transistor-level circuit for the flip flop. Sizes of the transistors are presented in Table 3. In addition, implementation of AND and OR gates is shown in Fig. 8. Sizes of the transistor are summarized in Table 4.

4. SIMULATION RESULTS

Quartus software has been used to verify the operation of the proposed multiplier. a1, a2, a3 and a4 are in fact 4 bit sections of serial input. Consider an as 4 bit sections. As it is shown in Fig. 9, the other multiplier input b_1 is 16-bit parallel input to the proposed multiplier. As it can be seen, a1 low order 4-bit section is 0001, second 4-bit section a2, is 0001. Furthermore, a3 has the value of 0000, and finally, the last 4-bit section is equal to 0000. The first number is 0000000000010001 (17 in decimal). The second input is equal to 3 and fed in parallel. 17 multiplied by 3 is equal to 51. As can be seen in Fig. 9, Number 51 is generated after 220 ns, which is equal to 20 clock cycles. Period of the clock is 10ns and inputs are 15 ns. After 20 clock cycles, the final result is generated. These 20 cycles are equal to 16-bit parallel and 4-bit serial. In other words, while in serial multiplier to multiply n by n serial numbers we have to wait 2n cycles, in this method cycles are reduced to n+4.

The layout of the proposed 16 by 16 multiplier has been done using 180nm TSMC CMOS technology in Cadence. As shown in Fig. 10, the proposed multiplier occupies 52414 μm² of the silicon wafer.

To compare different multiplier structures, beside power consumption and propagation delay, Power Delay
Product (PDP) can be used in order to compare merits of different multiplier as shown in comparison table (Table 5), in last column a criterion called PDP is specified which can be used to compare different structures regardless of their occupied area. In comparison, the fabrication process also must be taken into account. As table 5 indicates that the proposed structure has consumed about one-third less power in comparison with [15], and about half less power in comparison with retimed-multiplier. However, delay propagation of the proposed multiplier to some extent is higher than [15], but in comparison with the retimed-multiplier, it is about half less.

5. CONCLUSION
In this paper, the structure of an unsigned 16×16 serial-parallel multiplier has been presented. This structure utilizes retime technique to improve propagation delay and reduce power consumption. Simulation results show that its propagation delay is 5.7 ns while consuming 2.65 mW. The figure of merit for energy consumption is 15.2 PJ. The structure of the proposed multiplier has been designed and simulated using 180 nm TSMC CMOS process in Cadence. The layout of the multiplier occupies 52414 μm². In comparison with none retimed multiplier [27] power consumption of the proposed structure is mostly decreased and also in comparison with retimed multiplier [27], but none pipelined technique in the proposed multiplier has less power consumption.

![Fig. 9. Simulation results for the proposed multiplier.](image)

![Fig. 10. The layout of the proposed multiplier.](image)
Table 5. Comparison table for several multiplier structures.

<table>
<thead>
<tr>
<th>References</th>
<th>Technology (nm)</th>
<th>Power (mW)</th>
<th>(µm²) Area</th>
<th>Propagation Time (ns)</th>
<th>EDP (P)</th>
<th>Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>[24]</td>
<td>45</td>
<td>2.667</td>
<td>NA</td>
<td>0.75</td>
<td>2</td>
<td>16x16</td>
</tr>
<tr>
<td>[21]</td>
<td>32</td>
<td>1.157</td>
<td>NA</td>
<td>2.747</td>
<td>3.18</td>
<td>16x16</td>
</tr>
<tr>
<td>[13]</td>
<td>90</td>
<td>6.85</td>
<td>15000</td>
<td>7.92</td>
<td>54.3</td>
<td>16x16</td>
</tr>
<tr>
<td>[25]</td>
<td>90</td>
<td>0.560</td>
<td>5104</td>
<td>4.2</td>
<td>2.35</td>
<td>16x16</td>
</tr>
<tr>
<td>[26]</td>
<td>180</td>
<td>4.182</td>
<td>38723</td>
<td>NA</td>
<td>NA</td>
<td>16x16</td>
</tr>
<tr>
<td>[14]</td>
<td>180</td>
<td>4.3</td>
<td>NA</td>
<td>4.96</td>
<td>21.4</td>
<td>16x16</td>
</tr>
<tr>
<td>[15]</td>
<td>180</td>
<td>9.11</td>
<td>372177</td>
<td>4.71</td>
<td>42.9</td>
<td>16x16</td>
</tr>
<tr>
<td>[27]-none retimed</td>
<td>90</td>
<td>9.1</td>
<td>NA</td>
<td>2.87</td>
<td>NA</td>
<td>16x16</td>
</tr>
<tr>
<td>[27]-Retimed</td>
<td>90</td>
<td>6.31</td>
<td>NA</td>
<td>2.35</td>
<td>NA</td>
<td>16x16</td>
</tr>
<tr>
<td>This Work</td>
<td>180</td>
<td>2.65</td>
<td>52414</td>
<td>5.7</td>
<td>15.2</td>
<td>16x16</td>
</tr>
</tbody>
</table>

REFERENCES

[18] X. Jiayang, P. K. Meher, and J. He, “Low-latency Area-delay-efficient Systolic Multiplier over GF (2 m) for a Wider Class of Trinomials using Parallel

57
58