Proposal for an Efficient Reconfigurable Fixed-Width Multiplier

ASWATHY SUDHAKAR and D. GOKILA
VLSI Design Group, Department of Electronics and Communication Engineering
Amrita School of Engineering
Amrita Vishwa Vidyapeetham
Coimbatore
India
1achu1921@gmail.com, 2d_gokila@cb.amrita.edu

Abstract :-
FPGA device sizes can only increase as fast as the silicon process technology will allow. The multipliers in FPGAs are implemented separately with separate functionalities even if they are performing the same calculation. The redundancy caused by this can be removed by Dynamic Reconfiguration. The reconfiguration problems limit conventional multiplier architectures for getting efficient reconfigurable structures. In this proposal, the reconfiguration on Baugh-Wooley multipliers is explained with respect to the optimum reconfiguration constraints. The proposed architecture improves the reconfiguration speed in Baugh-Wooley fixed-width multipliers which is normally limited due to two’s complement calculation by 2-D pipeline gating technique. This paper discusses techniques to improve the reconfigurable multiplier characteristics like accuracy, low-resolution, speed, redundancy reduction and latency reduction. Due to the timing mismatches in the Baugh-Wooley architectures, latency problems occurs that is reduced through pipelining of multiplier architectures using pipeline registers. Sub-word multiplication technique is used to make the multiplier adaptable for low-resolution multiplication.

Key-Words:-
Baugh-Wooley multiplier, dynamic reconfiguration, fixed-width multiplier, resolution, reconfiguration speed, sub-word multiplication, 2-D pipeline gating and pipeline register.

1 Introduction

Multipliers are widely used in almost all fields of applications like communication, speech processing, digital signal processing etc. Among these multipliers, the basic add-and-shift multiplier with carry-save scheme [1] performs operations similar to the unsigned multiplication performed by paper and pencil. For the reduction of \( n^2 \) summands \( n-1 \) half-adders and \( n-1 \) full-adders are needed resulting in slow multiplication. C.S. Wallace proposed a tree multiplier architecture [2] which performs fast unsigned multiplication of operands. The speeds achievable appear to be greater by a factor of at least four than those obtained in conventional units. This has a high structural irregularity problem which is later removed by the Dadda scheme [3] which is a high-speed, regular, unsigned multiplication technique.

In all these, the repetition of the multiplier modules serving the same functionality being implemented separately results in redundancy causing an area overhead which is not allowable above some limits by the silicon process technology even though this seems easy to implement. Also a multi-million gate FPGA era is the order of the day. When the number of gates in an FPGA area increases, power consumption of the chip also increases. When a constant area is used for implementation, redundant modules cause heat loss resulting in device damage after certain manageable limits. It is essential to minimize these redundant multipliers which are implemented separately but serving the same functionality at different instances. With dynamic reconfiguration [4,5], the same multiplier modules are designed to perform the multiplication function at different times. The unsigned multiplication architecture makes tree architectures and conventional add-and-shift multipliers poor choices for reconfiguration.

Later, A. D. Booth [6] proposed a multiplication technique for signed binary numbers which is a good candidate for reconfiguration. Area overhead, speed limitation and delay problems become severe here. Also accuracy reduction [7] in low-resolution multiplication caused by the redundant bits in the sign-extension circuit limits its reconfiguration efficiency. Baugh and Wooley developed an array multiplier architecture [8] which provides high-speed, regular and area-efficient signed multiplication. In [9], the presented work reduced the accuracy degradation in fixed-width multipliers by truncation with rounding technique which has accuracy almost equal to the rounding technique with a little circuit complexity. With the 2-D pipeline gating [10] technique in the Baugh-Wooley reconfigurable multiplier, the low-resolution errors are minimized. But this has area overhead and increased redundancy and are
minimized in [11, 12] by the sub-word multiplication technique.

Here the limited pipelining structures fail to achieve high speed and 2-D pipeline gating causes timing mismatch which leads to increased latency. In this paper, a proposal for efficient reconfigurable multiplier architecture is given to optimize the speed, area, accuracy and latency characteristics. The rest of the paper is organized as follows. The proposed architecture is explained in Section 2. In Section 3, implementation results in terms of power, area and speed for 4-bit multipliers and comparisons are presented.

2 Proposed Architecture

The proposed architecture for efficient reconfigurable multiplier is shown in Fig.1. In 2n×2n multiplication truncation with rounding is used to generate the partial product bits. Proposed improvements in the Baugh-Wooley multiplier architecture is shown in Fig. 1(a). Here partial product bits created are stored in the pipeline registers. After segregation of partial products pipeline registers follows the positive partial product block. This avoids the delay and the latency caused by the two’s complement calculation. The partial product bits are ensured to be ready for summation at the same time so that the errors due to latency can be removed. Latency causes the error to be propagated to the next stages of calculations. Pipelined registers are used to decrease the delay, thereby improving the speed. Here the multiplier module itself is pipelined along with the pipelined reconfigurable structure.

The reconfigurable structure using the above efficient Baugh-Wooley multiplier is shown in Fig.1 (b). Once the reconfiguration is described next is to define the reconfigurable modules that can be implemented as a single module and can be shared among the two modes that are 2n×2n fixed-width and n×n fixed-width multiplier. Configuration mode 1 (CM 1) is assigned as the 2n×2n multiplication. Configuration mode 2 (CM 2) is the n×n fixed-width multiplication.

The configuration modules are selected by mode select bits. These bits select which multiplier functionality to be performed in a particular time. The mode select bits are determined according to the reconfigurable regions or modules designed. The truncation with rounding technique generates n-bit output product for a 2n×2n multiplier. Pipeline registers improves reconfiguration speed. When these registers are used the continuous streams of data is possible through the multiplier and reconfiguration architecture since intermediate results can be stored thus improving the speed.

The n×n multiplication is a sub-word multiplication technique of 2n×2n multiplier as shown in Fig.2. In this multiplication technique, the 2n-bit multiplication process is divided into various sub-words; here for example, in Fig. 2(a), the n×n multiplication can be performed with the same 2n×2n implementation without any module redundancy. The partial product array diagram is shown in Fig.2 (b). The 2n-bit input operands are split into n-bit called the sub-words. The multiplication of these sub-words is performed independently so that the low resolution errors due to data bits overlapping in 2-D pipelined gating are removed.
### Table 1: Synthesis results of different 4-bit non-pipelined multiplier architectures

<table>
<thead>
<tr>
<th>Multiplier Architecture</th>
<th>Number of LUTs</th>
<th>Fan-out</th>
<th>Clock-Period (ns)</th>
<th>Power Dissipation (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-and-Shift</td>
<td>52</td>
<td>18</td>
<td>16.918</td>
<td>68.56</td>
</tr>
<tr>
<td>Wallace Tree</td>
<td>34</td>
<td>123</td>
<td>15.852</td>
<td>68.52</td>
</tr>
<tr>
<td>Dadda Tree</td>
<td>30</td>
<td>102</td>
<td>15.877</td>
<td>68.44</td>
</tr>
<tr>
<td>Booth</td>
<td>42</td>
<td>108</td>
<td>16.318</td>
<td>67.98</td>
</tr>
<tr>
<td>Baugh-Wooley</td>
<td>28</td>
<td>104</td>
<td>15.861</td>
<td>67.84</td>
</tr>
</tbody>
</table>

### Table 2: Synthesis results of different 4-bit pipelined multiplier architectures

<table>
<thead>
<tr>
<th>Multiplier Architecture</th>
<th>Number of LUTs</th>
<th>Fan-out</th>
<th>Clock-Period (ns)</th>
<th>Power Dissipation (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-and-Shift</td>
<td>74</td>
<td>18</td>
<td>8.939</td>
<td>68.89</td>
</tr>
<tr>
<td>Baugh-Wooley</td>
<td>32</td>
<td>104</td>
<td>15.029</td>
<td>67.97</td>
</tr>
</tbody>
</table>

For performing $n \times n$ reconfigured multiplication, the sub-word multiplication is performed on the reconfigured regions follows the same procedures in Fig. 1(a). Thus the efficiency can be increased and the latency can be removed by the use of pipelined structures. The multiplier section is pipelined to make the multiplication faster. Also reconfiguration architecture is pipelined with registers to reduce latency.

### 3 Implementation Results and Comparisons

In this study, different 4-bit multipliers are implemented in Verilog HDL and logic simulation is done in HDL Designer and the synthesis is done using Altera Quartus II 7.2. The synthesis results of different 4-bit non-pipelined multipliers are shown in Table 1.

The results show that the Baugh-Wooley multiplier has increased speed since clock period is only 15.861ns. It is also seen that Wallace Tree structures have a little improvement in speed than the Baugh-Wooley multipliers as shown. Pipeline stages further improve the Baugh - Wooley architecture speed. Number of LUTs represents the area required for implementation. The number of LUTs required in Baugh-Wooley architecture is 28 compared to 42 in Booth multiplier giving considerable reduction in area.

Power consumption in Baugh-Wooley multipliers is minimum compared to other conventional multiplier units. So it clears that the signed binary multiplication through Baugh-Wooley multiplication is suited for the reconfigurable multiplier implementation. The improvements in constraint can be used to make Baugh-Wooley multiplier efficient.

---

**Figure 2:** $n \times n$ multiplication: (a) Sub-word $n \times n$ multiplication and (b) Partial Product array diagram.
The fan-out of the different multiplier architectures are also given which directly gives the possibility of the multiplier to form large circuits. This can be extended to the pipelined multiplier architecture also to verify the parameters. Latency and speed are the important factors with pipelining under consideration. The synthesis results of 4-bit pipelined multipliers are shown in Table 2. The pipeline constraint increases the speed of the multiplier architectures considerably with a little increase in power consumption. For the Baugh-Wooley multipliers, the clock period reduces to 15.029ns as a result of pipeline registers implemented. This improves the speed which may reduce due to multiple reconfiguration modes.

The incorporation of the pipeline multipliers thus can be effectively done to make the chip efficiently reconfigurable among the two reconfiguration modes and this work is in progress. The possibility of other reconfiguration constraints is under work and the implementation of the reconfiguration modes according to these constraints are the future work.

4 Conclusion

An efficient reconfigurable fixed-width multiplier to deal with the latency problem is proposed. This paper presents a comparison between various multiplier architectures with area, speed and power as the main constraints. It is observed that the Baugh-Wooley architecture gives optimized values for various constraints and hence suited for reconfiguration. The truncation with rounding, pipelining and sub-word multiplication techniques is used to improve the multiplier characteristics.

References