INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

SIGNED PIPELINED MULTIPLIER USING HIGH SPEED COMPRESSORS

[1]T.Thomas Leonid, [2]M.Mary Grace Neela, and [3]Jose Anand
[1 - 2]Assistant Professor, [3]Associate Professor, Department of Electronics and Communication Engineering
[1 - 3]KCG College of Technology, [2]TJS Engineering College, Chennai, Tamil Nadu, India.

Abstract — In this paper, a unified implementation of signed/unsigned multiplication is proposed using a simple sign control unit together with a line of multiplexers. The proposed approach is demonstrated using CMOS implementation of a 32-bit signed/unsigned multiplier. The results show that the proposed unified signed/unsigned implementation is very compact with only 0.45% silicon area overhead. The critical path delay of the proposed multiplier is about 3.65 ns.

Keywords — signed/unsigned multiplier; Booth encoding; Wallace-tree; fast adder.

1. INTRODUCTION

Both unsigned and signed binary number operation instructions are essential in configurable Digital Signal Processors (DSPs) and special-purpose computers [1]. However, multipliers are designed for either signed or unsigned binary numbers. To the best of our knowledge, the only reported implementations of signed/unsigned multipliers are: (a) a programmable signed/unsigned tree-based architecture for redundant binary arithmetic [2] and (b) a signed/unsigned Booth multiplier using a 2-bit Most Significant Bit (MSB) extension to select the mode of operation [3]. In this paper, we propose a novel programmable signed/unsigned multiplier architecture that compares favourably against prior art in terms of silicon area and power consumption.

Compared with the conventional signed multiplier, the proposed multiplier results in only 0.45 percent silicon area overhead for the implemented 32-bit signed/unsigned multiplier. This is achieved by using three stages with a sign-control unit in the first pipelined stage. In the first stage, Modified Booth Encoding (MBE) [4] is utilized to reduce the Partial Product Rows (PPRs) by half.

Instead of Partial Product Generators (PPGs) based on MBE used in the bit-extension scheme, a line of multiplexers are proposed here to generate a configurable PPR for signed and unsigned modes. The second stage comprises a two-level Wallace-tree compression structure to efficiently sum up PPRs using carry-save adders. The final two partial product rows are processed by a hybrid adder mixed with Conditional Carry Adder (CCA) and Conditional Sum Adder (CSA) based on the MLCSMA algorithm [5].

The proposed signed/unsigned multiplication scheme was optimized in terms of speed, power consumption and silicon area by: (a) exploring more regular partial product array, (b) developing more efficient compression methods and (c) combining several types of fast adders. This paper is organized as follows. Section II introduces the signed/unsigned algorithm. Section III describes the multiplier architecture and details the VLSI implementation of each of the multiplier

stages. Section IV presents implementation results and compares them with prior art. Finally, a conclusion is given in Section V.

II. SIGNED/UNSIGNED ALGORITHM

The multiplier presents two modes of operations, namely 32-bit 2's complement number operand and unsigned 32-bit binary number operand. Assume the multiplication operation is Y (m) × X (n), where Y (m) and X (n) represent the m-bit multiplicand and the n-bit multiplier respectively. The 2's complement number representation of Y (m) is

$$\begin{aligned} Y(m) &= -y_{m-1} \times 2^{m-1} + \sum_{j=0}^{m-2} y_j \times 2^j \\ &= -y_{m-1} \times (2-1) \times 2^{m-1} + \sum_{j=0}^{m-2} y_j \times 2^j \\ &= -y_{m-1} \times 2^m + \sum_{j=0}^{m-1} y_j \times 2^j \end{aligned} \quad (1)$$

In its unsigned representation, $Y(m)$ can be written as

$$Y(m) = -0 \times 2^m + \sum_{j=0}^{m-1} y_j \times 2^j \quad (2)$$

These two representations can be combined using (m+1) bits:

$$Y(m) = y' \times 2^m + \sum_{j=0}^{m-1} y_j \times 2^j \quad (3)$$

Where, y'is equals to y_{m-1} in the signed mode or equals to zero in the unsigned mode. When radix-4 Booth encoding is used on the multiplier, the expression of the multiplier X (n) in its signed form is the different transformations operate on the intermediate result called, the state.

$$X(n) = -x_{n-1} \times 2^{n-1} + \sum_{i=0}^{n-2} x_i \times 2^i$$

$$= 0 \times 2^{n-1} + \sum_{i=0}^{n/2-1} (-2x_{2i+1} + x_{2i} + x_{2i-1}) \times 2^{2i} \quad (4)$$

where $x_{-1} = 0$. The n-bit unsigned representation of $X(n)$ can also be expressed as

$$X(n) = \sum_{i=0}^{n-1} x_i \times 2^i$$

$$= x' \times 2^{n-1} + \sum_{i=0}^{n/2-1} (-2x_{2i+1} + x_{2i} + x_{2i-1}) \times 2^{2i} \quad (5)$$

where $x_{-1} = 0$, $x' = x_{n-1}$ and $x_{n-1} = 0$ in equation (5).

Equation (5) also can represent equation (4) in the condition where $x_{-1} = x' = 0$.

According to equation (3) and equation (5), we attain

$$X(n) \times Y(m) = x' \times 2^{n-1} \times Y(m) + X(n)' \times Y(m) \quad (6)$$

$$\text{where } X(n)' = \sum_{i=0}^{n/2-1} (-2x_{2i+1} + x_{2i} + x_{2i-1}) \times 2^{2i}$$

Equation (6) describes both signed and unsigned multiplication. A control unit is used to select the value of x' and y' which defines the type of the operands. A line of multiplexers is used to implement the first term x'× 2n × Y (m) in equation (6).

III. VLSI IMPLEMENTATION

A. Architecture

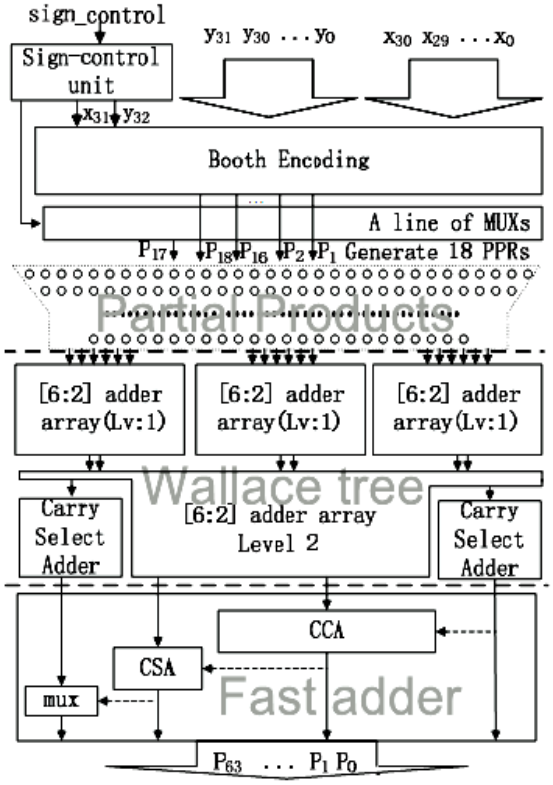

Figure 1 Architecture of the 32-bit signed/unsigned multiplier

The architecture of the proposed 32-bit signed/unsigned multiplier is shown in Figure 1. The sign-control unit generates the MSBs of the multiplier and multiplicand and the select signal for the line of multiplexers. Meanwhile, Modified Booth Encoding (MBE) is used to reduce the number of PPRs by a factor of two.

After generating the PPRs, Wallace-tree structures are used to efficiently add-up all PPRs in parallel. More specifically [3:2], [4:2], [5:2] and [6:2] adders are combined to sum up all the PPRs until only two rows are left. Carry Select adders are inserted in the second stage to reduce the third-stage long-length fast adder's delay, area and power without delay overhead. In the last step, a fast carry-propagation adder is used to add the final two PPRs.

The final adder is characterized by the fact that the input signals do not arrive simultaneously as a result of the Wallace tree compression. Ordinary single carry-propagation adder designs that assume all the inputs arrive simultaneously. A full adder combining both CSA and CCA is developed in the last stage [6].

B. Sign-control Unit

The control unit determines whether the multiplier operates on signed or unsigned numbers. This reconfigurability results in a negligible 0.45% silicon area overhead. Figure 2 shows the building blocks of the control unit. The first two AND gates are used to pre-process the operands' MSBs and generate the correct bit value for the signed or unsigned operands. The third AND gate makes the control signal for the extra 17th partial product row.

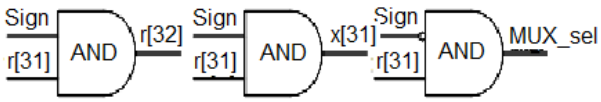

Figure 2 Building blocks of the sign control unit

Figure 3 compares the bit-extension scheme circuit with the proposed mux-based scheme circuit to generate the extra 17th partial product row. The circuit consists of two AND gates and 33 multiplexers while prior art requires a PPG, which includes 35 XNOR gates, 2 XOR gates and 33 OAI (OR-AND-INV) gates. Our approach is thus not only more compact but also faster than the previously reported bit-extension scheme.

C. Modified Booth Encoding

The Modified Radix-4 Booth encoding, proposed in [5], was adopted to balance the critical paths of MBE stage and Wallace-tree. The scheme is detailed in Table I while the Booth encoder and selector circuits, proposed in [5], are shown in Figure 4 (a) and (b) respectively.

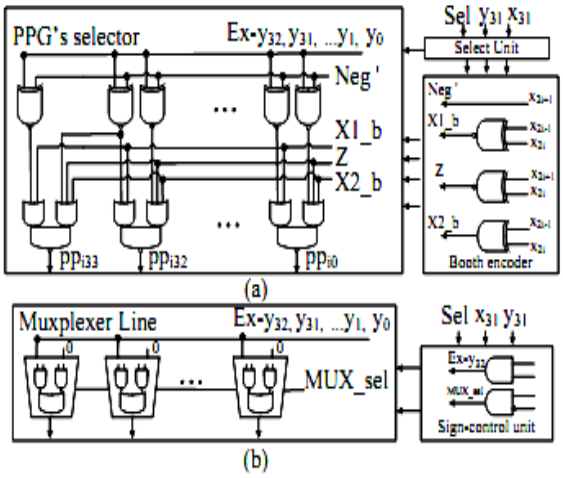

Figure 3 Partial product generation schemes of signed/unsigned Booth multiplier, (a) the bit-extension scheme to generate the 17th partial product row, (b) our mux-based scheme to generate the 17th PPR

Table I

x_{2i+1}	x_{2i}	x_{2i-1}	Op	Neg	X1_b	X2_b	Neg'	Z
0	0	0	+0xY	0	1	0	0	1
0	0	1	+1xY	0	0	1	0	1
0	1	0	+1xY	0	0	1	0	0
0	1	1	+2xY	0	1	0	0	0
1	0	0	-2xY	1	1	0	1	0
1	0	1	-1xY	1	0	1	1	0
1	1	0	-1xY	1	0	1	1	1
1	1	1	-0xY	0	1	0	1	1

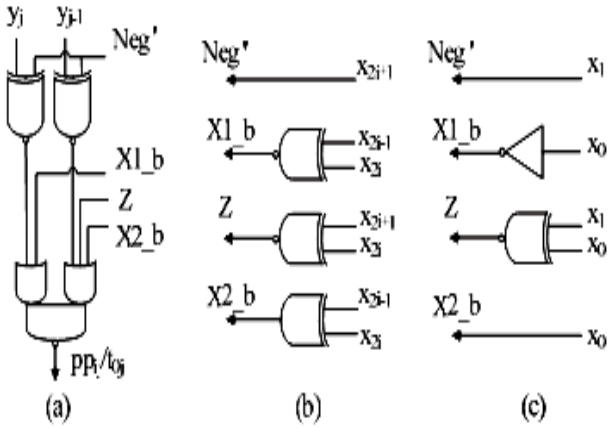

Figure 4 The Encoder and Decoder circuits for New MBE scheme, (a) Decoder; (b) Encoder, (c) Encoder for the last two LSB bits in our design

According to equation (4), the bit x_{-1} is always 0. As a result, the PPR generated by the last two LSB 1 and 0can be simplified to the circuits in figure 4 (c). The extra partial product bit (Neg') at the LSB position of each partial product row for negative encoding leads to an irregular partial product array and a complex reduction tree. In the conventional MBE scheme [4], the LSB of PPR (LSB) and logic equations have the same bit weight. They are

$$P_{LSB} = \overline{(y_{LSB} + \overline{A}) \cdot (y_{-1} + \overline{x_{2i+1} \oplus x_{2i}} + A} \quad (7)$$

$$Neg = x_{2i+1} \cdot \overline{(x_{2i} \cdot x_{2i-1})} \quad (8)$$

where $A = x_{2i} \oplus x_{2i-1}, y_{-1} = 0$. By pre-computing the sum of P_{LSB} and Neg and manipulating the logical equations

$$[Neg', P_{LSB}'] = Neg + P_{LSB} \quad (9)$$

The logic function is changed to

$$P_{LSB}' = y_{LSB} \cdot (x_{2i} \oplus x_{2i-1}) \quad (10)$$

$$Neg' = x_{2i+1} \cdot \overline{(y_{LSB} + \overline{x_{2i} \oplus x_{2i-1}})(x_{2i} + x_{2i-1})} \quad (11)$$

By using equation (9), the silicon area and speed of the MBE stage was optimized. VLSI implementation of MBE is decreased and the speed for the LSB operation is optimized. Note that all the optimized bits in MBE are generated no later than other conventional partial product bits. Figure 5 is a sample for 8-bit signed/unsigned multiplication using sign-extend protection, optimized Booth encoding scheme in LSB bit and the mux-based signed/unsigned scheme.

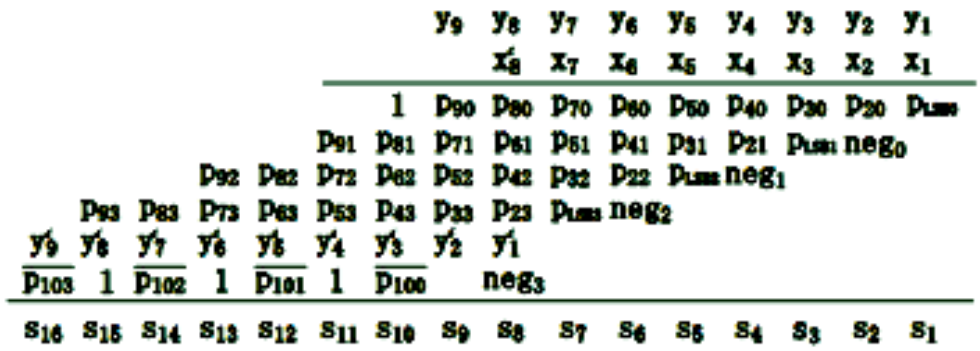

Figure 5 Illustration of an 8-bit signed/unsigned Booth multiplication

D. Partial Product Reduction

Traditionally, half and full adders, organized in a carry save adder format, have been used in the partial product reduction process. However, since their inception by Wein Berger [7], [4:2] adders have become a topic of significant research in the arithmetic community. It has transformed the standard frame of mind of counter for partial product reduction by introducing the notion of horizontal data path within stages of reduction. Furthermore, optimized [5:2] [6:2], [7:2] adders are proposed to improve the architecture of partial product reduction process. The optimized [5:2][6:2] adders shown in Figure6 are adopted in this design.

The MBE algorithm typically generates n/2+1 PPRs in-stead of n/2 due to the extra partial product bit (Neg bit) [11]. One more PPR is needed for signed/unsigned configurations in our multiplier. Instead of using [11] to reduce the number of PPRs, all the MSBs of PPR for sign-protection scheme are grouped with the Neg bit to form one more PPR, shown in Fig. 5 for the case of an 8-bit multiplier. As a result, there are 18 PPRs in the 32-bit multiplier presented in this paper. The 18 PPRs are reduced to two rows, as shown in figure 7.

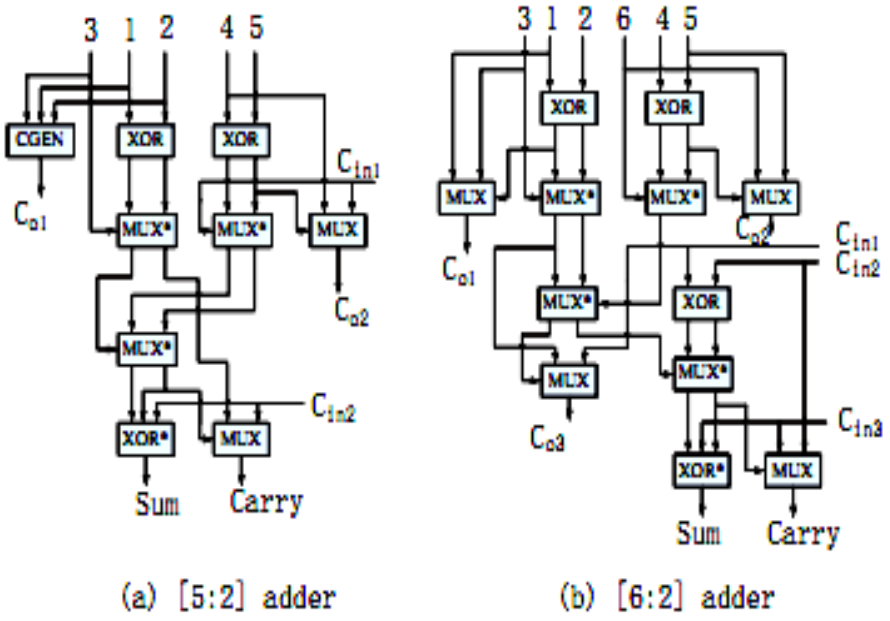

Figure 6 Proposed [5:2] Adder and Optimized [6:2] Adder, (a) [5:2] Adder, (b) [6:2] Adder

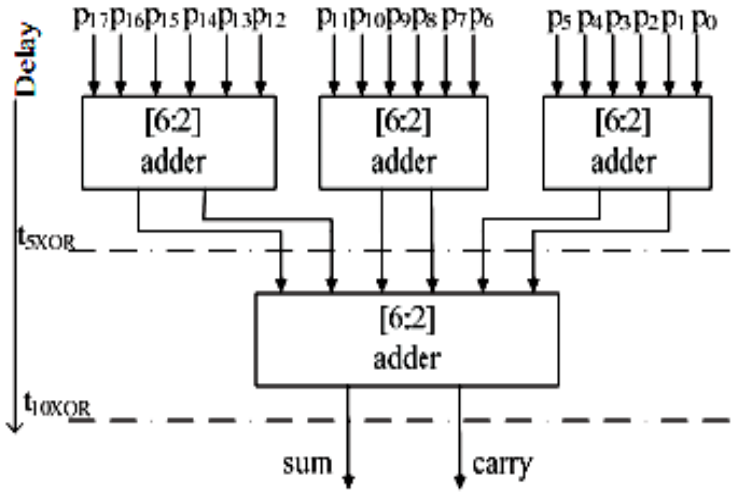

Figure 7 Tree-base Compression Scheme

E. Final Fast Addition

CSA, CCA and Carry Look-Ahead Adder (CLAs) can be used to implement the final fast addition. CLA is widely used and can be easily implemented in dynamic domino CMOS logic with the limitation of full-custom design. For standard static CMOS circuit, CCA and CSA [6] are preferred and can easily be implemented using a standard cell library.

In contrast to the CSA, CCA needs to use XOR logic to produce the final results. This translates in more delay as compared to a same bit-width CSA. The CSA needs to store both the conditional sum and carry together. As a result, more multiplexers are used than for a CCA. To combine the benefits of both adders, a mixed CSA-CCA architecture was implemented to compute a final fast addition. Figure 8 shows the last-stage architecture of a 32-bit CCA followed by a 16-bit CSA, which has the same performance than a 48-bit CSA because the carry-out from CCA is used as the 16-bit CSA final select signal. In this situation, 48 bit results could be generated simultaneously.

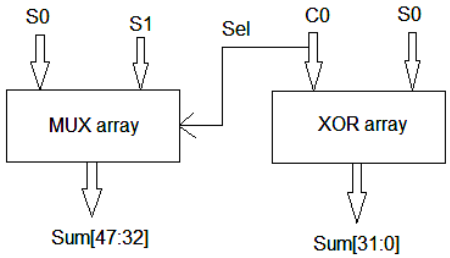

Figure 8 48-bit CCA and CSA Mixed Adder's Final Stage

IV. RESULTS

The multiplier was modeled in Verilog HDL and synthesized using Xilinx 9.1i with a UCF. This Verilog code was simulated using the Modelsim-6.3. Final multiplication of the multiplicand and multiplier was verified. The shortest time required for multiplication is 3.65 ns from Xilinx-9.1i.

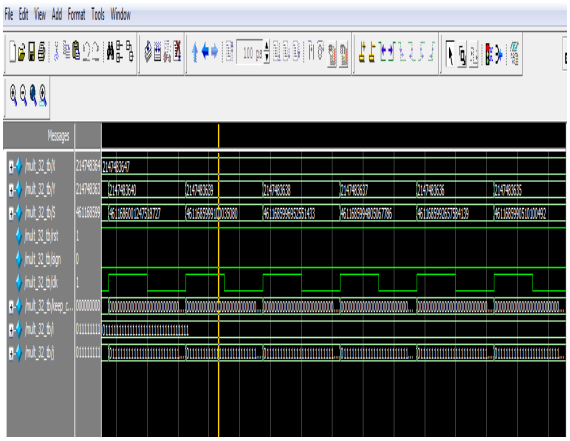

Figure 9 Modelsim Verilog Output for Unsigned Multiplication

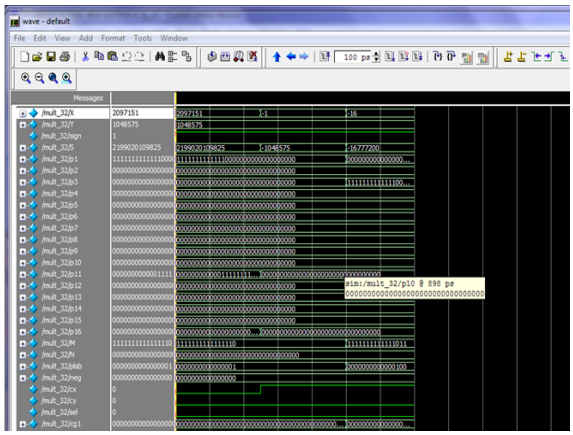

Figure 10 Modelsim Verilog Output for Signed Multiplication

The circuit occupies 642 m×636 m for a standard-cell design using auto-placing and routing tools. We have also implemented the multiplier using a full-custom design flow. The corresponding full-custom design occupies 360 m × 900 m.

Figure 9 shows the total latency "final" and the latency "sum" and "carry" produced before the final addition for all bits from 0 to 63. With the proposed multiplier architecture, bits 30 to 64 come out almost simultaneously while bits 0 to12 come out slower than for the case of a 64-bit CLA. This is explained by the fact that we used a carry-select adder to reduce area and power consumption where delay is not as critical.

Table II compares the performance of the proposed multiplier against recent implementations [10] [11]. The proposed multiplier achieves a delay as small as 3.65ns because registers are used for pipelining. This translates in a relatively larger silicon area. The power dissipation is also improved by optimizing MBE stage's logic function and balancing the signal paths of tree-based parallel compression stage.

Table II Performance Comparison of 32-bit Multiplier for the Cell-based Design Methodology

Cell-based	Delay (ns)	Power (mW)		Area (μm^2)
		@50 M	@100 M	
Paper [10]	7.25	-	40.65	74.6×10^3
Paper [11]	6.99	16.80	-	103.6×10^3
Signed/unsigned Multiplication	3.65	10.20	21.17	132.8×10^3

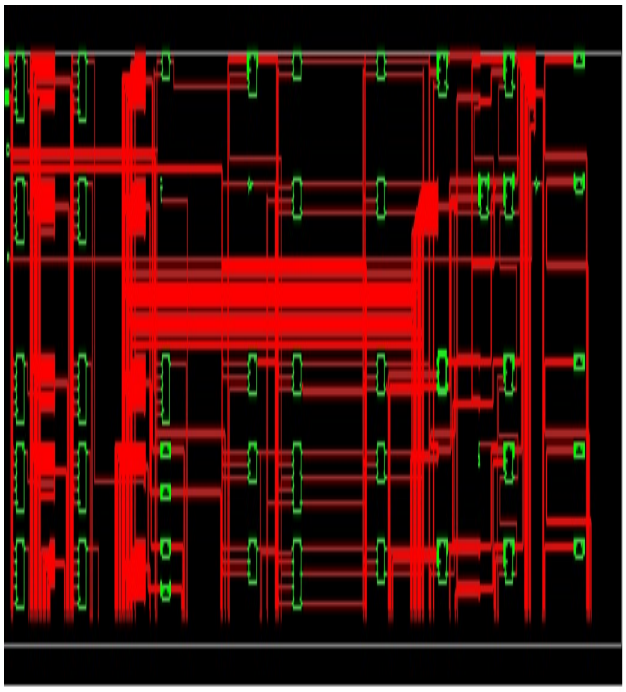

Figure 11 Synthesis diagram of multiplier from Xilinx 9.1i

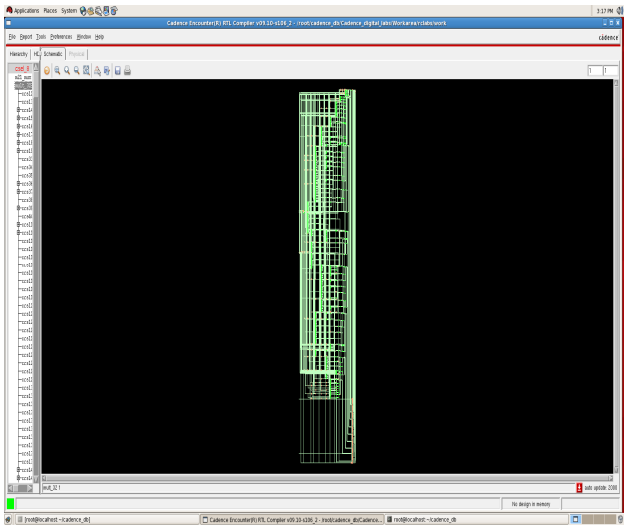

Figure 12 Synthesis diagram of multiplier from cadence

V. CONCLUSION

In this paper, we present a 32-bit×32-bit pipelined multiplier capable of carrying out both signed and unsigned operations. The proposed novel unified signed/unsigned multiplication scheme requires only a simple sign-control unit together with a line of multiplexers, resulting in only 0.45% silicon area overhead in a 0.18 m CMOS process. The critical path delay of the proposed multiplier is about 3.65ns.The signed/unsigned multiplier was optimized in terms of speed, power consumption and silicon area by exploiting more regular partial product array, developing more efficient compression methods and combining several types of fast adders.

REFERENCES

[1] A. Bermak, D. Martinez, J. L. Noullet, "High-density 16/8/4-bit Configurable Multiplier", IEEE Proceedings on Circuits, Devices and Systems, Vol. 144, pp. 272-276, Oct. 1997.
[2] G. Wang, "A Unified Unsigned/signed Binary Multiplier", The 38th Asilomar Conference on Signals, Systems and Computers, Vol. 1, pp. 513 - 516, Nov 7-10, 2004.
[3] J. Y. Kim, "Multiplier to Selectively Perform Unsigned Magnitude Multiplication or Signed Magnitude Multiplication", US patent 5, 870, 322, Feb 9, 1999.
[4] J. Fadavi-Ardekani, "M×N Booth Encoded Multiplier Generator using Optimized Wallace-trees", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 1, Issue 2, pp. 120- 125, June, 1993.
[5] W. C. Yeh, and C. W. Jen, "High-Speed Booth Encoded Parallel Multiplier Design", IEEE Transactions on Computers, Vol. 49, No. 7, July 2000.
[6] K. H. Cheng, S. M. Chiang, and S. W. Cheng, "The Improvement of Conditional Sum Adder for Low Power Applications", Proceedings of Eleventh Annual IEEE International on ASIC Conference 1998, pp. 131 - 134, 13-16 Sept. 1998.
[7] D. T. Shen, and A. Weinberger, "4-2 Carry-Save Adder Implementation Using Send Circuits", IBM Technical Disclosure Bulletin, Vol. 20, pp. 3594-3597, 1978.
[8] C. H. Chang, J. Gu, and Zhang M, "Ultra Low-Voltage Low-power CMOS 4-2 and 5-2 Compressors for Fast Arithmetic Circuits", IEEE Transactions on Circuits and Systems, Vol. 51, Issue 10, pp. 1985 - 1997, Oct. 2004.
[9] K. Amir, R. Kaamran, and A. Majid, "A Novel Multiplier for High-speed Applications", Proceedings of IEEE International Conference on SOC 2005, pp. 305 - 308, Sept 25-28, 2005.
[10] Z. J. Huang, M. D. Ercegovac, and J. Cater, "High-performance Low-Power Left-to-Right Array Multiplier Design", IEEE Transactions on Computers, Vol. 54, No. 3, March, 2005.
[11] S. R. Kuang, J. P. Wang, and C. Y. Guo, "Modified Booth Multipliers With a Regular Partial Product Array", IEEE Transactions on Circuits and Systems II: Express Briefs, Vol. 56, Issue 5, pp. 404 - 408, May, 2009.