Design and Implementation of a 2's Complement 16-bit Booth Multiplier

Davi K. S. Leonel
Informatics Inst. - UFRGS
PO Box. 15064
Porto Alegre, RS, Brazil
+55(51)3316-6165
dleonel@inf.ufrgs.br

Vagner S. Rosa
Informatics Inst. - UFRGS
PO Box. 15064
Porto Alegre, RS, Brazil
+55(51)3316-6165
vsrosa@inf.ufrgs.br

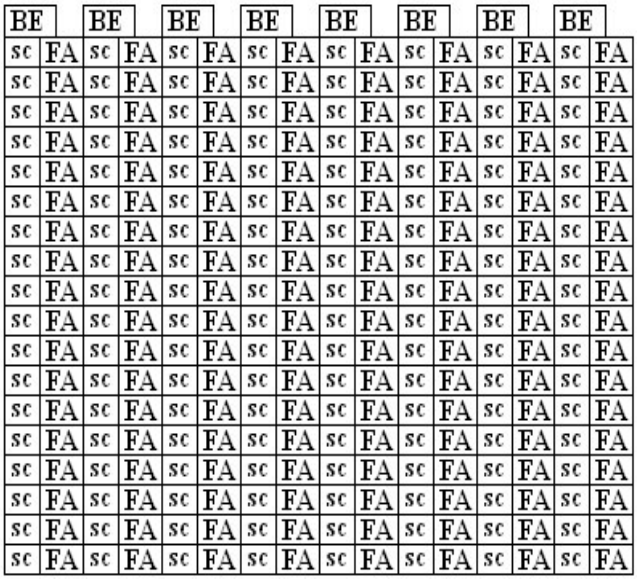
Sergio Bampi
Informatics Inst. - UFRGS
PO Box. 15064
Porto Alegre, RS, Brazil
+55(51)3316-6165
bampi@inf.ufrgs.br

ABSTRACT
A high-performance multiplier is one of the most important blocks of a digital signal processor. This work describes a full custom layout design and implementation of a multiplier, which uses Booth architecture, trying to achieve a balance of area, timing, and power. The objective is to get an IP (intellectual property) for a DSP processor. The Booth architecture is based on an algorithm that reduces the number of partial products to a half when compared to the parallel array multiplier. The design flow began with a detailed study of the Booth algorithm, followed by an architecture description. After the basic blocks were defined, they were grouped hierarchically in order to structure the design. A bottom-up approach was employed and each module was tested and characterized before the upper level of design using LVS (layout versus schematic) and simulations. This circuit was developed in AMS 0.35um CMOS process using CADENCE design tools.

1. INTRODUCTION
The most important operation in DSP applications is multiplication. Several architectures for binary multiplication are present in the literature [3]. The most employed architecture for hardware multiplication is the Booth multiplier, which was realized for 2's complement [2]. The most common Booth implementation uses radix-four for layout regularity and performance [3, 4, 5]. A full custom design and implementation of a radix-four 16-bit Booth multiplier is presented in this work. A hierarchical and spiral model was employed to achieve progressive verification and characterization [1].

2. ALGORITHM STUDY AND DESIGN HIERARCHY
The essence of the Booth algorithm is to label bits as beginning, middle or end of 1's sequence. The radix-four algorithm evaluates two bits at once, analyzing three bits of the multiplier operand, where one bit is shared between successive stages. From this analysis we define the operation of the current stage. This operation can be: addition or subtraction of the multiplicand, which should be multiplied by two or one or zeroed. The first step is to evaluate the operation to be performed. This is done by means of Booth Encoding (BE). Three bits of the multiplier operator are used to define one of the operations described above.
The second step is to perform the operation. The Selector/Complementer (SC) will prepare the multiplicand operator, making zeroing, shifting or negation operation, while the Full Adders (FA) will perform it. The SC uses the three output bits of a BE, one bit from the multiplicand operator and one bit coming from the right bit SC as input. That generates as output one bit of partial product and other bit that goes to the left bit SC.
After the generation of each basic module, the floorplanning is defined for the entire multiplier. Figure 1 shows the floorplanning for the multiplier developed.

Figure 1. Floorplanning of the 16 x 16 booth multiplier



The regularity of this kind of multiplier is clear in this floorplanning, which is straightforward for n-bit multipliers.

3. IMPLEMENTATION
In the implementation, the Cadence Virtuoso tool was used for layout entry. The design followed the spiral model proposed by Boehm for a process execution [6]: objectives determination, alternatives evaluation, product development, and planning for the next iteration.
The basic blocks and their iteration in a 4-bit multiplier were designed in the first loop. In the second loop, a working 16-bit multiplier was obtained. This multiplier operates properly, but with low performance. Three more steps were needed for the design to be considered approved, in particular by refining the FA module.
In the final design, the BE and SC modules were built in static CMOS, while the FA module was built in PTL (Pass-Transistor-Logic). As the FA is the critical point delay of the circuit, several designs were considered, all of them analyzed in the circuit. The best one was employed in the final layout.

In all development phase loops, each basic module layout was simulated using all the input combinations -exhaustive testing- to have assured operation.
Figures 2 and 3 shows the basic blocks and the final design layout for the multiplier developed.

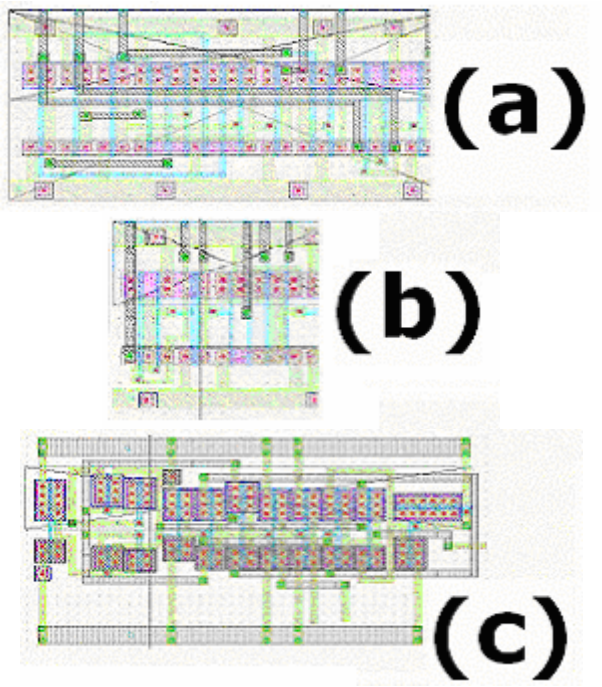


Figure 2. Basic blocks layout. (a) Booth Encoder; (b) Selector/Complementer; (c) Full Adder.

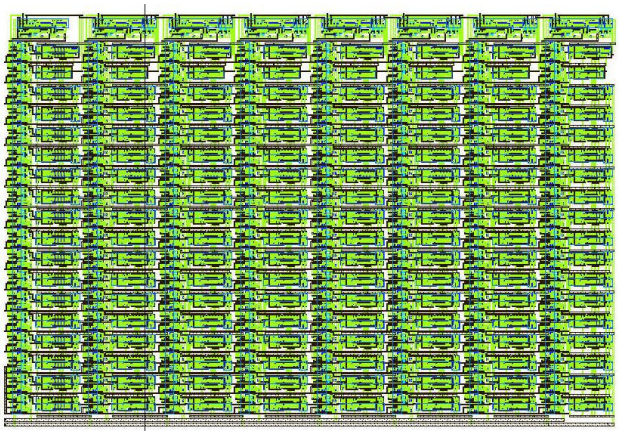


Figure 3. Final design layout

4. RESULTS

Area, timing, and power for the whole multiplier and for each of its basic modules are presented in this section. Table 1 summarizes the results obtained

Table 1. Results at Vdd =3.3V and 25ºC nominal temperature

	N. Transistors	Area (um x um)	Delay (ns)
BE	26	53.4 x 19.8	0.74
SC	16	21.3 x 19.8	0.62
FA	22	44.3 x 23.5	0.42
16-bit multiplier	5990	628.3 x 423.3	12.34

The transistor count was obtained during the layout netlist extraction process, the area was measured in the layout and the delay was measured by simulation, using Cadence Spectre simulator and verifying the worst-case delay output. A topological analysis was performed to the final 16-bit multiplier to identify the longest path input-to-output and the delay in this path was reported. The power consumption of the multiplier, measured using random input vectors with 50MHz frequency, was 30.78 mW.

5. CONCLUSION

The regularity of the multiplier was explored in layout design implementation. The design approach employed during the implementation phase proved to be very effective for this design. The final results are very attractive when compared to similar implementations. A good area optimization was made, shrinking the blank areas within and around the basic modules of the multiplier.

6. REFERENCES

[1] CLEIN, Dan. CMOS IC Layout: Concepts, Methodologies, and Tools, Boston, Newnes, 1999.
[2] HWANG, Kai. Computer Arithmetic: Principles, Architecture and Design. New York, John Wiley & Sons, 1979.
[3] COSTA, Eduardo. da; MONTEIRO J.; BAMPI, Sergio. Operadores Aritméticos de Baixo Consumo para Arquiteturas de Circuitos DSP. Porto Alegre : PGCC da UFRGS, 2002
[4] RABAEY, Jan. Digital Integrated Circuits: A Design Perspective. New York, Prentice Hall, 1996.
[5] WESTE, Neil; ESHRAGHIAN Kamran. Principles of CMOS VLSI Design. Massachusetts, Addison-Wesley, 1995.
[6] Boehm, Barry W. Software risk management. Washington: IEEE Computer Society Press. C1989. 496p.