

# 國立交通大學

資訊科學與工程研究所

## 碩士論文

低功耗可重組固定寬度乘法器之設計與實作



Design and Implementation of Power-Efficient  
Reconfigurable Fixed-Width Multipliers

研究生：涂晉豪

指導教授：范倫達 博士

中華民國九十七年七月

低功耗可重組固定寬度乘法器之設計與實作

Design and Implementation of Power-Efficient Reconfigurable

Fixed-Width Multipliers

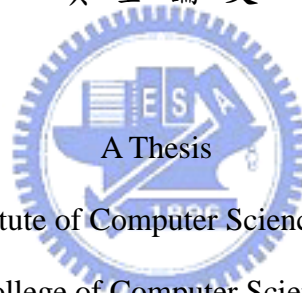
研究生：涂晉豪

Student : Jin-Hao Tu

指導教授：范倫達博士

Advisor : Dr. Lan-Da Van

國立交通大學  
資訊科學與工程研究所  
碩士論文



Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

July 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年七月

# 低功耗可重組固定寬度乘法器之設計與實作

學生：涂晉豪

指導教授：范倫達 博士

國立交通大學  
資訊科學與工程研究所

## 摘 要

在本論文中，我們提出了具有四種運算模式的可重組固定寬度 Booth 乘法器以及可重組固定寬度 Baugh-Wooley 乘法器。此四種運算模式提供了高精準度運算、平行運算、全精準度運算等特性，可因應多種不同的運算需求。根據模擬結果，對於一個 16x16 可重組固定寬度 Booth 乘法器，平均四種運算模式的功率消耗可比 16x16 非重組固定寬度 Booth 乘法器節省 14.0% 的功率消耗。而 16x16 可重組固定寬度 Baugh-Wooley 乘法器亦可節省 12.56% 的功率消耗。另外我們將可重組固定寬度乘法器應用於 FIR 濾波器中，藉此說明可重組固定寬度乘法器具有功率調節的能力，進而達到省電的目的。

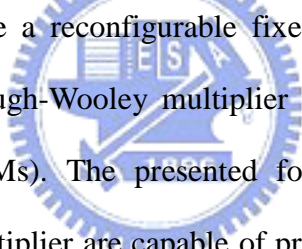
# Design and Implementation of Power-Efficient Reconfigurable Fixed-Width Multipliers

Student : Jin-Hao Tu

Advisor : Dr. Lan-Da Van

Institute of Computer Science and Engineering  
College of Computer Science  
National Chiao Tung University

## ABSTRACT



In this thesis, we propose a reconfigurable fixed-width Booth multiplier and a reconfigurable fixed-width Baugh-Wooley multiplier design framework that provides four configuration modes (CMs). The presented four configuration modes of the reconfigurable fixed-width multiplier are capable of providing high resolution, parallel, and full-precision multiplications for different computation demands. From the simulation results, the proposed 16x16 reconfigurable fixed-width Booth multiplier can attain the power saving of 14.0% on average with respect to that of the 16x16 non-reconfigurable fixed-width Booth multiplier. On the other hand, the proposed 16x16 reconfigurable fixed-width Baugh-Wooley multiplier can save 12.56% power consumption on average in comparison with that of the 16x16 non-reconfigurable fixed-width Baugh-Wooley multiplier. Furthermore, we apply the proposed reconfigurable multiplier to FIR filter to show the power scalable capability under four different modes.

## 誌 謝

首先感謝指導教授 范倫達老師在這兩年多以來的悉心指導與建議，並提供我各方面的協助，使我可以確立並完成我的論文研究。此外，亦要感謝王旭昇學長無私地提供協助，讓我的研究得以順利地進行下去。其次是感謝 VIPLab 實驗室的夥伴們，感謝你們在我的研究生生活中所帶來的溫馨與歡笑。

最後要感謝家人和親友們的關心、支持與鼓勵，尤其是親愛的爸爸媽媽，你們讓我可以無後顧之憂的完成學業。再次感謝以上所有幫助過我的人，謹以此文獻給你們。



---

# Contents

---

摘要.....	I
ABSTRACT.....	II
誌謝.....	III
CONTENTS.....	IV
LIST OF TABLES.....	VI
LIST OF FIGURES.....	VIII
<b>Chapter 1 <i>Introduction</i></b> .....	<b>1</b>
1.1 Motivation.....	2
1.2 Thesis Organization.....	3
<b>Chapter 2 <i>Fundamental Concepts</i></b> .....	<b>4</b>
2.1 Array Multipliers.....	4
2.1.1 Booth Multiplier.....	4
2.1.2 Baugh-Wooley Multiplier.....	7
2.2 Subword Multiplication.....	8
2.3 Low-Error Fixed-Width Multipliers.....	10

<b>Chapter 3</b>	<b><i>Design of Reconfigurable Fixed-width Multipliers</i></b>	<b>13</b>
3.1	Reconfigurable Fixed-Width Booth Multiplier	13
3.1.1	CM1: $n \times n$ Fixed-Width Multiplier	14
3.1.2	CM2: Two $n/2 \times n/2$ Fixed-Width Multipliers	16
3.1.3	CM3: $n/2 \times n/2$ Full-Precision Multiplier	19
3.1.4	CM4: Sum of Two $n/2 \times n/2$ Fixed-Width Multipliers	22
3.1.5	Proposed Structure	22
3.2	Reconfigurable Fixed-Width Baugh-Wooley Multiplier	26
3.2.1	CM1: $n \times n$ Fixed-Width Multiplier	28
3.2.2	CM2: Two $n/2 \times n/2$ Fixed-Width Multipliers	30
3.2.3	CM3: $n/2 \times n/2$ Full-Precision Multiplier	31
3.2.4	CM4: Two $n/4 \times n/4$ Full-Precision Multipliers	32
3.2.5	Proposed Structure	34
<b>Chapter 4</b>	<b><i>Implementation and Comparison</i></b>	<b>44</b>
4.1	Simulation Results of Reconfigurable Fixed-Width Multiplier	45
4.1.1	Reconfigurable Fixed-Width Booth Multiplier	46
4.1.2	Reconfigurable Fixed-Width Baugh-Wooley Multiplier	48
4.2	Application of Reconfigurable Fixed-Width Multiplier	52
<b>Chapter 5</b>	<b><i>Conclusion and Future Work</i></b>	<b>55</b>
<b>Bibliography</b>		<b>56</b>
<b>Biography</b>		<b>60</b>

---

# List of Tables

---

## Chapter 2

- 2.1: Modified Booth recoding table ..... 6

## Chapter 3

- 3.1: Proposed four configuration modes of the reconfigurable fixed-width Booth multiplier ..... 14
- 3.2: Truth table of the decoder for the reconfigurable fixed-width Booth multiplier ..... 26
- 3.3: Truth table of sub-calibration-circuit1 (SCC1) and sub-calibration-circuit2 (SCC2) ..... 26
- 3.4: Proposed four configuration modes of the reconfigurable fixed-width Baugh-Wooley multiplier ..... 28
- 3.5: Truth table of the decoder for the reconfigurable fixed-width Baugh-Wooley multiplier ..... 36

## Chapter 4

- 4.1: Qualitative comparison between different reconfigurable architectures ..... 45
- 4.2: Chip Characteristics of the 8x8 reconfigurable fixed-width Booth multipliers and the 8x8 non-reconfigurable fixed-width Booth multipliers..... 47



---

4.3: Chip Characteristics of the 16x16 reconfigurable fixed-width Booth multipliers and the 16x16 non-reconfigurable fixed-width Booth multipliers .....	<b>48</b>
4.4: Chip Characteristics of the 8x8 reconfigurable fixed-width Baugh-Wooley multipliers and the 8x8 non-reconfigurable fixed-width Baugh-Wooley multipliers .....	<b>50</b>
4.5: Chip Characteristics of the 16x16 reconfigurable fixed-width Baugh-Wooley multipliers and the 16x16 non-reconfigurable fixed-width Baugh-Wooley multipliers .....	<b>50</b>
4.6: Chip Characteristics of the 24x24 reconfigurable fixed-width Baugh-Wooley multipliers and the 24x24 non-reconfigurable fixed-width Baugh-Wooley multipliers .....	<b>51</b>
4.7: Chip Characteristics of the 32x32 reconfigurable fixed-width Baugh-Wooley multipliers and the 32x32 non-reconfigurable fixed-width Baugh-Wooley multipliers .....	<b>51</b>
4.8: Comparison results of error signals and power consumption obtained with the proposed 8x8 reconfigurable fixed-width Booth multiplier for FIR filter application .....	<b>54</b>
4.9: Comparison results among FIR filters .....	<b>54</b>

# List of Figures

## Chapter 2

- 2.1: Block diagram of Booth recoding circuit..... 6
- 2.2: Modified Booth partial-product diagram with sign-generate sign extension scheme for an  $nxn$  multiplier..... 7
- 2.3: Partial-product array diagram for an  $nxn$  Baugh-Wooley multiplier ..... 8
- 2.4: Subword multiplication (a) two  $n/2xn/2$  multiplications, (b) two  $n/2xn/2$  partial-product array distribution, (c) four  $n/4xn/4$  multiplications, and (d) four  $n/4xn/4$  partial-product array distribution ..... 9
- 2.5: The fixed-width  $8 \times 8$  Booth multiplier with  $\theta_{Q=0,w=1}$  .....11
- 2.6: (a) The fixed-width  $8 \times 8$  Baugh-Wooley multiplier with  $\theta_{Q=0,w=1}$ , and (b) logic diagrams of AOR, ANOR, AHA, AFA, NFA..... 12

## Chapter 3

- 3.1: Prototype structure of the proposed reconfigurable fixed-width Booth multiplier involving MUL1, MUL2, and discarding truncated region of LSP ..... 14
- 3.2: (a) Partial-product array diagram for  $nxn$  fixed-width multiplier with  $n=8$ , and (b) configuration parameter settings..... 15

3.3:	Subword operation for two $n/2 \times n/2$ fixed-width multiplications .....	<b>17</b>
3.4:	(a) Partial-product array diagram for two $n/2 \times n/2$ fixed-width multipliers with $n=8$ , (b) configuration settings of $S_{2,4}$ and $S_{3,4}$ , (c) configuration parameter settings, and (d) configured Booth recoding circuit.....	<b>18</b>
3.5:	Subword operation for one $n/2 \times n/2$ full-precision multiplication .....	<b>20</b>
3.6:	(a) Partial-product array diagram for $n/2 \times n/2$ full-precision multiplier with $n=8$ , (b) configuration settings of $S_{2,4}$ and $S_{3,4}$ , (c) configuration parameter settings, and (d) configured Booth recoding circuit.....	<b>21</b>
3.7:	Partial-product array diagram for sum of two $n/2 \times n/2$ fixed-width multipliers with $n=8$ .....	<b>22</b>
3.8:	Overall structure of the proposed reconfigured fixed-width Booth multiplier for $n=8$ .....	<b>25</b>
3.9:	Logical diagram of SCC1 and SCC2 .....	<b>26</b>
3.10:	Prototype structure of the proposed reconfigurable fixed-width Baugh-Wooley multiplier involving MUL1, MUL2, MUL3 and discarding truncated region of LSP .....	<b>27</b>
3.11:	(a) Partial-product array diagram for $n \times n$ fixed-width multiplication, (b) proposed partial-product array diagram using MUL1, MUL2, and MUL3 for CM1, and (c) configuration parameter settings .....	<b>29</b>
3.12:	Subword operation for two $n/2 \times n/2$ fixed-width multiplications .....	<b>31</b>
3.13:	(a) Proposed partial-product array diagram for CM2, and (b) configuration parameter settings.....	<b>31</b>
3.14:	(a) Proposed partial-product array diagram for CM3, and (b) configuration	

parameter settings.....	32
3.15: Subword operation for two $n/4 \times n/4$ full-precision multiplications.....	33
3.16: (a) Proposed partial-product array diagram for CM4, and (b) configuration parameter settings.....	33
3.17: Proposed pipelined reconfigurable multiplier.....	36
3.18: Structure of MUL1 .....	37
3.19: Structure of MUL2.....	37
3.20: Structure of MUL3.....	38
3.21: Logic diagrams of the other processing elements.....	39
3.22: Proposed power-efficient pipelined reconfigurable fixed-width Baugh-Wooley multiplier.....	42
<b>Chapter 4</b>	
4.1: Proposed reconfigurable fixed-width Booth multiplier layout for $n=8$ .....	47
4.2: Proposed pipelined reconfigurable fixed-width Baugh-Wooley multiplier layout for $n=16$ .....	49

# Chapter

# 1

## Introduction

---

During the past decade, the multipliers [1-32] in VLSI signal processing systems stand the test for multimedia-communication applications. Among these multipliers, the basic multiplication either follows Booth [1-3] or Baugh-Wooley algorithms [4]. In many digital signal processing (DSP) algorithms such as digital filters, discrete cosine transform (DCT), and wavelet transform, it is desirable to provide full-precision multiplication [5-8], and fixed-width multiplication [9-20] that produces  $n$ -bit output product with  $n$ -bit multiplier and  $n$ -bit multiplicand with low truncation error. A fixed-width multiplier (also referred to as single precision multiplier) with area and power saving can be achieved either by directly truncating  $n$  least significant columns and preserving  $n$  most significant columns or by other efficient methods [9-20]. By the former method, significant truncation errors will be introduced since no error compensation is considered. Thus, the latter schemes explore issues on low truncation error and small area. Lim [9] first utilized statistical techniques to estimate and simulate the error-compensation bias. However, in his analysis, the reduction and rounding errors are separately treated such that this scheme does not lead to an accurate enough error-compensation bias. Note that the sum of the reduction and rounding errors equals the truncation error [10]. In [10-11], the presented work improved the error-compensation bias to be more accurate and practical since the reduction and rounding errors are concurrently treated. Later, in [12-20], many researchers analyzed

an adaptive error-compensation bias under keeping  $n + w$  most significant columns and proposed various fixed-width multipliers. On the other hand, much work, recently, focuses on constructing reconfigurable full-precision multipliers [21-31]. In [21-26], one reconfigurable full-precision multiplier has been proposed by the subword partitioning technique, where one  $n \times n$ , two  $n/2 \times n/2$ , or four  $n/4 \times n/4$  full-precision multiplications can be performed. In [27-29], a reconfigurable full-precision multiplier consists of an array of  $4 \times 4$  or  $8 \times 8$  small multipliers, where the multiplier introduced in [28] has more configuration modes than that of [27, 29]. The complicated reconfigurable architecture can provide multiple  $4 \times 4$ ,  $8 \times 8$ ,  $16 \times 16$ ,  $32 \times 32$ , and  $64 \times 64$  operations and support unsigned, signed and 2's-complement multiplications. Nevertheless, the architecture [28] led to larger hardware area. The low-power multiplier designs are debated in [30-32]. In [30], a 2-D pipeline gating technique is employed to design a power-aware array multiplier that is adaptive to the high or low resolution operations. In [31], the power cut-off technique is employed to reduce power consumption when lower resolution multiplication is demanded. In [32], a Baugh-Wooley multiplier made use of the dynamic range detection unit and truncated multiplication technique to save power consumption. Nevertheless, the proposed multiplier provided only truncated output precisions under  $n \times n$  truncated multiplication and didn't discuss how to generate the full-precision multipliers and other fixed-width type multiplier.

## 1.1 Motivation

As growing demands on portable computing and communication systems, the power-efficient multiplier plays an important role in very large-scale integration (VLSI) systems. As we know that the conventional reconfigurable multiplier designs [21-31] are based on the full-precision multiplier infrastructure to generate the full-precision multipliers. However, it can be seen that the full-precision multiplier is much more cost ineffective and

power-inefficient than the fixed-width multipliers [19-20]. The fixed-width multiplier and the reconfigurable multiplier are two feasible approaches for the design of power-efficient multipliers. To our best knowledge, we are the first one to explore the power-efficient reconfigurable fixed-width multiplier and discuss how to reconfigure the structure to generate a family of useful fixed-width and full-precision multipliers.

## 1.2 Thesis Organization

The rest of the paper is organized as follows. The Booth multiplier, Baugh-Wooley multiplier, subword multiplication and low-error fixed-width multiplier are briefly reviewed in Chapter 2. In Chapter 3, the proposed reconfigurable fixed-width multiplication engine with four configuration modes is presented. The comparison results in terms of area size and power saving are presented in Chapter 4. For FIR filter application, the error comparison and power scalable performance using various fixed-width and full-precision multipliers are illustrated in the same chapter. Last, brief statements conclude the presentation of this thesis.

# Chapter

# 2

## Fundamental Concepts

In this chapter, the fundamental concepts will be given, including the introduction to the Booth multiplier, Baugh-Wooley multiplier, subword multiplication, and low-error fixed-width multiplier.

### 2.1 Array Multipliers



In this thesis, we will introduce two kinds of reconfigurable fixed-width multipliers. One is based on the Booth multiplier and the other is based on the Baugh-Wooley multiplier. The Booth and Baugh-Wooley multipliers are very famous algorithms used in digital signal processing. In the following, we will briefly review these two algorithms.

#### 2.1.1 Booth Multiplier

Considering two 2's-complement integer operands, we can respectively represent an  $n$ -bit multiplicand  $X$  and an  $n$ -bit multiplier  $Y$  as follows.

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

$$Y = -y_{n-1}2^{n-1} + \sum_{i=0}^{n-2} y_i 2^i \quad (2)$$



where  $x_i, y_i \in \{0,1\}$ . The  $2n$ -bit full-precision product  $P_{FP}$  can be written as

$$P_{FP} = X \cdot Y \quad (3)$$

If  $n$  is even,  $Y$  can be rewritten as

$$Y = \sum_{i=0}^{(n-2)/2} y'_i 2^{2i} \quad (4)$$

where  $y'_i = y_{2i-1} + y_{2i} - 2y_{2i+1}$  and  $y_{-1} = 0$ . The term  $y'_i$  has a value of  $\{-2, -1, 0, 1, 2\}$ .

Each recoded value performs a certain operation on the multiplicand  $x$ ; the multiple additions at each stage are required to generate the product. Substituting (4) into (3), we obtain

$$P_{FP} = \sum_{i=0}^{(n-2)/2} y'_i \cdot X \cdot 2^{2i} = \sum_{i=0}^{(n-2)/2} S_i \quad (5)$$

where  $S_i = y'_i \cdot X \cdot 2^{2i}$ . Triplet scanning takes place from  $y_{-1}$  to the most significant bit (MSB) with a one-bit overlap. Table 2.1 lists the recoding rule and Fig. 2.1 shows the block diagram of the Booth recoding circuit. In Fig. 2.1, the Booth encoder generates three Booth recoding bits  $neg$ ,  $X1$ , and  $X2$  to the Booth selector and then the Booth selector selects input multiplicand  $\{x_j, x_{j-1}\}$  as output partial products. In order to simplify the representation of each partial product, we define the following notation.

$$S_i = S_{i,n-1} 2^{2i+n-1} + S_{i,n-2} 2^{2i+n-2} + S_{i,n-3} 2^{2i+n-3} + \dots + S_{i,0} 2^{2i} \quad (6)$$

where  $s_{i,j}$  represents the  $j$ -th bit product of the  $i$ -th row. In conventional 2's-complement Booth arithmetic operations, the partial product sign extensions are required for each stage, but these extended sign bits lead to large amount of area and power overhead. The sign  $S$  of an  $n$  by  $n$  multiplier can be expressed as

$$S = (S_{0,n} \sum_{j=n}^{2n-1} 2^j) 2^0 + (S_{1,n} \sum_{j=n}^{2n-3} 2^j) 2^2 + (S_{2,n} \sum_{j=n}^{2n-5} 2^j) 2^4 \\ + \dots + (S_{n/2-1,n} \sum_{j=n}^{n+1} 2^j) 2^{2(n/2-1)}$$

$$\begin{aligned}
&= (2^{n+1} + \overline{S_{0,n}} \cdot 2^n) + (2^{n+3} + \overline{S_{1,n}} \cdot 2^{n+2}) \\
&\quad + (2^{n+5} + \overline{S_{2,n}} \cdot 2^{n+4}) + \dots + (2^{2n-1} + \overline{S_{n/2-1,n}} \cdot 2^{2n-2}) + 2^n
\end{aligned} \tag{7}$$

Substituting (6) and (7) into (5), we can obtain the partial-product array diagram for  $n \times n$  Booth multiplier as depicted in Fig. 2.2, where notation  $w$  means to keep  $n+w$  most significant columns of the partial products for fixed-width multiplications. If  $w=n$ , the fixed-width multiplier becomes a full-precision multiplier. In this thesis, we would like to reconfigure the fixed-width multiplication engine to generate several useful multipliers under the limited hardware resource for DSP and image processing applications.

Table 2.1: Modified Booth recoding table

$y_{2i+1}$	$y_{2i}$	$y_{2i-1}$	$y'_i$	$neg$	$X1$	$X2$
0	0	0	0	0	0	0
0	0	1	1	0	1	0
0	1	0	-1	0	1	0
0	1	1	2	0	0	1
1	0	0	-2	1	0	1
1	0	1	-1	1	1	0
1	1	0	-1	1	1	0
1	1	1	0	0	0	0

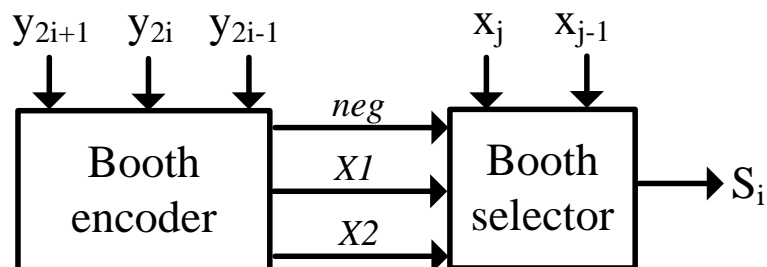


Fig. 2.1. Block diagram of Booth recoding circuit.

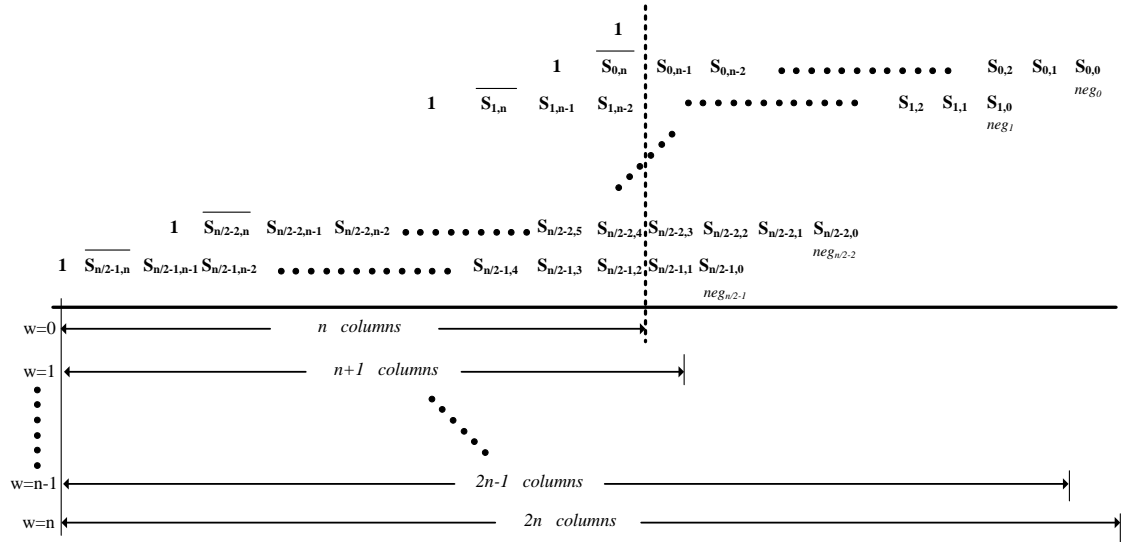


Fig. 2.2. Modified Booth partial-product diagram with sign-generate sign extension scheme for an  $nxn$  multiplier.

### 2.1.2 Baugh-Wooley Multiplier

Considering two 2's-complement integer operands, we can respectively represent an  $n$ -bit multiplicand  $X$  and an  $n$ -bit multiplier  $Y$  as (1) and (2). The  $2n$ -bit full-precision product  $P_{FP}$  can be written as

$$\begin{aligned}
 P_{FP} &= X \cdot Y \\
 &= x_{n-1}y_{n-1}2^{2n-2} + \sum_{i=0}^{n-2} \sum_{j=0}^{n-2} x_i y_j 2^{i+j} \\
 &\quad + 2^{n-1}(-2^{n-1} + \sum_{j=0}^{n-2} x_{n-1} y_j 2^j + 1) \\
 &\quad + 2^{n-1}(-2^{n-1} + \sum_{i=0}^{n-2} y_{n-1} x_i 2^i + 1)
 \end{aligned} \tag{8}$$

Eq. (8) represents the Baugh-Wooley algorithm [4-6] in which this array multiplier sums partial-product bits corresponding to each weighting. The partial-product array for  $nxn$  2's-complement multiplication are depicted in Fig. 2.3, where notation  $w$  means to keep  $n+w$  most significant columns of the partial products for fixed-width multiplications. If  $w=n$ , the fixed-width multiplier becomes a full-precision multiplier.

In this thesis, we would like to reconfigure the fixed-width multiplication engine to generate four useful multipliers under the limited hardware resource.

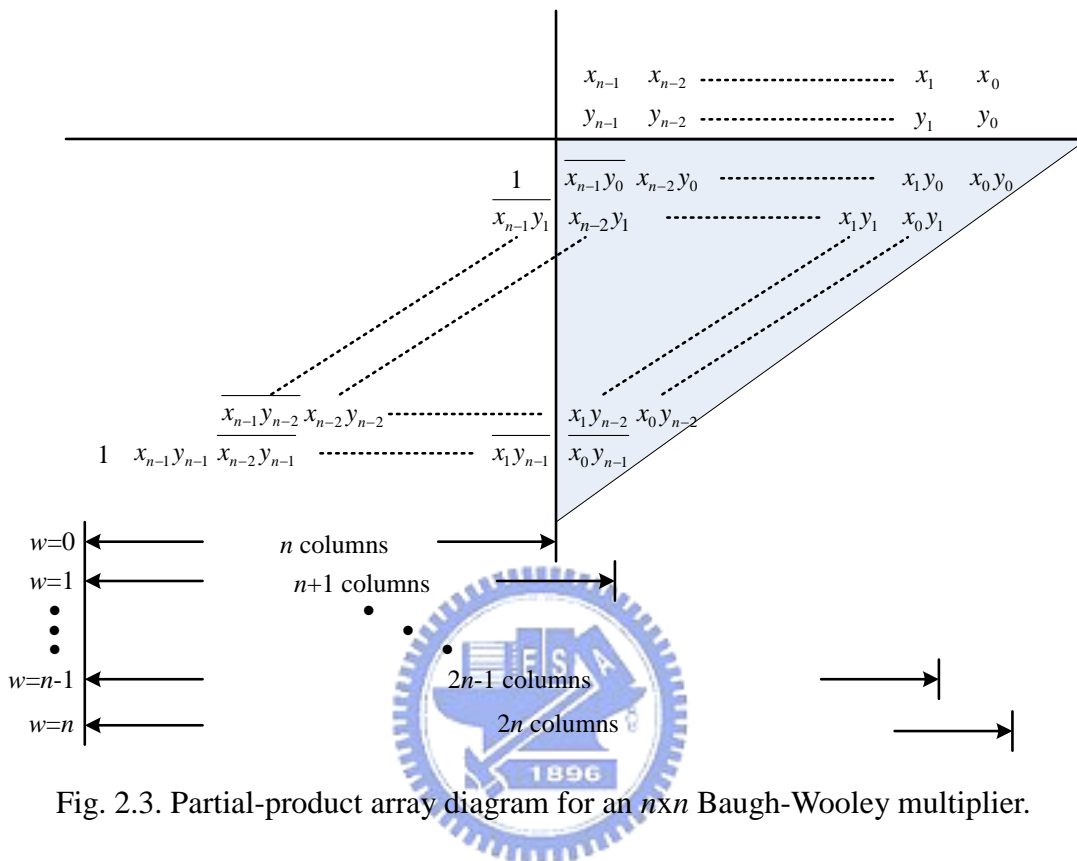


Fig. 2.3. Partial-product array diagram for an  $nxn$  Baugh-Wooley multiplier.

## 2.2 Subword Multiplication

Many DSP and computer applications demand to operate at lower resolution, where the data can be expressed in a half-word length [21-26]. Generally, applying the subword multiplication scheme, we can partition an  $n$ -bit operand into two independent  $n/2$ -bit operands or four independent  $n/4$ -bit operands; hence, the subword multiplier can perform not only  $nxn$  full-precision multiplication but also two  $n/2 \times n/2$  or four  $n/4 \times n/4$  full-precision multiplications in parallel. Fig. 2.4 illustrates subword multiplication and the partial product array distribution [21-26]. In Fig. 2.4(a), two  $n$ -bit operands,  $X$  and  $Y$ , are partitioned into two independent pairs of  $n/2$ -bit subwords, and

then the two pairs of  $n/2$ -bit subwords are multiplied to produce two independent  $n$ -bit products:  $P_1=X_1Y_1$  and  $P_0=X_0Y_0$ , where the partial product array distribution is addressed in Fig. 2.4(b). On the other hand,  $n/4 \times n/4$  subword multiplication and the partial product array distribution are illustrated in Fig. 2.4(c) and 2.4(d), respectively. To our best knowledge, the current subword scheme is applied only to full-precision multiplication based on the full-precision multiplier infrastructure. In the following section, we will extend this subword scheme to fixed-width and full-precision multiplication using the fixed-width prototype multiplier.

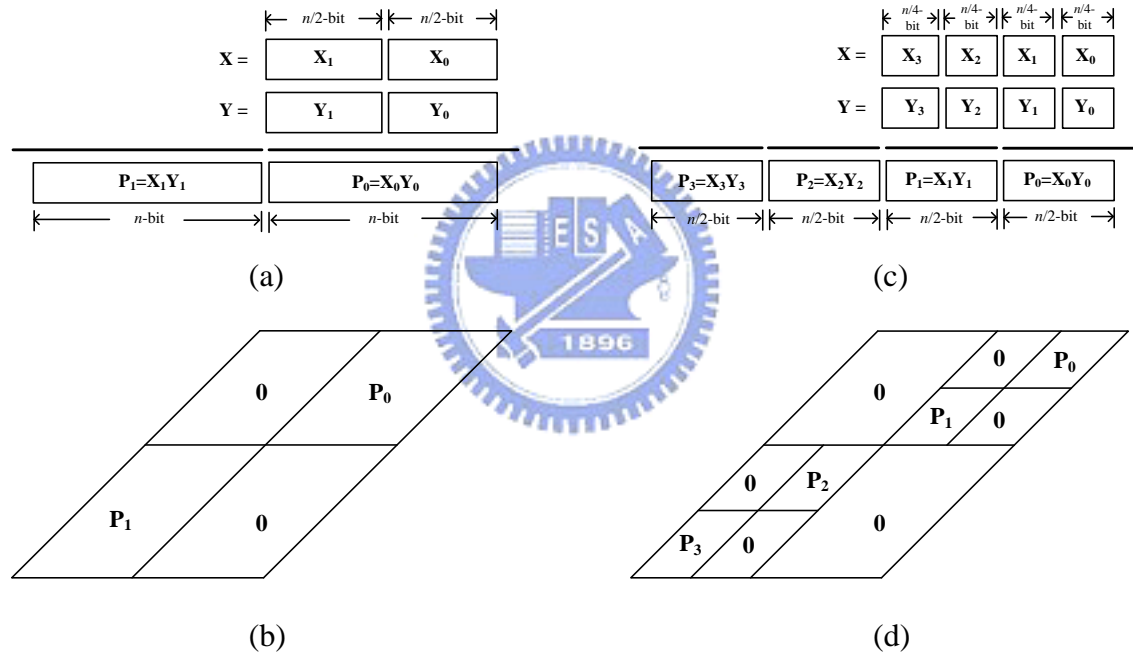


Fig. 2.4. Subword multiplication (a) two  $n/2 \times n/2$  multiplications, (b) two  $n/2 \times n/2$  partial-product array distribution, (c) four  $n/4 \times n/4$  multiplications, and (d) four  $n/4 \times n/4$  partial-product array distribution.

## 2.3 Low-Error Fixed-Width Multipliers

It is known that the various fixed-width multipliers with adaptive compensation biases have been widely discussed in [12-20]. Herein, regarding the tradeoffs of the

truncation error and area cost in [19-20], we choose  $w=1$  (i.e., keeping  $n+1$  most significant columns) and  $Q=0$  for the prototype multiplier structure, where  $Q$  has been clearly defined in [19-20]. The error-compensation bias can be summarized as

$$\sigma_{Type1, Q=0, w=1} = \begin{cases} \left[ \frac{1}{2}(E_{main} + \theta_{Q=0, w=1}) + \frac{1}{2} \right], & \text{if } \theta_{Q=0, w=1} = 0 \\ \left[ \frac{1}{2}(E_{main} + \theta_{Q=0, w=1}) + 0 \right], & \text{if } \theta_{Q=0, w=1} > 0 \end{cases} \quad (9)$$

where  $E_{main} = S_{0,n-1} + S_{1,n-3} + S_{2,n-5} + \dots + S_{n/2-2,3} + S_{n/2-1,1}$  (i.e., the  $(n+1)$ th column counted from left to right of Fig. 2.2) and  $\theta_{Q=0, w=1} = S_{0,n-2} + S_{1,n-4} + S_{2,n-6} + \dots + S_{n/2-2,2} + S_{n/2-1,0}$  (i.e., the  $(n+2)$ th column counted from left to right of Fig. 2.2) for the Booth architecture. If the Baugh-Wooley architecture is the basic multiplier,  $E_{main} = \overline{x_{n-1}y_0} + x_{n-2}y_1 + x_{n-3}y_2 + \dots + x_1y_{n-2} + \overline{x_0y_{n-1}}$  (i.e., the  $(n+1)$ th column counted from left to right of Fig. 2.3) and  $\theta_{Q=0, w=1} = x_{n-2}y_0 + x_{n-3}y_1 + x_{n-4}y_2 + \dots + x_1y_{n-3} + x_0y_{n-2}$  (i.e., the  $(n+2)$ th column counted from left to right of Fig. 2.3). Fig. 2.5 and Fig. 2.6(a) are the prototype Booth multiplier structure and the prototype Baugh-Wooley multiplier structure for  $n=8$ , respectively, where A, ND, HA, and FA denote AND gate, NAND gate, a half adder and a full adder, respectively, and the logic diagrams of the other processing elements are depicted in Fig. 2.6(b).

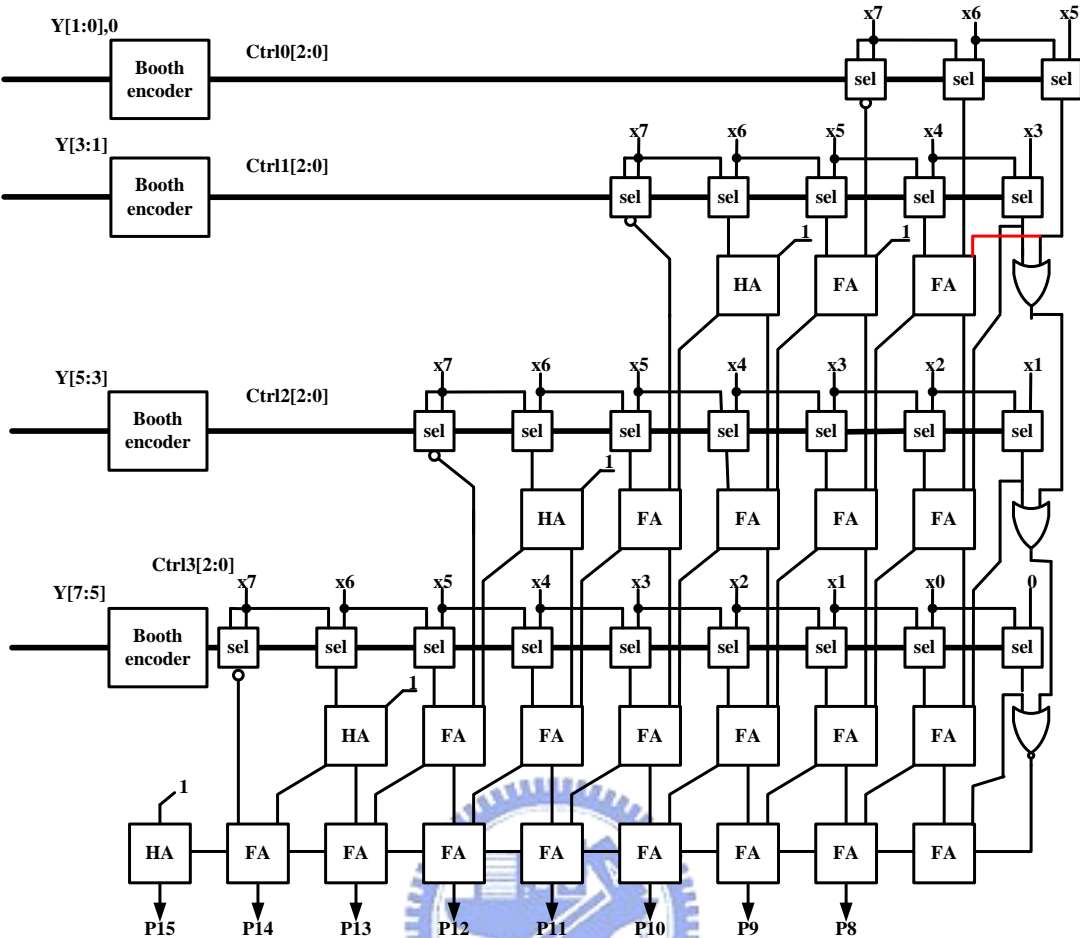
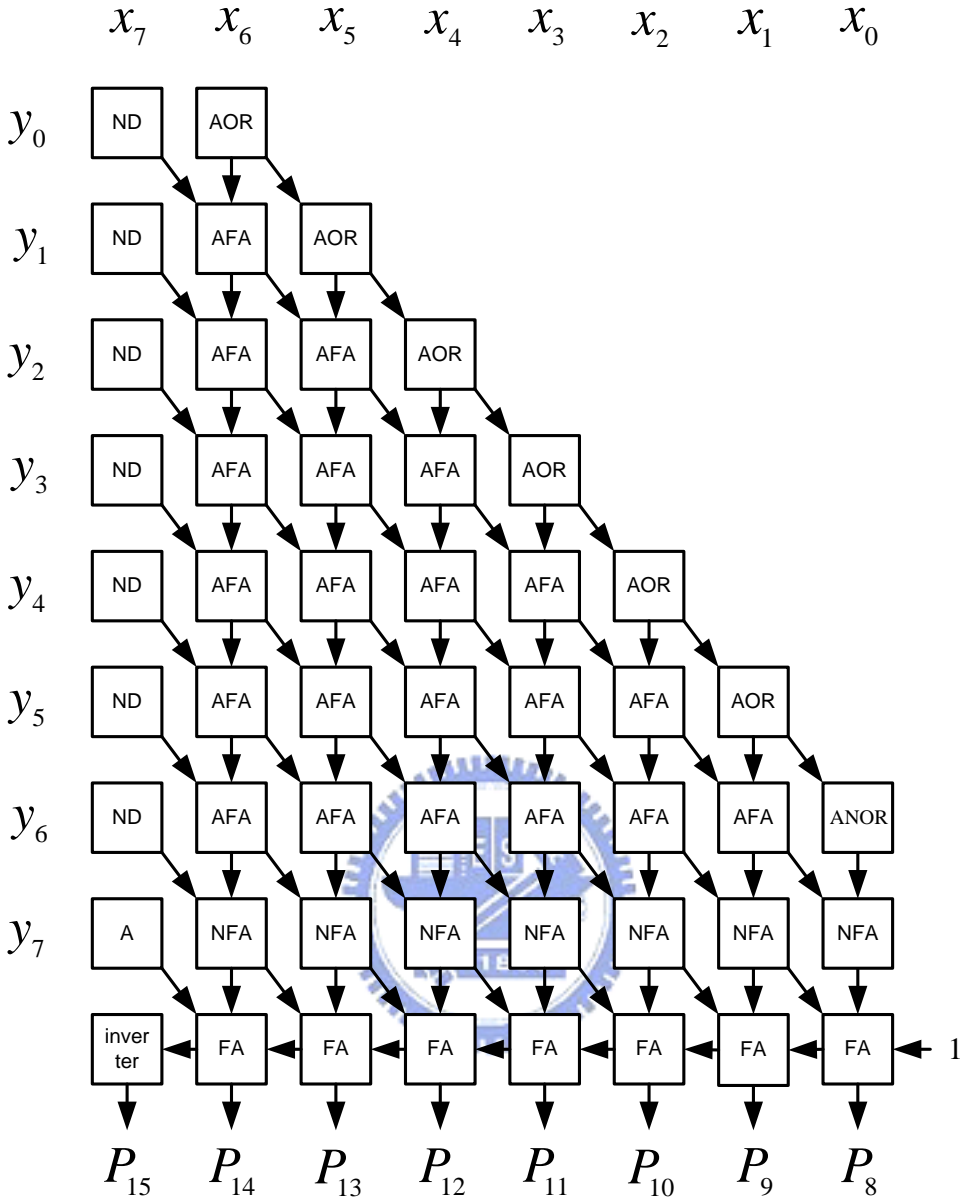
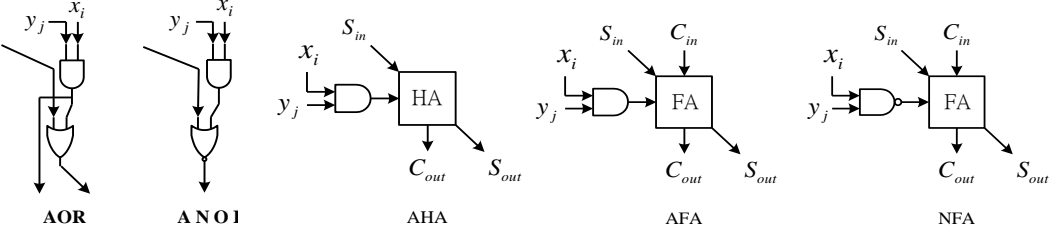


Fig. 2.5. The fixed-width  $8 \times 8$  Booth multiplier with  $\theta_{Q=0,w=1}$ .



(a)



(b)

Fig. 2.6. (a) The fixed-width  $8 \times 8$  Baugh-Wooley multiplier with  $\theta_{Q=0,w=1}$ , and (b)

logic diagrams of AOR, ANOR, AHA, AFA, NFA.



# Chapter

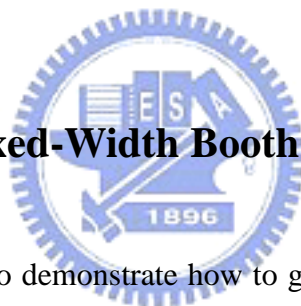
# 3

## Design of Reconfigurable Fixed-Width Multipliers

---

In this chapter, we describe the design methodology of the reconfigurable fixed-width multipliers based on the Booth architecture and the Baugh-Wooley architecture, respectively.

### 3.1 Reconfigurable Fixed-Width Booth Multiplier



In this section, we begin to demonstrate how to generate four different multipliers under the limited hardware resource of the fixed-width Booth multiplier. In this thesis, we use the fixed-width multiplier in Fig. 3.1 as our reconfigurable Booth multiplier prototype instead of the full-precision multiplier structure, where the fixed-width multiplier truncates partial products of the least significant part (LSP) as shown in the dash-lined region of Fig. 3.1 and compensate the error with adaptive compensation bias. In Fig. 3.1, two modules denoted as MUL1 and MUL2 are used to reconfigure the following four different multipliers as listed in Table 3.1 through the corresponding four configuration modes (CMs). Thus, the proposed reconfigurable fixed-width Booth multiplier employing MUL1 and MUL2 is essentially different from the full-precision one [21-31]. Without loss of the generality, we use  $n=8$  to investigate each CM case.

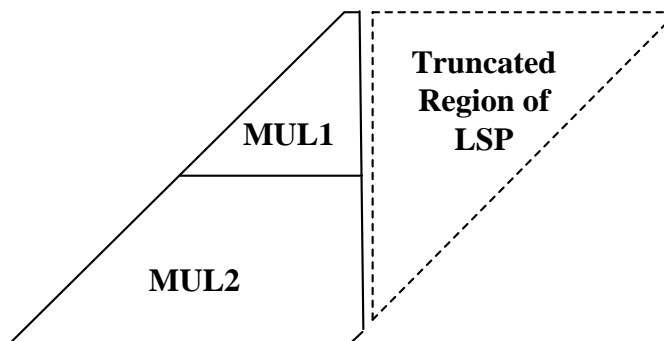


Fig. 3.1. Prototype structure of the proposed reconfigurable fixed-width Booth multiplier involving MUL1, MUL2, and discarding truncated region of LSP.

Table 3.1: Proposed four configuration modes of the reconfigurable fixed-width Booth multiplier

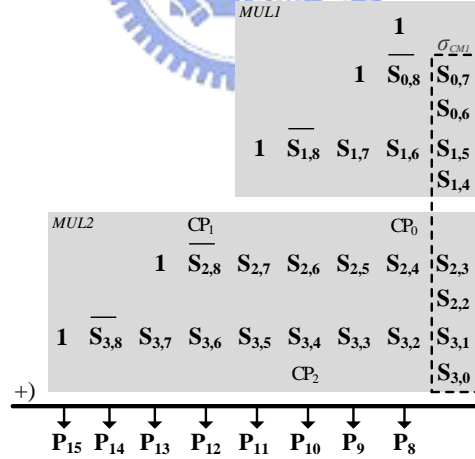
Configuration Mode (CM)	Function Descriptions	Mode Applications
CM1	$n \times n$ fixed-width multiplier	High resolution computations: Multiplication, matrix multiplication, square-root operation, filter, transform
CM2	two $n/2 \times n/2$ fixed-width multipliers	Parallel computations: Multiplication, matrix multiplication, square-root operation, filter, transform
CM3	$n/2 \times n/2$ full-precision multiplier	Full-precision computations: Multiplication, matrix multiplication, square-root operation, filter, transform
CM4	sum of two $n/2 \times n/2$ fixed-width multipliers	Parallel multiplication and add computations: Matrix multiplication, filter, transform

### 3.1.1 CM1: $n \times n$ Fixed-Width Multiplier

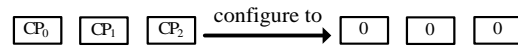
CM1 is in charge of operating  $n \times n$  fixed-width multiplication that receives two  $n$ -bit numbers and produces an  $n$ -bit product. Since CM1 is confined to  $w=1$ , the partial-product array diagram as shown in Fig. 3.2(a) with  $n=8$  can be easily obtained from Fig. 2.2. The partial products in the dash-lined region of Fig. 3.2(a) denoted as  $\sigma_{CM1}$

are used to compute the error compensation bias in (9). Note that the error performance results of the Booth-based CM1 are the same as those of [20] because we implement the same adaptive compensation bias circuits. Throughout the section 3.1, in order to completely achieve four configuration modes, we provide three configuration parameters  $CP_0$ ,  $CP_1$ , and  $CP_2$  combining with the partial product setting to generate four multipliers. For the fixed-width multiplier with  $n=8$ , the bit position of these parameters are shown in Fig. 3.2(a). In CM1,  $CP_0$ ,  $CP_1$ , and  $CP_2$  are set to 0 as shown in Fig. 3.2(b). Associated with  $CP_0$ ,  $CP_1$ , and  $CP_2$ , the product of CM1 can be generally expressed as

$$\begin{aligned}
 P_{CM1} &= \{P_{2n-1}, P_{2n-2}, P_{2n-3}, \dots, P_n\} \\
 &= \sum_{i=0}^{(n-2)/2} \sum_{j=1}^{2i} S_{i,n-j} \cdot 2^{2i+n-j} \\
 &+ (2^{n+1} + \overline{S_{0,n}} \cdot 2^n) + (2^{n+3} + \overline{S_{1,n}} \cdot 2^{n+2}) \\
 &+ (2^{n+5} + \overline{S_{2,n}} \cdot 2^{n+4}) + \dots + (2^{2n-1} + \overline{S_{n/2-1,n}} \cdot 2^{2n-2}) + 2^n \\
 &+ \sigma_{CM1} + CP_0 \cdot 2^n + CP_1 \cdot 2^{\frac{3n}{2}} + CP_2 \cdot 2^{\frac{3n}{2}-2}
 \end{aligned} \tag{10}$$



(a)



(b)

Fig. 3.2. (a) Partial-product array diagram for  $n \times n$  fixed-width multiplier with  $n=8$ , and (b) configuration parameter settings.

### 3.1.2 CM2: Two $n/2 \times n/2$ Fixed-Width Multipliers

CM2 plays a role of concurrently performing two  $n/2 \times n/2$  fixed-width multiplications. In this configuration mode, we need two copies of hardware resource to implement CM2. The corresponding fixed-width subword operation of CM2 is illustrated in Fig. 3.3, where two subword products are  $\mathbf{X}_1\mathbf{Y}_0$  and  $\mathbf{X}_0\mathbf{Y}_1$  because of the limited hardware resource and each fixed-width multiplication has  $n/2$ -bit wide output. Note that we can produce subword products  $\mathbf{X}_0\mathbf{Y}_0$  and  $\mathbf{X}_1\mathbf{Y}_1$  as the conventional subword multiplication by exchanging  $\mathbf{X}_0$  and  $\mathbf{X}_1$  before the Booth selector. However,  $\mathbf{X}_0\mathbf{Y}_0$  and  $\mathbf{X}_1\mathbf{Y}_1$  will lead to larger exchange hardware overhead. Thus, we adopt  $\mathbf{X}_1\mathbf{Y}_0$  and  $\mathbf{X}_0\mathbf{Y}_1$  subword operations. The partial-product array diagram is depicted in Fig. 3.4(a), where  $\sigma_{CM2-1}$  and  $\sigma_{CM2-2}$  denote the error compensation biases of  $\mathbf{X}_1\mathbf{Y}_0$  and  $\mathbf{X}_0\mathbf{Y}_1$ , respectively. In Fig. 3.4(a) with  $n=8$ , compared with CM1, MUL1 can be unchanged while MUL2 must configure  $\{s_{3,4}, s_{2,4}\}$  to  $\{\overline{s_{3,4}}, \overline{s_{2,4}}\}$ , and  $\{s_{3,5}, s_{2,5}\}$  to  $\{1, 1\}$  circled by dash-line and solid-line, respectively. In the logic gate level, we OR the original partial-product with the control signal to generate 1 and use XOR gate to produce inverted sign-bit as shown in Fig. 3.4(b), where  $CS$  denotes the control signal and note that the input  $\{x_4, x_3\}$  are configured to  $\{x_3, x_3\}$ . In addition,  $S_{2,6}$ ,  $S_{2,7}$ ,  $S_{3,6}$ , and  $S_{3,7}$  are set to zero. The configuration parameter settings of CM2 are addressed in Fig. 3.4(c), where  $CP_0$ ,  $CP_1$ , and  $CP_2$  are set to 1, 1, and 0, respectively. Since other partial products are configured to 0, we do not need to AND these partial products with control signal. Our approach is to AND three Booth recoding bits with the control signal as shown in Fig. 3.4(d) to save the number of AND gates while  $n$  increases. Hence, no matter how many partial products needed to be configured to 0, we merely require three AND gates if these partial products are in the same row. In addition, due to permanent inverted MSB output of the partial products in MUL2, the bits denoted as  $\bar{0}$  represent one, but

the summation result of the most significant half of MUL2 still produces zero. In summary, the two products of CM2 can be generally expressed in (11a) and (11b).

$$\begin{aligned}
 P_{CM2-1} &= \{P_{1,n-1}, P_{1,n-2}, P_{1,n-3}, \dots, P_{1,n/2}\} \\
 &= \sum_{i=0}^{(n/2-2)/2} \sum_{j=1}^{2i} S_{i,n-j} \cdot 2^{2i+(n/2)-j} \\
 &\quad + (2^{n/2+1} + \overline{S_{0,n}} \cdot 2^{n/2}) + (2^{n/2+3} + \overline{S_{1,n}} \cdot 2^{n/2+2}) \\
 &\quad + (2^{n/2+5} + \overline{S_{2,n}} \cdot 2^{n/2+4}) \\
 &\quad + \dots + (2^{n-1} + \overline{S_{(n/2-2)/2,n}} \cdot 2^{n-2}) + 2^{n/2} \\
 &\quad + \sigma_{CM2-1}
 \end{aligned} \tag{11a}$$

$$\begin{aligned}
 P_{CM2-2} &= \{P_{2,n-1}, P_{2,n-2}, P_{2,n-3}, \dots, P_{2,n/2}\} \\
 &= \sum_{i=(n/2-2)/2+1}^{(n-2)/2} \sum_{j=(n/2)+1}^{2i} S_{i,n-j} \cdot 2^{2i+(n/2)-j} \\
 &\quad + (2^{n/2+1} + \overline{S_{(n/2-2)/2+1,n/2}} \cdot 2^{n/2}) \\
 &\quad + (2^{n/2+3} + \overline{S_{(n/2-2)/2+2,n/2}} \cdot 2^{n/2+2}) \\
 &\quad + (2^{n/2+5} + \overline{S_{(n/2-2)/2+3,n/2}} \cdot 2^{n/2+4}) \\
 &\quad + \dots + (2^{n-1} + \overline{S_{(n-2)/2,n/2}} \cdot 2^{n-2}) \\
 &\quad + \sigma_{CM2-2} + CP_0 \cdot 2^{n/2} + CP_2 \cdot 2^{n-2}
 \end{aligned} \tag{11b}$$

where  $\{\overline{S_{(n/2-2)/2+1,n/2}}, \overline{S_{(n/2-2)/2+2,n/2}}, \overline{S_{(n/2-2)/2+3,n/2}}, \dots, \overline{S_{(n-2)/2,n/2}}\}$  are the reconfigured partial products as similar to that in Fig. 3.4(b), rather than to complement these partial-products of CM1.

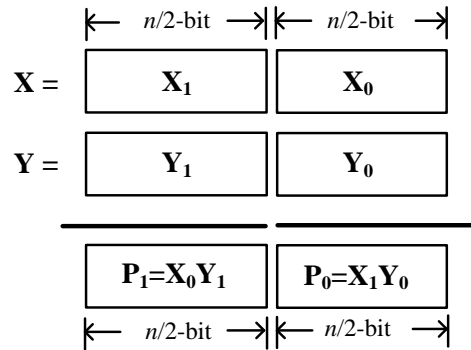
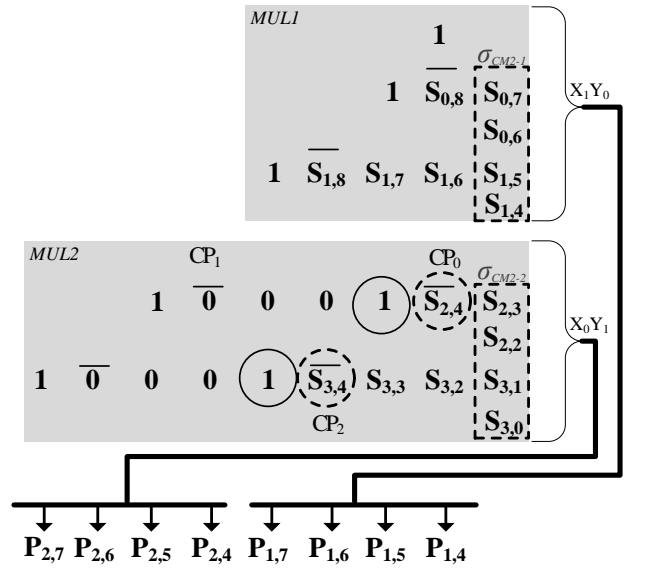
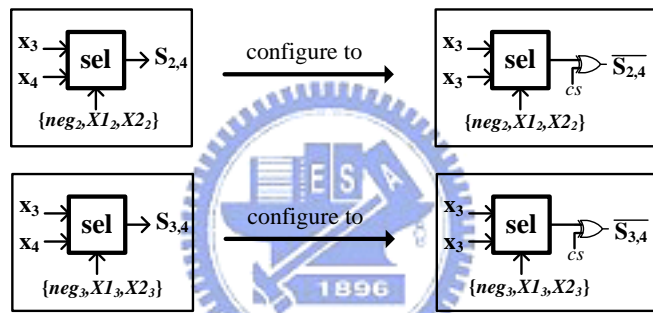


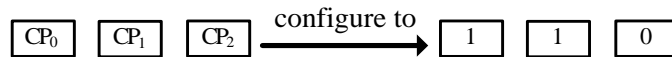
Fig. 3.3. Subword operation for two  $n/2 \times n/2$  fixed-width multiplications.



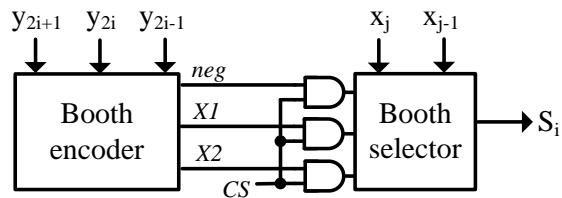
(a)



(b)



(c)



(d)

Fig. 3.4. (a) Partial-product array diagram for two  $n/2 \times n/2$  fixed-width multipliers with  $n=8$ , (b) configuration settings of  $S_{2,4}$  and  $S_{3,4}$ , (c) configuration parameter settings, and (d) configured Booth recoding circuit.

### 3.1.3 CM3: $n/2 \times n/2$ Full-Precision Multiplier

CM3 serves as performing an  $n/2 \times n/2$  full-precision multiplication. The corresponding subword operation of CM3 is illustrated in Fig. 3.5, where the subword product is  $\mathbf{X}_1\mathbf{Y}_1$  with  $n$ -bit wide output. Since the proposed reconfigurable structure to implement full-precision multiplication is based on the fixed-width multiplier fabric, it can be seen that MUL2 is able to achieve this mode operation. Next, the partial-product array diagram is depicted in Fig. 3.6(a) with  $n=8$ , where the configuration of  $S_{2,4}$  and  $S_{3,4}$  circled by dash-line are different from the ones of other modes because their multiplicand inputs of the Booth selector have to be configured from  $\{x_4, x_3\}$  to  $\{x_4, 0\}$  as shown in Fig. 3.6(b). The partial-product  $S_{3,2}$  is configured to  $neg_2$  circled by solid-line for 2's-complement computation. In addition,  $S_{2,2}$ ,  $S_{2,3}$ ,  $S_{3,0}$ ,  $S_{3,1}$ , and  $S_{3,3}$  are configured to 0 in MUL2. In this mode, the output of MUL1 needs to generate zero as shown in the upper diagram of Fig. 3.6(a). The configuration parameter settings of CM3 are addressed in Fig. 3.6(c), where  $CP_0$ ,  $CP_1$ , and  $CP_2$  are set to 0, 1, and  $neg_{n/2-1}$ , respectively. For  $n=8$ ,  $neg_{n/2-1}$  is  $neg_3$ . On the other hand, we AND three Booth recoding bits with the control signals to generate 0 in MUL2 as mentioned in the above paragraph. However, in order to generate 0 in MUL1, we can use AND gates before the Booth encoder to AND the multiplier inputs with the control signal as shown in Fig. 3.6(d) such that the Booth encoder generates 0 before the Booth selector. In summary, the product of CM3 can be generally expressed in (12).

$$\begin{aligned}
 P_{CM3} &= \{P_{n-1}, P_{n-2}, P_{n-3}, \dots, P_0\} \\
 &= \sum_{i=(n/2-2)/2+1}^{(n-2)/2} \sum_{j=1}^{n/2} S_{i,n-j} \cdot 2^{2i-j} \\
 &+ (2^{n/2+1} + \overline{S_{(n/2-2)/2+1,n}} \cdot 2^{n/2}) \\
 &+ (2^{n/2+3} + \overline{S_{(n/2-2)/2+2,n}} \cdot 2^{n/2+2}) \\
 &+ (2^{n/2+5} + \overline{S_{(n/2-2)/2+3,n}} \cdot 2^{n/2+4}) \\
 &+ \dots + (2^{n-1} + \overline{S_{(n-2)/2,n}} \cdot 2^{n-2}) \\
 &+ \sum_{k=(n/2-2)/2+1}^{(n-2)/2-1} neg_k \cdot 2^{2(k-(n/2-2)/2-1)} \\
 &+ CP_0 \cdot 2^0 + CP_1 \cdot 2^{n/2} + CP_2 \cdot 2^{n/2-2}
 \end{aligned} \tag{12}$$

where  $S_{i,n/2}$  are the reconfigured partial products as similar to that in Fig.3.6(b).

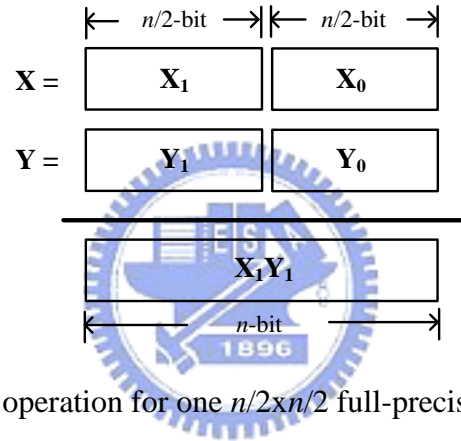
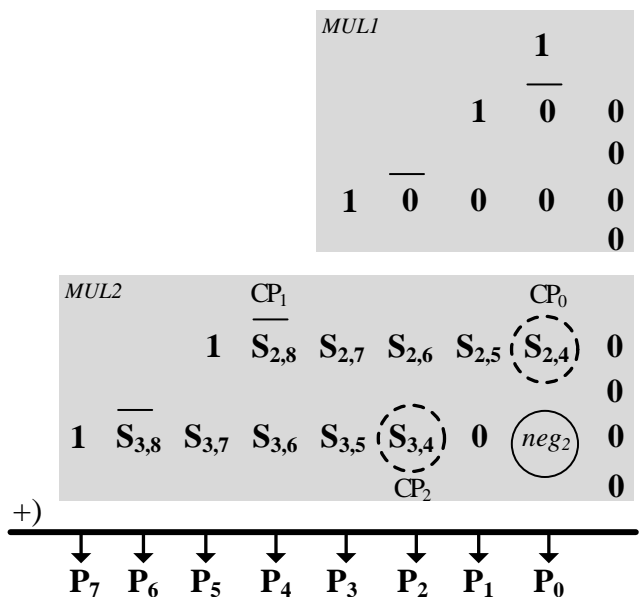


Fig. 3.5. Subword operation for one  $n/2 \times n/2$  full-precision multiplication.





(a)

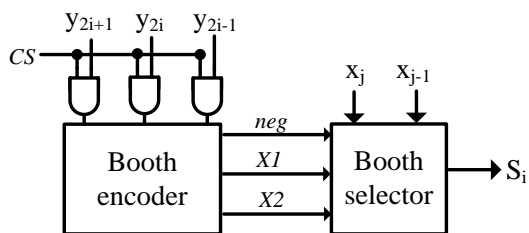
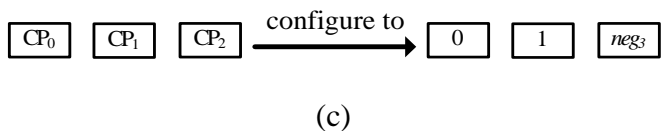
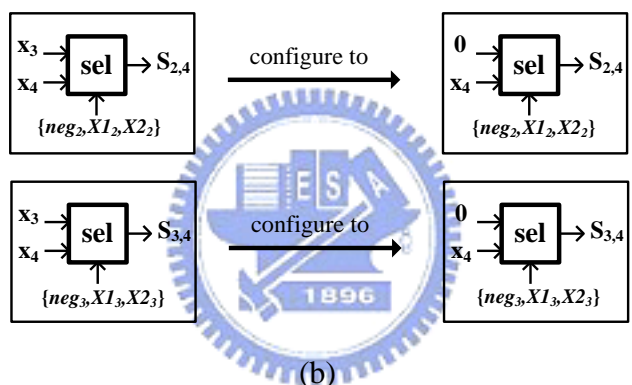


Fig. 3.6. (a) Partial-product array diagram for  $n/2 \times n/2$  full-precision multiplier with  $n=8$ , (b) configuration settings of  $S_{2,4}$  and  $S_{3,4}$ , (c) configuration parameter settings, and (d) configured Booth recoding circuit.

### 3.1.4 CM4: Sum of Two $n/2 \times n/2$ Fixed-Width Multipliers

The main function of CM4 is to add two  $n/2$ -bit wide fixed-width multiplication results. The partial product array diagram is sketched in Fig. 3.7. In Fig. 3.7, the configuration settings are the same as those of CM2; however, the output arrangement is different. The details will be explained in the next paragraph.

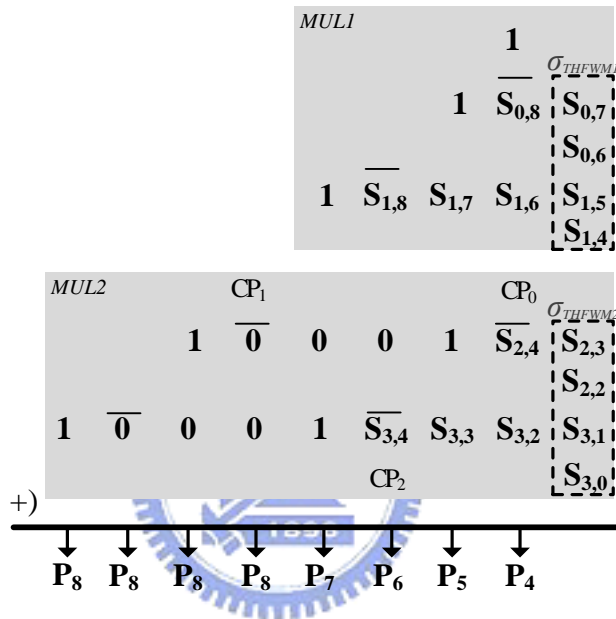


Fig. 3.7. Partial-product array diagram for sum of two  $n/2 \times n/2$  fixed-width multipliers with  $n=8$ .

### 3.1.5 Proposed Structure

According to the above partial-product array analysis of the four configuration modes, we observe the following architecture design viewpoints.

- 1) Need to individually operate MUL1 and MUL2 for CM2. There exists a last-stage adder to sum the outputs of MUL1 and MUL2 to generate product for CM1, CM3, and CM4.
- 2) Need to reconfigure adaptive compensation circuit for  $n \times n$  and  $n/2 \times n/2$  fixed-width multiplication.

- 3) Need to reconfigure Booth encoder circuit for  $n \times n$  and  $n/2 \times n/2$  multiplication.
- 4) Need to control carryout signals according to different CMs.
- 5) Need to rearrange the output of the final product according to CMs.

According to the above viewpoints, the proposed reconfigurable structure for  $n=8$  can be depicted in Fig. 3.8, where we separate the fixed-width multiplication array into two multiplier modules MUL1 and MUL2 and then their outputs are fed into the last-stage adder. First of all, there is one decoder which is charge of decoding OP code to generate control signals for two multiplier modules, where the truth table of this decoder is listed in Table 3.2. After observation, the parameters  $CP_0$ ,  $CP_1$ , and  $CP_2$  circled by dash-line in Fig. 3.8 can be easily realized by  $CS[1]$ ,  $\overline{CS[0]}$ , and  $CS[2] \cdot neg_3$ , respectively. For last-stage adder, since we do not need to sum up MUL1 and MUL2 for CM2, the least significant half inputs of the last-stage adder must be switched from the least significant half products of MUL2 to zero. Thus, the least significant half products of the last-stage adder can directly output the products of MUL1. In Fig. 3.8, we use AND gates to switch the least significant half products of MUL2 to zero. Second, for CM4, sign extension is required before summing up MUL1 and MUL2 so that sign bits are generated after summation. In Fig. 3.8, there are two multiplexers to select sign bits of MUL1 and MUL2, which are denoted as  $P_{MUL1}[11]$  and  $P_{MUL2}[11]$ , respectively. Thus, the most significant half products of the last-stage adder will generate the sign bit denoted as  $S[12]$ .

From viewpoint 2, since  $\sigma_{CM1}$  is composed of  $\sigma_{CM2-1}$  and  $\sigma_{CM2-2}$ , two adaptive compensation biases  $\sigma_{CM2-1}$  and  $\sigma_{CM2-2}$  are needed to carefully control. According to the binary thresholding mentioned in [20], if each adaptive compensation bias adds a constant  $K=1/2$  for  $\theta_{Q=0,w=1}=0$ , the two adaptive compensation biases are not equivalent to the compensation design as shown in Fig. 7 of [20]. Thus, the design will lead to

larger truncation error for CM1 than that of adding a constant  $K=1/2$  one time. Herein, we propose sub-calibration-circuit 1 (SCC1) and sub-calibration-circuit 2 (SCC2) to keep away from double constant addition and to achieve this reconfiguration for  $n \times n$  and  $n/2 \times n/2$  fixed-width multiplications. The logic diagram of SCC1 and SCC2 as shown in Fig. 3.9 is little area overhead, where the truth table of SCC1 and SCC2 is tabulated in Table 3.3. For CM1, if  $K_{m1}=1$  and  $K_{m2}=1$  (i.e.  $\theta_{Q=0,w=1}=0$ ), then  $SCC1=1$  and  $SCC2=0$  to avoid double addition of constant  $K=1/2$ . Otherwise,  $SCC1=0$  and  $SCC2=0$  since  $\theta_{Q=0,w=1} \neq 0$ . For CM2 or CM4, two independent  $n/2 \times n/2$  multipliers are operated in parallel. Thus, SCC1 and SCC2 follow the values of  $K_{m1}$  and  $K_{m2}$  (i.e.,  $SCC1=K_{m1}$  and  $SCC2=K_{m2}$ ).

From viewpoint 3, apart from the above mentioned configurations, the multiplier  $y_{n/2-1}$  has to be configured to 0 when CM2, CM3, or CM4 are performed. In Fig. 3.8, the input of the Booth encoder2 is  $\{Y[5], Y[4], Y[3]\}$  for CM1. On the other hand, while one of other three modes is selected, the input of the Booth encoder2 is  $\{Y[5], Y[4], 0\}$ . In addition, three carryout signals denoted as  $Co_{m1}$ ,  $Co_{m2\_1}$ , and  $Co_{m2\_2}$  are also configured from viewpoint 4.  $Co_{m1}$  is propagated to the last-stage adder only if CM1 is performed.  $Co_{m2\_1}$ , and  $Co_{m2\_2}$  are propagated if either CM1 or CM3 is performed. From viewpoint 5, the output arrangement of the final product is different according to the different modes. As shown in Fig. 3.8, there exists a multiplexer to select the most significant half of the final product. If  $\{CS[1], CS[3]\}=\{0, 0\}$ , that means either CM1 or CM3 is performed. Thus, the output is switched to the most significant half of the last-stage adder (i.e.  $S[15:12]$ ). If  $\{CS[1], CS[3]\}=\{1, 0\}$ , that means CM2 is performed and then the output is switched to the least significant half of MUL2 (i.e.  $P_{MUL2}[11:8]$ ). If  $\{CS[1], CS[3]\}=\{1, 1\}$ , that means CM4 is performed and the output is extended to 8 bits with sign bit  $S[12]$ .

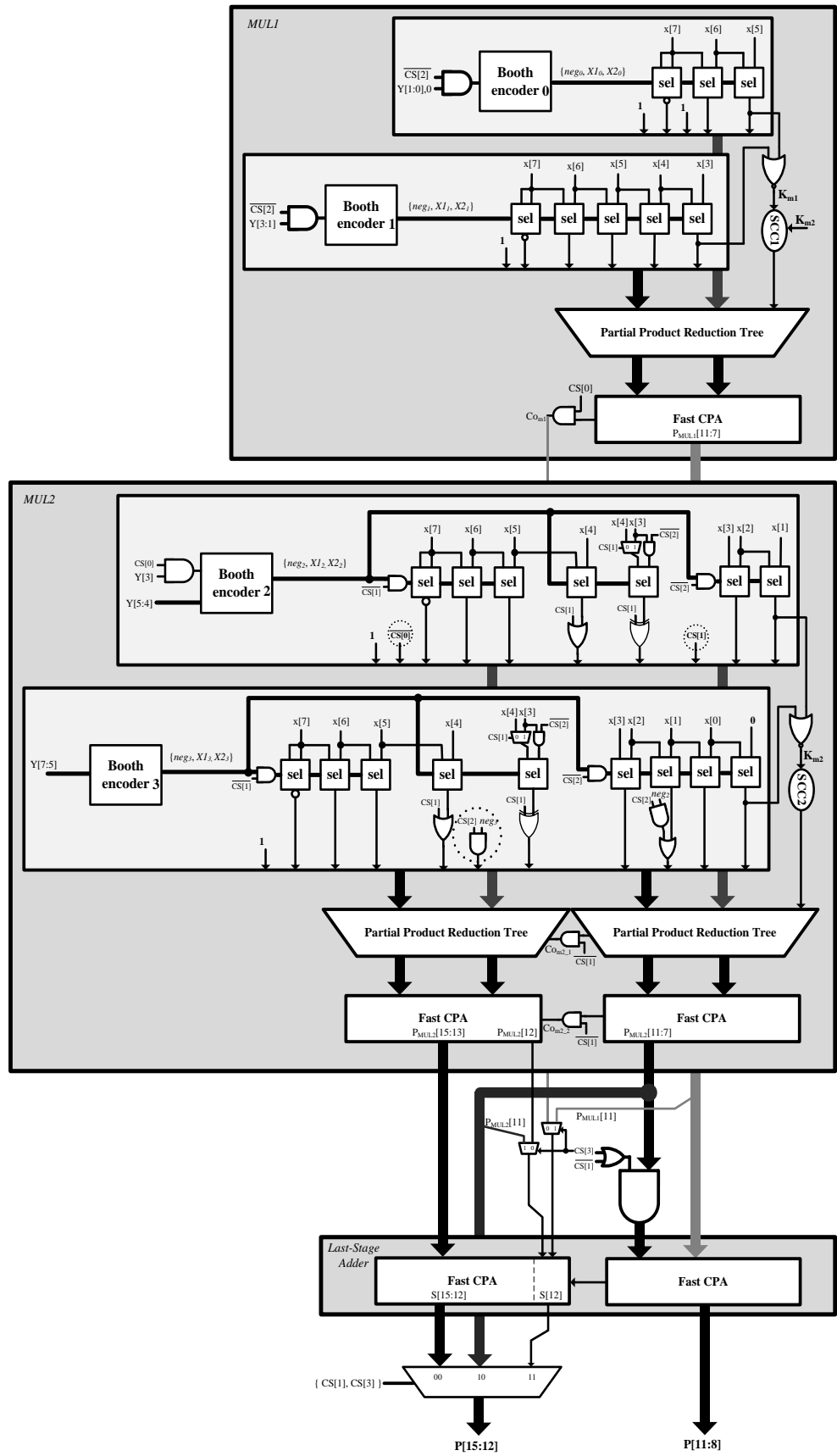


Fig. 3.8. Overall structure of the proposed reconfigurable fixed-width Booth multiplier for  $n=8$ .

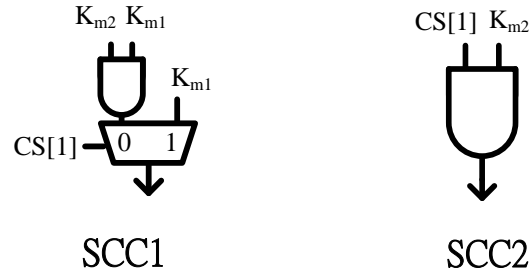


Fig. 3.9. Logical diagram of SCC1 and SCC2.

Table 3.2: Truth table of the decoder for the reconfigurable fixed-width Booth multiplier

OP[2:0]	CS[3:0]			
00(CM1)	0	0	0	1
01(CM2)	0	0	1	0
10(CM3)	0	1	0	0
11(CM4)	1	0	1	0

Table 3.3: Truth table of sub-calibration-circuit1 (SCC1) and sub-calibration-circuit2 (SCC2)

$K_{m1}$	$K_{m2}$	CM1		CM2 or CM4	
		Output of SCC1	Output of SCC2	Output of SCC1	Output of SCC2
0	0	0	0	0	0
0	1	0	0	0	1
1	0	0	0	1	0
1	1	1	0	1	1

### 3.2 Reconfigurable Fixed-Width Baugh-Wooley Multiplier

In this section, we begin to demonstrate how to generate four different multipliers under the limited hardware resource of the fixed-width Baugh-Wooley multiplier. In this

thesis, we use the fixed-width multiplier in Fig. 3.10 as our reconfigurable Baugh-Wooley multiplier prototype instead of the full-precision multiplier structure, where the fixed-width multiplier truncates partial products of the least significant part (LSP) as shown in the dash-lined region of Fig. 3.10. In Fig. 3.10, three modules denoted as MUL1, MUL2, and MUL3 are used to reconfigure the following four different multipliers as listed in Table 3.4 through the corresponding four configuration modes (CMs). Thus, the proposed reconfigurable fixed-width Baugh-Wooley multiplier employing MUL1, MUL2, and MUL3 is essentially different from the full-precision one [21-31]. Without loss of the generality, we use  $n=8$  to investigate each CM case in the following.

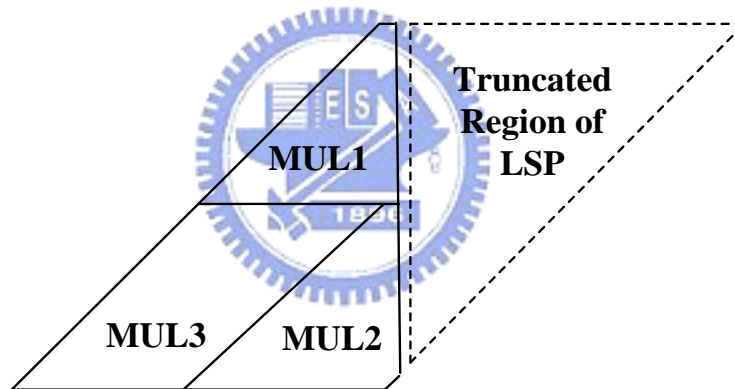


Fig. 3.10. Prototype structure of the proposed reconfigurable fixed-width Baugh-Wooley multiplier involving MUL1, MUL2, MUL3 and discarding truncated region of LSP.

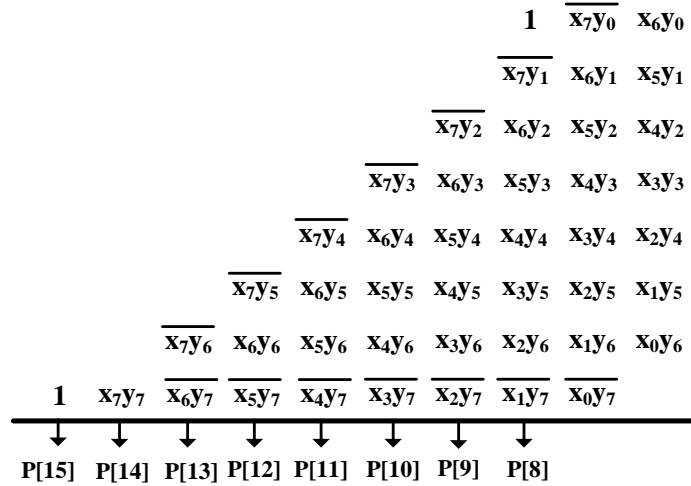
Table 3.4: Proposed four configuration modes of the reconfigurable fixed-width Baugh-Wooley multiplier

Configuration Mode (CM)	Function Descriptions	Mode Applications
CM1	$n \times n$ fixed-width multiplier	High resolution computations: Multiplication, matrix multiplication, square-root operation, filter, transform
CM2	two $n/2 \times n/2$ fixed-width multipliers	Parallel computations: Multiplication, matrix multiplication, square-root operation, filter, transform
CM3	$n/2 \times n/2$ full-precision multiplier	Full-precision computations: Multiplication, matrix multiplication, square-root operation, filter, transform
CM4	two $n/4 \times n/4$ fixed-width multipliers	Parallel computations: Multiplication, matrix multiplication, square-root operation, filter, transform

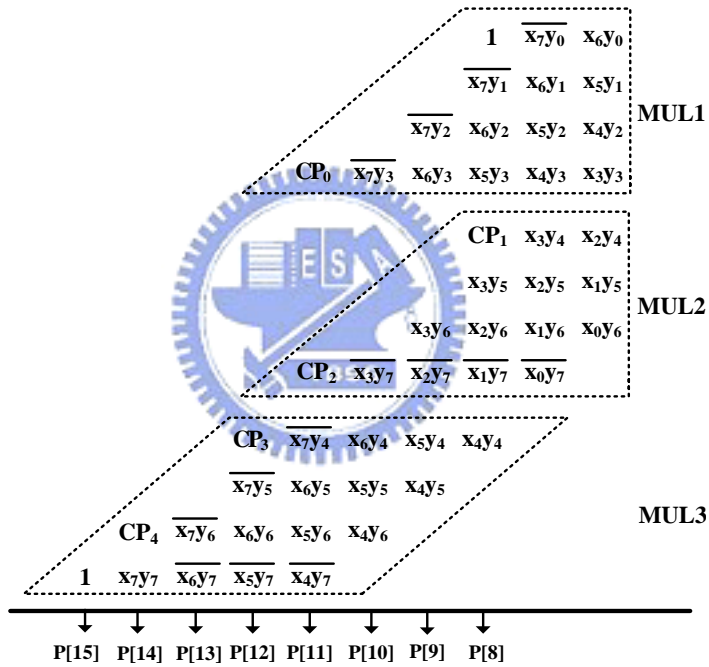
### 3.2.1 CM1: $n \times n$ Fixed-Width Multiplier

CM1 is in charge of operating  $n \times n$  fixed-width multiplication that receives two  $n$ -bit numbers and produces an  $n$ -bit product. It is known that the various fixed-width multipliers with adaptive compensation biases have been widely discussed in [12-20]. Herein, regarding the tradeoffs of the truncation error and area cost in [19], we choose  $w=1$  (i.e., keeping  $n+1$  most significant columns) and  $Q=0$  for the prototype multiplier structure in CM1, where  $Q$  has been clearly defined in [19]. Since CM1 is confined to  $w=1$ , the partial-product array diagram as shown in Fig. 3.11 (a) with  $n=8$  can be easily obtained from Fig. 2.3. As mentioned above in this section, the rest partial products are decomposed into three multiplication modules MUL1, MUL2, and MUL3 as depicted in Fig. 3.11(b). The partial products of the three blocks are summed up independently and then the three summations are added together to produce final product. Throughout the section 3.2, in order to completely achieve four configuration modes, we provide five configuration parameters  $CP_0$ ,  $CP_1$ ,  $CP_2$ ,  $CP_3$ , and  $CP_4$  combining with the proper partial product setting to generate other multipliers. In CM1,  $CP_0$ ,  $CP_1$ ,  $CP_2$ ,  $CP_3$ , and  $CP_4$  are set to 0 as shown in Fig. 3.11(c).

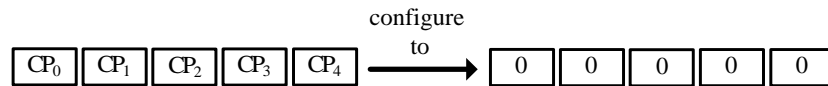




(a)



(b)



(c)

Fig. 3.11. (a) Partial-product array diagram for  $n \times n$  fixed-width multiplication, (b) proposed partial-product array diagram using MUL1, MUL2, and MUL3 for CM1, and (c) configuration parameter settings.

### 3.2.2 CM2: Two $n/2 \times n/2$ Fixed-Width Multipliers

CM2 plays a role of concurrently performing two  $n/2 \times n/2$  fixed-width multiplications. In this configuration mode, we need two copies of hardware resource to implement CM2. First, we have to determine which multiplier modules are suitable for two  $n/2 \times n/2$  fixed-width multiplications under the constraint of the minimum number of modules and partial-product configuration settings. It is manifest that MUL1 and MUL2 are suitable for two  $n/2 \times n/2$  fixed-width multiplications. Due to the use of MUL1 and MUL2, the corresponding fixed-width subword operation of CM2 is illustrated in Fig. 3.12, where two subword products are  $X_I Y_0$  and  $X_0 Y_I$ , and each fixed-width multiplication has  $n/2$ -bit wide output. If we choose MUL3 for  $X_I Y_I$  and either MUL1 for  $X_I Y_0$  or MUL2 for  $X_0 Y_I$ , we can find that it is difficult to implement two input-independent fixed-width multipliers owing to the same  $X_I$  or  $Y_I$ . Even though we can carry out one  $n/2 \times n/2$  fixed-width multiplier from partial products of  $X_I Y_I$ , larger number of configuration parameters is needed. That means lower flexibility and larger numbers of parameter settings are incurred. Once deciding the fixed-width subword product candidates, we can depict the partial-product array diagram using MUL1 and MUL2 in Fig. 3.13(a), where the partial products circled by dot-line are needed to be reconfigured in comparison with CM1. In Fig. 3.13(a), compared with partial products of MUL1 and MUL2 of CM1,  $x_4 y_3$ ,  $x_5 y_3$ ,  $x_6 y_3$ ,  $\overline{x_7 y_3}$ ,  $x_3 y_4$ ,  $x_3 y_5$ ,  $x_3 y_6$ , and  $\overline{x_3 y_7}$  are complemented,  $x_3 y_3$  is configured to zero. The configuration parameters of CM2 can be set as addressed in Fig. 3.13(b), where  $CP_0$ ,  $CP_1$ , and  $CP_2$  are set to 1. The rest partial products are unchanged.

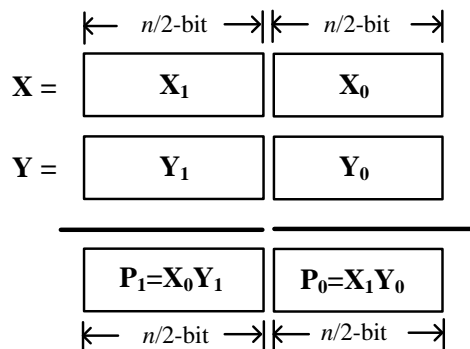


Fig. 3.12. Subword operation for two  $n/2 \times n/2$  fixed-width multiplications.

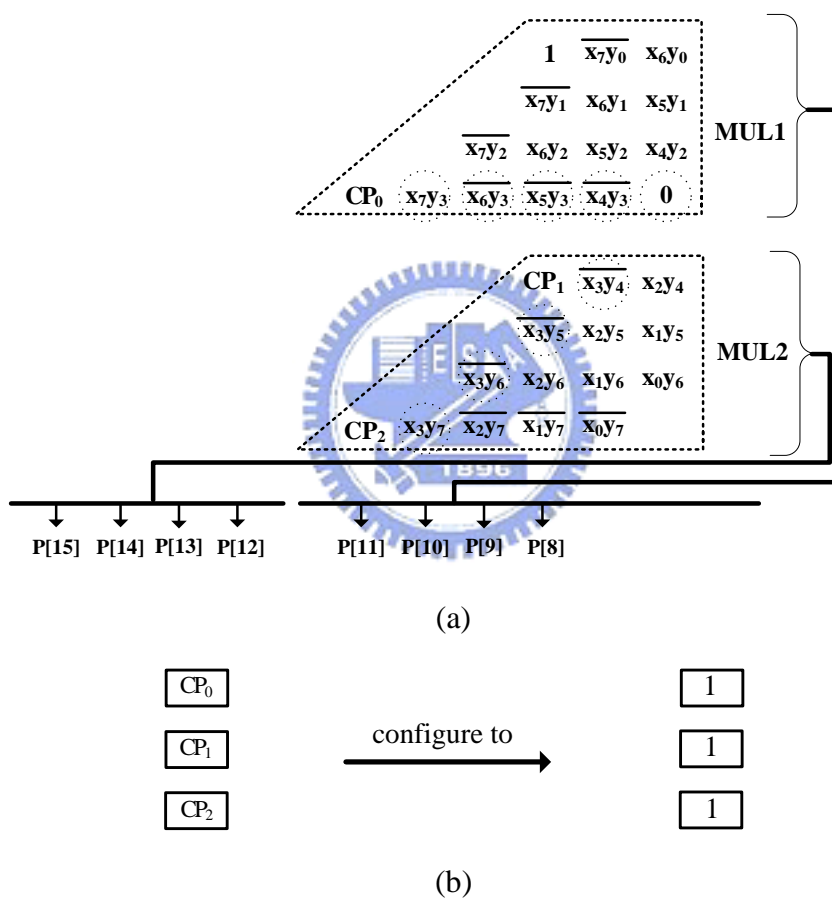


Fig. 3.13. (a) Proposed partial-product array diagram for CM2, and (b) configuration parameter settings.

### 3.2.3 CM3: $n/2 \times n/2$ Full-Precision Multiplier

CM3 serves as performing an  $n/2 \times n/2$  full-precision multiplication. In behavior similar to that in CM2, the design procedures can be stated as follows. First, we have to

determine which modules are suitable for  $n/2 \times n/2$  full-precision multiplications with the minimum number of modules and partial-product configuration settings. Under these constraints, since the proposed reconfigurable structure to implement full-precision multiplication is based on the fixed-width multiplier fabric, we can observe that just only one module, MUL3, can meet. Thus, the partial product array diagram of the MUL3 is depicted in Fig. 3.14, where  $CP_3$  and  $CP_4$  are set to 1 and 0, respectively.

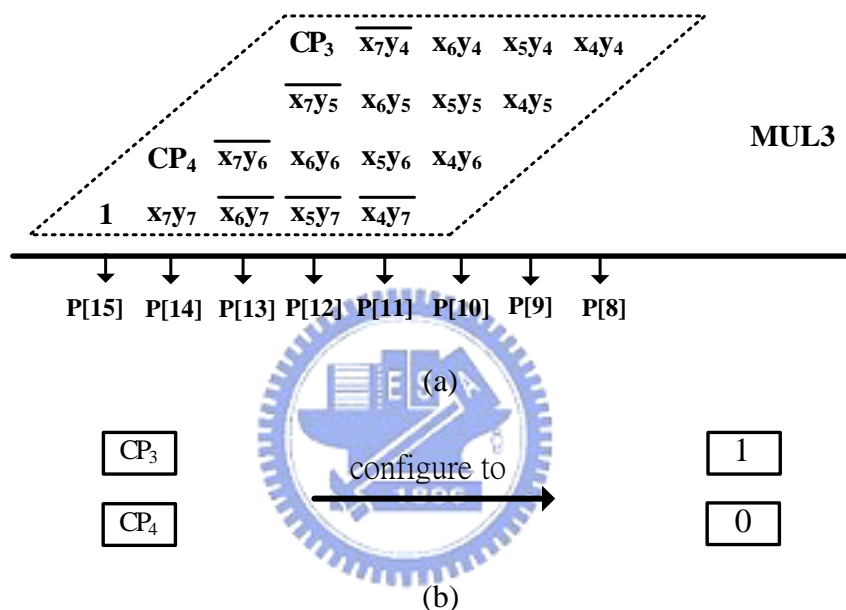


Fig. 3.14. (a) Proposed partial-product array diagram for CM3, and (b) configuration parameter settings.

### 3.2.4 CM4: Two $n/4 \times n/4$ Full-Precision Multipliers

CM4 widely used in lower resolution operation serves as performing two  $n/4 \times n/4$  full-precision multiplications. Under the minimum number of modules and partial-product configuration setting constraints, we make use of the MUL3 to fulfill the CM4 operation. Due to the use of MUL3, the corresponding subword operation of CM4 is illustrated in Fig. 3.15, where two subword products are  $X_2Y_2$  and  $X_3Y_3$ , and each fixed-width multiplication has  $n/2$ -bit wide output. Then, the partial product array

diagram of two  $n/4 \times n/4$  full-precision multipliers can be obtained in Fig. 3.16(a). In Fig. 3.16(a), compared with partial products of the MUL3 of CM1,  $x_5y_4$  and  $x_4y_5$  are complemented,  $x_6y_4$  and  $x_6y_5$  are configured to one,  $\overline{x_7y_4}$ ,  $\overline{x_7y_5}$ ,  $x_4y_6$ ,  $x_5y_6$ ,  $\overline{x_4y_7}$ , and  $\overline{x_5y_7}$  are configured to zero. The configuration parameters of CM4 can be set as addressed in Fig. 3.16(b), where  $CP_3$  and  $CP_4$  are set to 0 and 1, respectively. The rest partial products are unchanged.

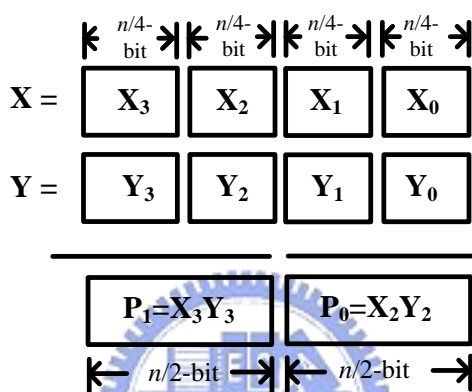
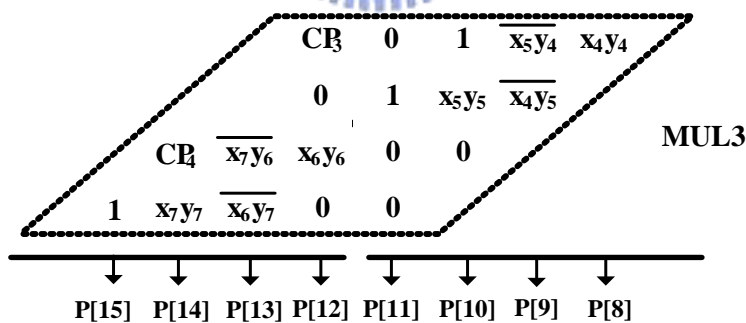
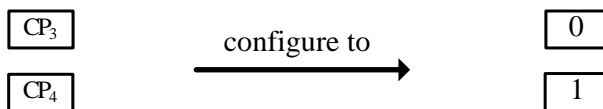


Fig. 3.15. Subword operation for two  $n/4 \times n/4$  full-precision multiplications.



(a)



(b)

Fig. 3.16. (a) Proposed partial-product array diagram for CM4, and (b) configuration parameter settings.

### 3.2.5 Proposed Structure

The proposed reconfigurable fixed-width Baugh-Wooley multiplier for  $n=8$  is a pipelined structure as depicted in Fig. 3.17, where ADD and MUX denote an adder and a multiplexer, respectively. The detailed diagrams of the corresponding MUL1, MUL2, MUL3 are exposed in Fig. 3.18, 3.19, and 3.20, respectively, where A, ND, HA, and FA denote an AND gate, a NAND gate, a half adder and a full adder, respectively, and the logic diagrams of the other processing elements are depicted in Fig. 3.21. The overall structure in Fig. 3.17 is partitioned into three stages. The first stage is responsible for decoding the operation (OP) code to generate control signals for the next stage, where the truth table of this decoder is listed in Table 3.5. According to the control signals, we can manipulate three multiplication modules involving MUL1, MUL2, and MUL3 are manipulated at the second stage. As shown in Fig. 3.18, 3.19 and 3.20, since CM1 and CM2 enable MUL1 and MUL2 to compute at the same time,  $t[2]$  are used to configure MUL1 and MUL2 for correct function. Similarly, since CM1, CM3 and CM4 need to enable MUL3,  $t[1]$  and  $t[0]$  with the values of  $\{00, 10, 01\}$  are used to configure the MUL3 in accordance with three different modes. As a consequence,  $CP_0$ ,  $CP_1$ , and  $CP_2$  can be implemented by  $t[2]$ ,  $CP_3$  and  $CP_4$  can be realized by  $t[1]$  and  $t[0]$ , respectively. In another viewpoint, from configuration parameter settings as shown in Fig. 3.11(c), 3.13(b), 3.14(b), 3.16(b), we can easily follow the above CP implementation.

A multiplexer at the second stage selects the output of MUL3 or the concatenation output of MUL1 and MUL2, and this design will be beneficial for power saving discussed in the next section. For CM1, since we have three multiplier modules MUL1, MUL2, and MUL3 to implement  $n \times n$  fixed-width multiplication for Type 1 with  $\theta_{Q=0,w=1}$  [19], two adaptive compensation biases of MUL1 and MUL2 are needed to carefully control. According to the binary thresholding mentioned in [19], if each

adaptive compensation bias adds a constant  $K = 1/2$  for  $\theta_{Q=0, w=1} = 0$ , the two adaptive compensation biases are not equivalent to the compensation design as shown in Fig. 5 of [19]. Thus, the design will lead to larger truncation error for CM1 than that of adding a constant  $K=1/2$  one time. As mentioned in chapter 3.1.5, we propose sub-calibration-circuit 1 (SCC1) and sub-calibration-circuit 2 (SCC2) to keep away from double constant addition and to achieve this reconfiguration for  $n \times n$  and  $n/2 \times n/2$  fixed-width multiplications. The logic diagram of SCC1 and SCC2 as shown in Fig. 3.21 is little area overhead, where the truth table of SCC1 and SCC2 is the same as Table 3.3.

The third stage is in charge of accumulating the output values of MUL1, MUL2, and MUL3 for CM1 and selecting output of final product according to four CMs. In Fig. 3.17, ADD1 adds the output of MUL1 and MUL2; however, the output bits of ADD1 only include carryout and ignore least significant bit due to the fixed-width output. For example, originally,  $A[3:0] + B[3:0]$  will produce {carry-out,  $C[3:0]$ }, but we only need {carryout,  $C[3:1]$ }. ADD2 adds the output of ADD1 and the output of the multiplexer at the second stage to achieve CM1. We make use of the control signal  $t[3]$  to determine the final correct product among different CMs. Note that the proposed reconfigurable methodology and concept can be applied to the larger bit width and used to increase configuration modes such as  $n/8 \times n/8$  and  $n/16 \times n/16$  multipliers while the larger word length is given. For example, from the above analysis, the conventional full-precision subword multiplication schemes [21-26] can be applied to MUL3 to increase configuration modes including four  $n/8 \times n/8$ , eight  $n/16 \times n/16$  full-precision multipliers, and so forth according to the larger input word length  $n$ . On the other hand, although we discuss only 2's-complement multiplication in this thesis, this reconfigurable concept can be easily extended to un-signed array multiplication.

Table 3.5: Truth table of the decoder for the reconfigurable fixed-width Baugh-Wooley multiplier

OP[2:0]	t[3]	t[2]	t[1]	t[0]
00(CM1)	1	0	0	0
01(CM2)	0	1	0	0
10(CM3)	0	0	1	0
11(CM4)	0	0	0	1

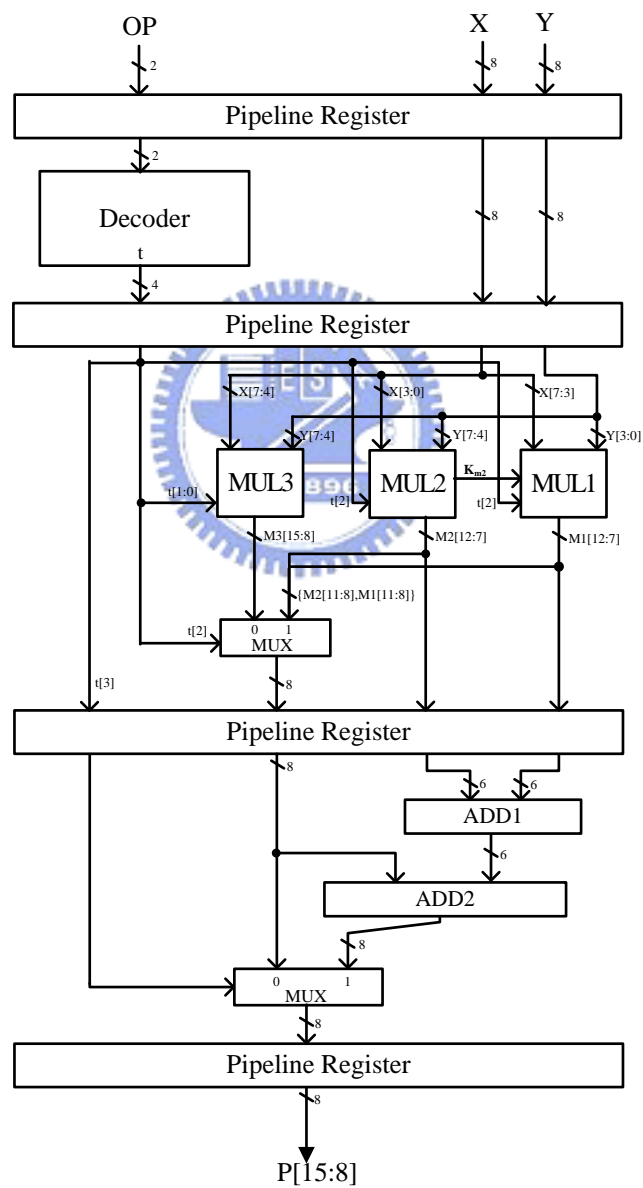


Fig. 3.17. Proposed pipelined reconfigurable multiplier.



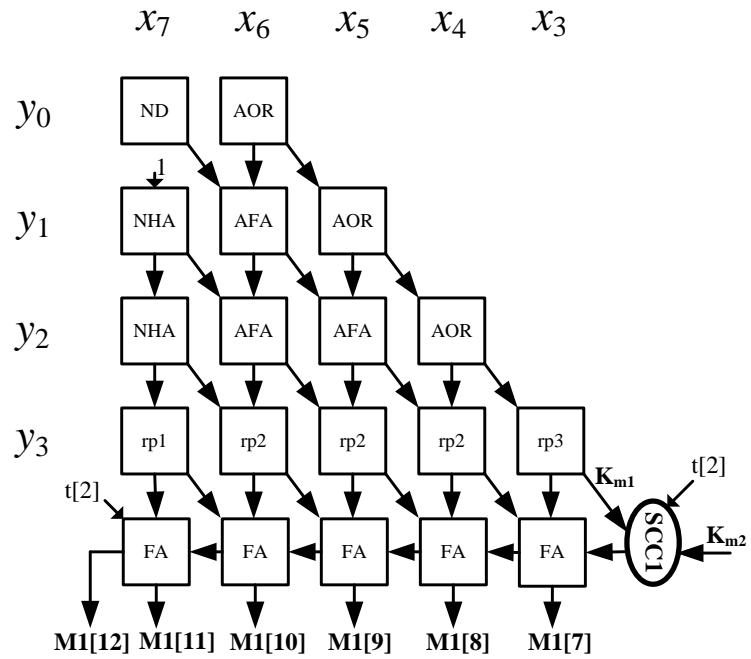


Fig. 3.18. Structure of MUL1.

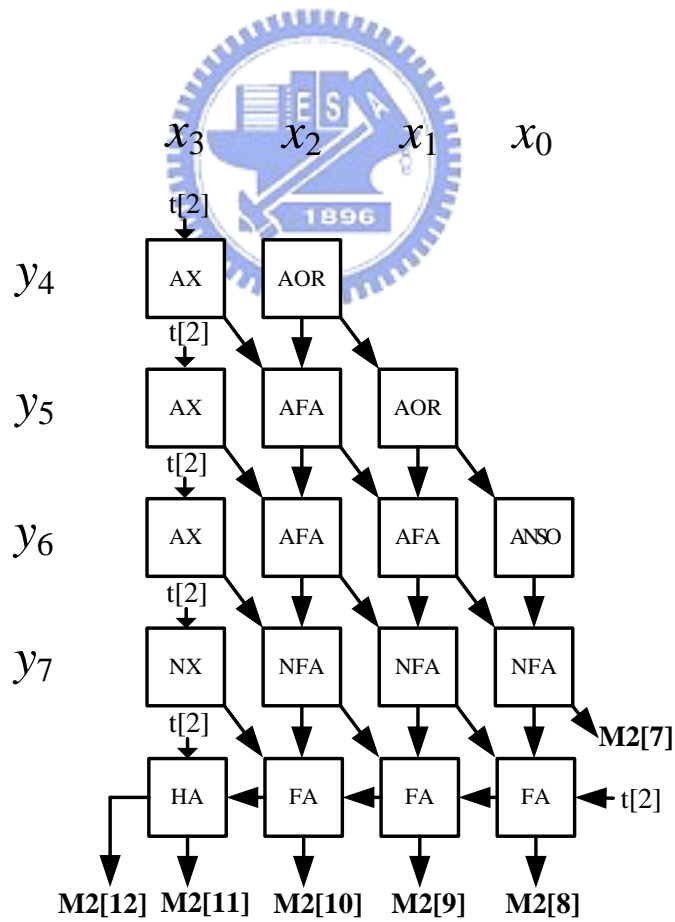


Fig. 3.19. Structure of MUL2.

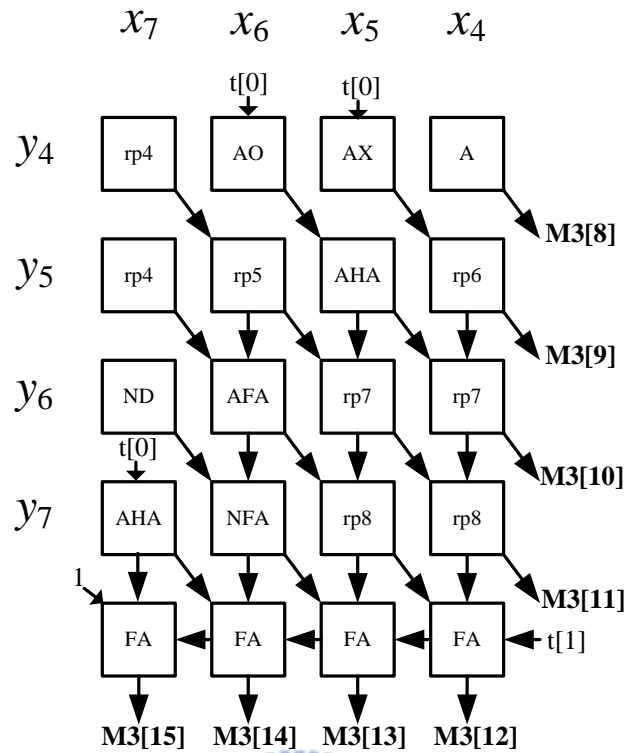


Fig. 3.20. Structure of MUL3.

In the following, we further discuss how to design power efficient pipelined reconfigurable multiplier. As mentioned in the above, the multiplications of CM2, CM3, and CM4 are of power-inefficient because they invoke all hardware resource to compute. It is desirable to apply low-power schemes such that the proposed reconfigurable fixed-width Baugh-Wooley multiplier possesses power-efficient capability. We apply low-power schemes including clock gating and zero input techniques to achieve power saving.

Name	rp1	rp6	AOR	AFA
Logic Diagram				
Name	rp2	rp7	AO	NFA
Logic Diagram				
Name	rp3	rp8	AX	AHA
Logic Diagram				
Name	rp4	SCC1	NX	NHA
Logic Diagram				
Name	rp5	SCC2	ANSO	
Logic Diagram				

Fig. 3.21. Logic diagrams of the other processing elements.

The clock-gating scheme is applied to the registers at the second and third stage of Fig. 3.22 in order to reduce unnecessary transitions. According to the following rules, we are able to disable the corresponding pipeline registers for power saving.

- a) If CM1 is performed, the input register of MUL1, MUL2, or MUL3 is conditionally disabled (i.e., referred to gated register in Fig. 3.22). The disable conditions depend on which input value of the register is zero.
- b) If CM2 is performed, input registers of MUL3 and ADD1 can be disabled.
- c) If CM3 is performed, input registers of MUL1, MUL2 and ADD1 can be disabled.
- d) If CM4 is performed, input registers of MUL1, MUL2 and ADD1 can be disabled.

The penalty of this scheme is the hardware overhead. The overhead covers the duplicated input registers so as to achieve the gated register for each multiplication module. If no duplicated input register is considered, for example of CM2 with disabling MUL3 (i.e., input registers for  $X[7:4]$  and  $Y[7:4]$  are disabled), the outputs of MUL1 and MUL2 must be wrong because MUL1 and MUL2 need  $X[7:4]$  and  $Y[7:4]$ , respectively, to generate the product. Hence, we duplicate input register for  $X[7:4]$  and  $Y[7:4]$  such that the input registers of MUL1, MUL2, and MUL3 are separated in Fig. 3.22.

Furthermore, in CM1, since the inputs of MUL1, MUL2, and MUL3 are duplicated, we can detect zero values of input data to disable the multiplication module. The conditions of zero value of the input are described in the following.

- a) If  $X[7:4]$  is zero, input registers of MUL1 and MUL3 can be disabled.
- b) If  $X[3:0]$  is zero, input registers of MUL2 can be disabled.
- c) If  $Y[7:4]$  is zero, input registers of MUL2 and MUL3 can be disabled.
- d) If  $Y[3:0]$  is zero, input registers of MUL1 can be disabled.

Note that although one of input operands is zero, the product of multiplication module is not equal to zero. Because some partial products are inverted as shown in Fig. 3.11(b), the actual product outputs of the disabled MUL3 and MUL2 should be  $(111100000)_2$

and  $(001111)_2$ , respectively. MUL1 is more particular since we must concern with partial product  $x_3y_3$  and  $K_{m2}$ . Let us consider the following cases ( $\wedge$ ,  $\vee$  denote AND and OR operators, respectively).

- a) If  $x_3y_3=0$  and  $K_{m2}=0$ , the output of SCC1 is 0 such that MUL1 produces  $(010001)_2$ .
- b) If  $x_3y_3=0$  and  $K_{m2}=1$ , the output of SCC1 is 1 such that MUL1 produces  $(010010)_2$ .
- c) If  $x_3y_3=1$  and  $K_{m2}=0$ , the output of SCC1 is 0 such that MUL1 produces  $(010010)_2$ .
- d) If  $x_3y_3=1$  and  $K_{m2}=1$ , the output of SCC1 is 0 such that MUL1 produces  $(010010)_2$ .

Since we would like to disable MUL1, the input  $x_3y_3$  and  $K_{m2}$  of MUL1 must be latched and thus the output signal of SCC1 will be unchanged. From the above four cases, the actual product of the disabled MUL1 is  $(0100, x_3y_3 \vee K_{m2}, \overline{x_3y_3 \vee K_{m2}})_2$  via logic operation of  $x_3y_3$  and  $K_{m2}$  as shown in Fig. 3.22. On the other hand, the CU (control unit) in Fig. 3.22 is used to treat  $K_{m2}=1$  when MUL2 is disabled. The block denoted as L is a latch to keep present value when MUL1 is disabled. According to the above analysis, the signals  $g\_M1$ ,  $g\_M2$ ,  $g\_M3$ , and  $t[3]$  are generated to control four gated registers and the former three signals are used to control three multiplexers of the actual product selection as shown in Fig. 3.22 such that low power consumption is achieved.

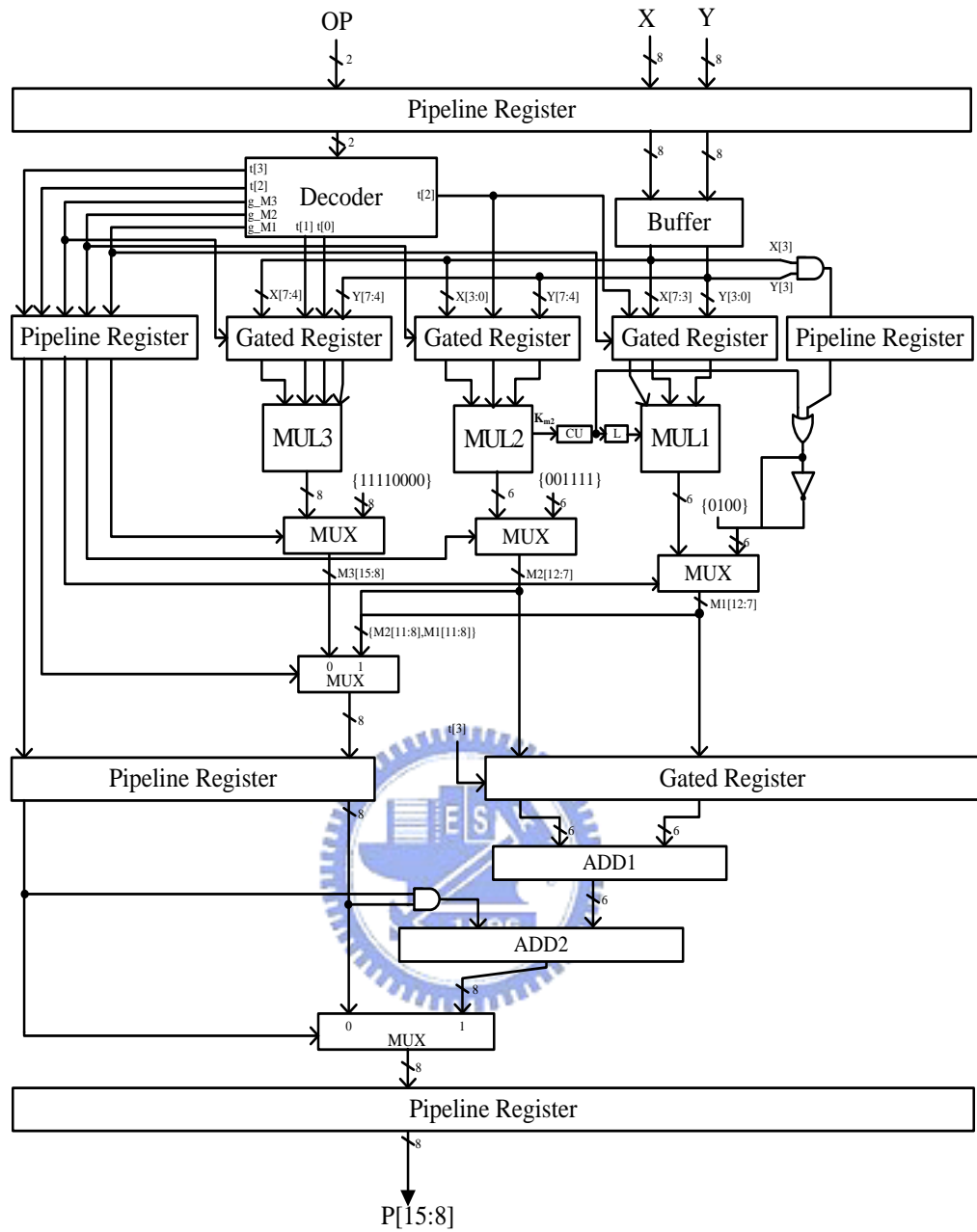


Fig. 3.22. Proposed power-efficient pipelined reconfigurable fixed-width Baugh-Wooley multiplier.

Zero-input scheme working for CM2, CM3, and CM4 is mainly aimed at providing zero input sequences for adder to keep value unchanged at the third stage of Fig. 3.22. If CM2, CM3, or CM4 is performed, we use AND gates to generate zero sequence and feed into the ADD2. In this case, for ADD1, we can use  $t[3]$  as the control signal of the

clock-gating register to latch its input value. At the same time, for ADD2, one of inputs comes from ADD1 which has been latched and we only need to set the other input to zero via AND operation with  $t[3]$ . Thus, we can further reduce the transition actively while the same CM is successively performed. On average, the gated-clock and zero-input schemes reduce around 98% and 2% of the total power reduction, respectively since the latter scheme only affects only ADD2 at the third stage.



# Chapter

# 4

## Implementation and Comparison

---

In this section, we present the main differences among the various reconfigurable multipliers in qualitative way and show the power and area comparison results among power-efficient reconfigurable and non-reconfigurable fixed-width multipliers in quantitative behavior. The qualitative comparison results between the proposed reconfigurable multiplier and other existing reconfigurable multipliers are listed in Table 4.1. From Table 4.1, only the proposed reconfigurable multiplier uses the fixed-width multiplier infrastructure to generate fixed-width and full-precision multipliers. Thus, we can directly provide two useful precision outputs for DSP and computer applications. Other reconfigurable multipliers [21-29] apply the full-precision multiplier infrastructure to generate only full-precision multipliers. The proposed reconfigurable multiplier and other reconfigurable multipliers [21-26, 27, 29] have compact design complexity in comparison with that of [28] because the multiplier in [28] needs to reconfigure more different function modes and pipeline stages. The number of operands of the proposed multiplier and published multipliers [21-28] are variable such that the designs can provide multiple lower resolution operations.



Table 4.1: Qualitative comparison between different reconfigurable architectures

	[21-26]	[27]	[28]	[29]	This work
Multiplication Infrastructure	Full-precision	Full-precision	Full-precision	Full-precision	Fixed-width
Precision Provided	Full-precision	Full-precision	Full-precision	Full-precision	Fixed-width, Full-precision
Complexity	Compact	Compact	Large	Compact	Compact
#Pipeline stages	Non-pipelined	Fixed	Variable	Non-pipelined	Fixed
Function	Multiplication	Inner product	Mac, multiplication, addition, data format conversation	Multiplication	Multiplication
#Operands	Variable	Variable	Variable	Fixed	Variable

## 4.1 Simulation Results of Reconfigurable Fixed-Width

### Multiplier

Concerning the chip implementation, we adopt the cell-based design flow with Artisan standard cell library and implement the reconfigurable fixed-width multiplier in TSMC 0.18 um CMOS process. Synopsys Design Compiler is employed to synthesize the RTL design of the proposed reconfigurable multiplier and Cadence SOC Encounter is adopted for placement and routing (P&R).

#### 4.1.1 Reconfigurable Fixed-Width Booth Multiplier

The active chip layout of the proposed reconfigurable 8x8 fixed-width Booth multiplier is shown in Fig. 4.1. Although we have mentioned the main differences in qualitative way as listed in Table 4.1, it is difficult to compare the performance with other previous reconfigurable multipliers [21-29] in quantitative way due to different CMs/functions, different number of CMs, different prototype multiplier infrastructures, and different targets. In order to show the power consumption and chip area comparison results in quantitative way, we reproduce non-reconfigurable fixed-width multiplier.

Note that the non-reconfigurable fixed-width Booth multiplier is the same as the fixed-width Booth multiplier with  $w=1$  and  $Q=0$  in [20]. Table 4.2 and 4.3 reveal chip characteristics of the proposed reconfigurable fixed-width Booth multipliers and the non-reconfigurable fixed-width Booth multiplier for  $n=8$  and 16, respectively. The power consumption is measured via Synopsys PrimePower using 10,000 random input vectors after RC extraction of the placed and routed netlists. All the simulation results are obtained at 125 MHz with 1.8V. From Table 4.2 and 4.3, the presented four configuration modes of the reconfigurable fixed-width multiplier are capable of providing high resolution, parallel, full-precision multiplications, or multiplication and add operations for different computation demands. In terms of power issue for  $n=8$ , CM2, CM3, and CM4 can attain the power saving of 17.29%, 26.68%, and 11.35% with respect to that of CM1. On the other hand, for  $n=16$ , CM2, CM3, and CM4 can attain the power saving of 23.44%, 33.66%, and 14.83% with respect to that of CM1. It is worth noting that the power consumption of CM2 or CM4 is the summation of two subword multiplier operations rather than single operation. Also, in comparison with the power dissipation of the non-reconfigurable multiplier, the proposed one can achieve power reduction of 6.10% and 14.0% on average for  $n=8$  and 16, respectively. Obviously, the proposed design can achieve computation and power scalable through four-mode control at the expense of slightly increment of area and power consumption by 13.02% and 4.84% for  $n=16$ , respectively, compared with the non-reconfigurable multiplier.

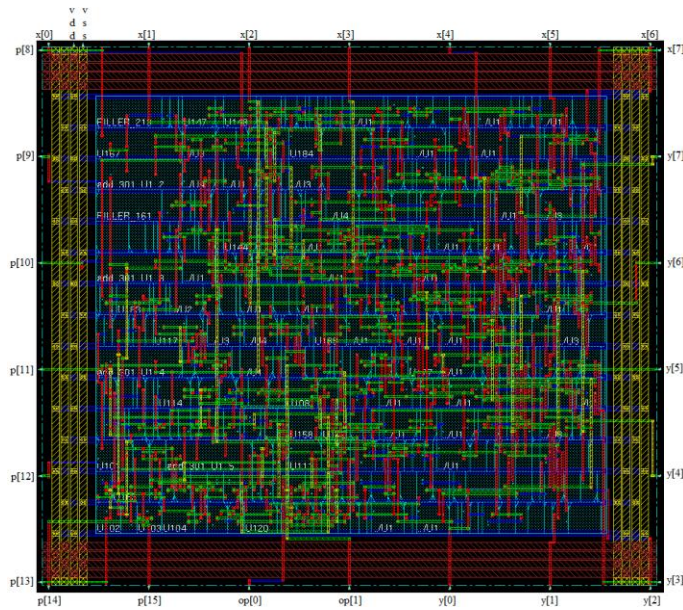


Fig. 4.1. Proposed reconfigurable fixed-width Booth multiplier layout for  $n=8$ .

Table 4.2: Chip Characteristics of the 8x8 reconfigurable fixed-width Booth multipliers and the 8x8 non-reconfigurable fixed-width Booth multipliers

		Non-reconfigurable Fixed-Width Booth Multiplier	Reconfigurable Fixed-Width Booth Multiplier
# of Supporting Configuration Modes		1	4
Multiplier & Multiplicand Word Length (n)		8 bits	8 bits
Product Word Length (n)		8 bits	8 bits
Active Chip Area		5096.0 $\mu\text{m}^2$	6080.7 $\mu\text{m}^2$
Power Consumption	CM1	1.722 mW	1.885 mW (100 %)
	CM2	N/A	1.559 mW (82.71 %) / two multiplications
	CM3	N/A	1.382 mW (73.32 %)
	CM4	N/A	1.671 mW (88.65 %) / two multiplications
	Average	1.722 mW	1.617 mW

Table 4.3: Chip Characteristics of the 16x16 reconfigurable fixed-width Booth multipliers and the 16x16 non-reconfigurable fixed-width Booth multipliers

		Non-reconfigurable Fixed-Width Booth Multiplier	Reconfigurable Fixed-Width Booth Multiplier
# of Supporting Configuration Modes		1	4
Multiplier & Multiplicand Word Length (n)		16 bits	16 bits
Product Word Length (n)		16 bits	16 bits
Active Chip Area		14992.1 $\mu\text{m}^2$	16944.7 $\mu\text{m}^2$
Power Consumption	CM1	5.305 mW	5.562 mW (100 %)
	CM2	N/A	4.258 mW (76.56%) / two multiplications
	CM3	N/A	3.690 mW (66.34%)
	CM4	N/A	4.737 mW (85.17%) / two multiplications
	Average	5.305 mW	4.562 mW

#### 4.1.2 Reconfigurable Fixed-Width Baugh-Wooley Multiplier

The active chip layout of the proposed pipelined reconfigurable 16x16 Baugh-Wooley fixed-width multiplier is shown in Fig. 4.2. Table 4.4, 4.5, 4.6, and 4.7 reveal the power consumption and chip area comparison among the non-reconfigurable pipelined fixed-width Baugh-Wooley multiplier and power-efficient pipelined reconfigurable fixed-width Baugh-Wooley multipliers for  $n=8$ , 16, 24, and 32, respectively. Note that the non-reconfigurable pipelined fixed-width Baugh-Wooley multiplier is a pipelined version of the fixed-width multiplier with  $w=1$  and  $Q=0$  in [19]. We measure the power consumption using 100,000 random input vectors via Synopsys PrimePower at 100 MHz with 1.8 V after RC extraction from the placed and routed netlists. From Table 4.4, 4.5, 4.6, and 4.7, CM2, CM3, and CM4 compared with CM1 result in lower power dissipation while  $n$  increases from 8 to 32. In comparison to the

power dissipation of the non-reconfigurable structure, the proposed one possesses lower average power consumption. We can see that, for  $n=32$ , CM2, CM3, and CM4 can attain the power saving of 36.25%, 44.55%, and 58.34% with respect to that of CM1. In the same case, although the proposed reconfigurable multiplier has 36.48% more area than that of the non-reconfigurable structures, the average power consumption of the proposed architecture can certainly attain the larger power saving than the non-reconfigurable structure. From the simulation results, the proposed multiplier can achieve power reduction of 0.81%, 12.56%, 17.93%, and 23.2% on average for  $n=8, 16, 24$ , and 32, respectively, compared with that of the non-reconfigurable multiplier. Importantly, the non-reconfigurable multiplier cannot provide more than one configuration mode compared with the reconfigurable design. Note that the power consumption of CM1 of the reconfigurable multiplier closely approaches that of the non-reconfigurable one while the length of  $n$  increases.

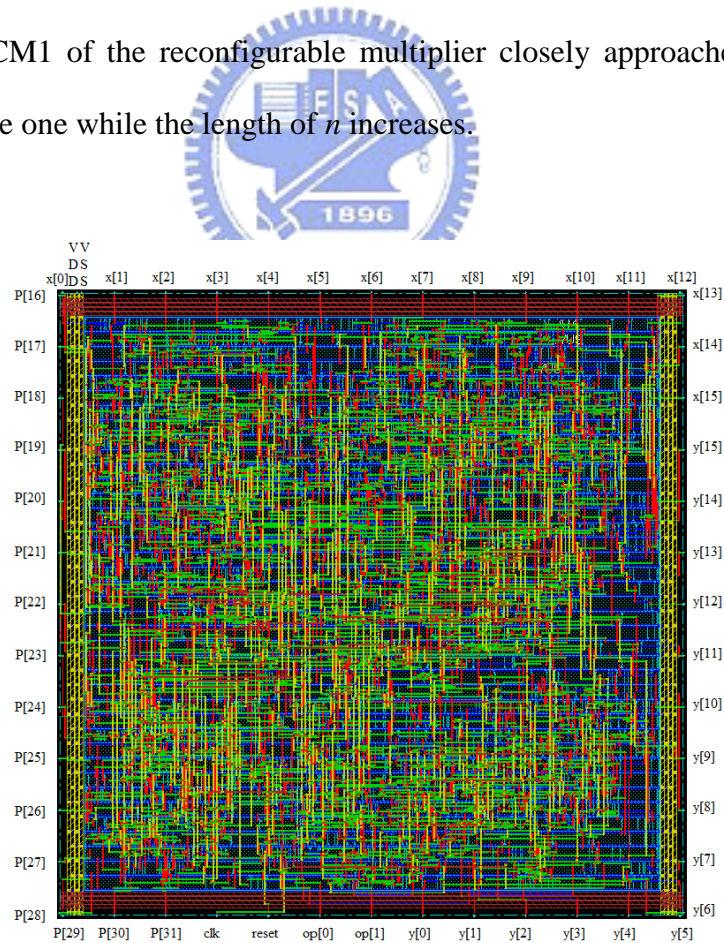


Fig. 4.2. Proposed pipelined reconfigurable fixed-width Baugh-Wooley multiplier layout for  $n=16$ .



Table 4.4: Chip Characteristics of the 8x8 reconfigurable fixed-width Baugh-Wooley multipliers and the 8x8 non-reconfigurable fixed-width Baugh-Wooley multipliers

		Non-Reconfigurable Pipelined Fixed-Width Baugh-Wooley Multiplier	Reconfigurable Pipelined Fixed-Width Baugh-Wooley Multiplier
Multiplier & Multiplicand Word Length (n)		8 bits	8 bits
Product Word Length (n)		8 bits	8 bits
Active Chip Area		9032.5 $\mu\text{m}^2$	13438.6 $\mu\text{m}^2$
Power Consumption	CM1	1.609 mW	2.145 mW (100 %)
	CM2	N/A	1.607 mW (74.92 %) / two multiplications
	CM3	N/A	1.389 mW (64.76 %)
	CM4	N/A	1.242 mW (57.90 %) / two multiplications
	Average	1.609 mW	1.596 mW

Table 4.5: Chip Characteristics of the 16x16 reconfigurable fixed-width Baugh-Wooley multipliers and the 16x16 non-reconfigurable fixed-width Baugh-Wooley multipliers

		Non-Reconfigurable Pipelined Fixed-Width Baugh-Wooley Multiplier	Reconfigurable Pipelined Fixed-Width Baugh-Wooley Multiplier
Multiplier & Multiplicand Word Length (n)		16 bits	16 bits
Product Word Length (n)		16 bits	16 bits
Active Chip Area		25883.5 $\mu\text{m}^2$	38723.3 $\mu\text{m}^2$
Power Consumption	CM1	4.777 mW	6.237 mW (100 %)
	CM2	N/A	4.358 mW (69.87 %) / two multiplications
	CM3	N/A	3.414 mW (54.74 %)
	CM4	N/A	2.719 mW (43.59 %) / two multiplications
	Average	4.777 mW	4.182 mW

Table 4.6: Chip Characteristics of the 24x24 reconfigurable fixed-width Baugh-Wooley multipliers and the 24x24 non-reconfigurable fixed-width Baugh-Wooley multipliers

		Non-Reconfigurable Pipelined Fixed-Width Baugh-Wooley Multiplier	Reconfigurable Pipelined Fixed-Width Baugh-Wooley Multiplier
Multiplier & Multiplicand Word Length (n)		24 bits	24 bits
Product Word Length (n)		24 bits	24 bits
Active Chip Area		59503.9 $\mu\text{m}^2$	85202.9 $\mu\text{m}^2$
Power Consumption	CM1	10.56 mW	13.23 mW (100 %)
	CM2	N/A	8.846 mW (66.86 %) / two multiplications
	CM3	N/A	7.042 mW (53.23 %)
	CM4	N/A	5.551 mW (41.96 %) / two multiplications
	Average	10.56 mW	8.667 mW

Table 4.7: Chip Characteristics of the 32x32 reconfigurable fixed-width Baugh-Wooley multipliers and the 32x32 non-reconfigurable fixed-width Baugh-Wooley multipliers

		Non-Reconfigurable Pipelined Fixed-Width Baugh-Wooley Multiplier	Reconfigurable Pipelined Fixed-Width Baugh-Wooley Multiplier
Multiplier & Multiplicand Word Length (n)		32 bits	32 bits
Product Word Length (n)		32 bits	32 bits
Active Chip Area		109203.8 $\mu\text{m}^2$	149046.6 $\mu\text{m}^2$
Power Consumption	CM1	20.17 mW	23.75 mW (100 %)
	CM2	N/A	15.14 mW (63.75 %) / two multiplications
	CM3	N/A	13.17 mW (55.45 %)
	CM4	N/A	9.894 mW (41.66 %) / two multiplications
	Average	20.17 mW	15.49 mW

## 4.2 Application of Reconfigurable Fixed-Width Multiplier

Next, the proposed 8x8 reconfigurable fixed-width multiplier is applied to the 35-tap FIR filter for speech processing. The behavior of a digital FIR filter can be expressed in (13).

$$O(m) = \sum_{i=0}^{L-1} H(i)S(m-i) \quad (13)$$

where  $O(i)$ ,  $H(i)$ , and  $S(i)$  denote output sequence, filter coefficient and input sequence at  $i$ th discrete time, respectively. For convenience of evaluation of various multipliers, we take 1,000 samples for the consonant part and the vowel part of “Chicken.” The error performance and power consumption of four CMs are tabulated in Table 4.8, where error performance is measured by signal-to-noise ratio (SNR) and the power consumption is still measured at 125 MHz. The floating-point output is regarded as an error-free standard output, which is used to assess the FIR filter performance using the proposed four modes and an 8x8 full-precision multiplier mode. The error performance and power consumption are tabulated in Table 4.8, where the error performance is measured by signal-to-noise ratio (SNR) and the power consumption is still measured at 125 MHz. From the comparison results in Table 4.8, the CM1 mode shows higher SNR value than that by other three CMs including CM2, CM4, and HPFM modes. However, the CM1 mode consumes more power than other three proposed modes. Compared with the CM1 mode, CM2 and CM4 modes can result in 53.20% and 51.48% power saving, respectively, but these two modes lead to 14.57 dB SNR loss in this benchmark. For fair comparison between CM1 and an 8x8 full-precision multiplier mode owing to the same input bit width, the CM1 mode can save 39.23% power consumption compared to the latter one with 10.21 dB loss. Under the same 4-bit input bit width, compared with the CM3 mode, CM2 and CM4 modes can attain power saving by 31.28% and 28.76%,



respectively, with 4.66 dB SNR loss. Since CM2 and CM4 modes can concurrently generate two multiplication products, only the number of  $L/2$   $n \times n$  multipliers is needed to finish FIR filter operation. Therefore, CM2 and CM4 modes can save the number of  $L/2$   $n \times n$  multipliers. As for the CM3 mode, the SNR and power consumption are between the performance of CM1 and CM2/CM4 modes. In order to compare with the published FIR filter work [33-35], one terminology introduced in [33] is used to indicate the normalized power dissipation per multiplier,  $P(mult)$ , and is given by

$$P(mult) = \frac{Total\ Power}{\#mults} \cdot \frac{8}{\#bits\ coeff} \cdot \frac{8}{\#bits\ sample} \cdot \left(\frac{1.8}{Vdd}\right)^2 \cdot \frac{0.18}{Tech} \cdot \frac{125}{clk\ freq}, \quad (14)$$

where  $Vdd$  and  $Tech$  denote the power supply voltage and process technology, respectively. From the comparison results of Table 4.9, the FIR filter using the proposed reconfigurable multiplier has the lowest normalized power consumption. The main reason is that the fixed-width multiplier has less power consumption than the full-precision multiplier does in Table 4.8. Hence, in this benchmark as listed in Tables 4.8 and 4.9, the proposed reconfigurable fixed-width Booth multiplier shows power scalable capability and better power saving with satisfactory error performance compared with other FIR filters.

Table 4.8: Evaluation results of error signals and power consumption obtained with by the proposed 8x8 reconfigurable fixed-width Booth multiplier for FIR filter application

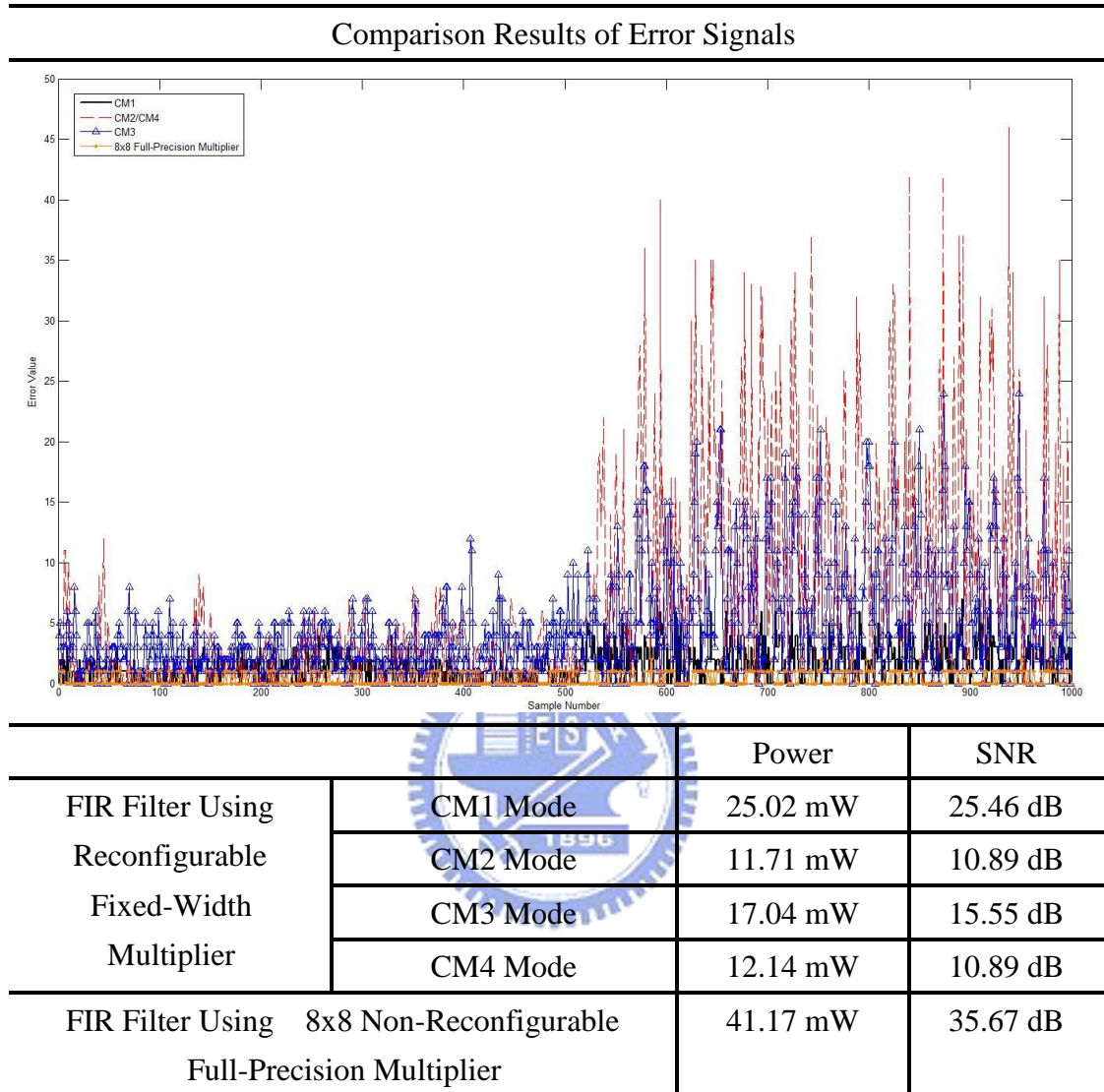


Table 4.9: Comparison results among FIR filters

Ref.	Description	Tech.	V <sub>dd</sub>	Power	P(mult)
[33]	128-tap, 32 10x12 mult@80MHz	0.5 um	3.3V	415 mW	1.16 mW
[34]	12-tap, 12 10x8 mult@40MHz	0.5 um	1.8V	10.9 mW	0.82 mW
[35]	32-digit, 11.52 12x8mult @100MHz	0.35 um	2.5V	80 mW	1.54 mW
This work (CM1 mode)	35-tap, 35 8x8 mult@125MHz	0.18 um	1.8 V	25.02 mW	0.71 mW

# Chapter

# 5

## Conclusion and Future Work

---

This thesis proposed a framework for the reconfigurable fixed-Width Booth multiplier and the reconfigurable fixed-width Baugh-Wooley multiplier to generate a family of fixed-width and full-precision multipliers. From the implementation results, the presented four configuration modes of the reconfigurable fixed-width multiplier are capable of providing high resolution, parallel, or full-precision multiplications for different computation demands. On the other hand, the proposed reconfigurable fixed-width Booth multiplier can attain the power saving of 14.0% on average with respect to that of non-reconfigurable fixed-width Booth multiplier with  $n=16$ . Also, the proposed pipelined reconfigurable fixed-width Baugh-Wooley multiplier can save 12.56% power consumption on average in comparison with that of pipelined non-reconfigurable fixed-width Baugh-Wooley multiplier with  $n=16$ . The future work is to apply the developed multipliers to the power-aware systems.

---

# Bibliography

---

- [1] A. D. Booth, "A signed binary multiplication techniques," *Quart. J. Mech. Appl. Math.*, vol. 4, pp. 236-240, 1951.
- [2] O. L. MacSorley, "High-speed arithmetic in binary computer," *Proc. IRE*, vol. 49, pp. 67-91, 1961.
- [3] H. Sam and A. Gupta, "A generalized multibit recoding of two's complement binary numbers and its proof with applications in multiplier implementations," *IEEE Trans. Comput.*, vol. 39, pp. 1006-1015, Aug. 1990.
- [4] C. R. Baugh and B. A. Wooley, "A two's complement parallel array multiplication algorithm," *IEEE Trans. Comput.*, vol. C-22, no. 12, pp. 1045-1047, Dec. 1973.
- [5] K. Hwang, *Computer Arithmetic: Principles, Architecture, and Design*. New York: John-Wiley, 1979.
- [6] F. Cavanagh, *Digital Computer Arithmetic: Design and Implementation*. New York: McGraw-Hill, 1984.
- [7] M. D. Ercegovac and T. Lang, *Digital Arithmetic*, Morgan and Kaufmann, 2004.
- [8] S. L. Freeny, "Special-purpose hardware for digital filtering," *Proc. IEEE*, vol. 63, no. 4, pp. 633-647, Apr. 1975.
- [9] Y. C. Lim, "Single-precision multiplier with reduced circuit complexity for signal processing applications," *IEEE Trans. Comput.*, vol. 41, no. 10, pp. 1333-1336, Oct. 1992.
- [10] M. J. Schulte and E. E. Swartzlander, Jr., "Truncated multiplication with correction

- constant,” *VLSI Signal Processing, VI, New York: IEEE Press*, 1993, pp. 388-396.
- [11] S. S. Kidambi, F. El-Guibaly, and A. Antoniou, “Area-efficient multipliers for digital signal processing applications,” *IEEE Trans. Circuits Syst. II*, vol. CAS-43, no. 2, pp. 90-94, Feb. 1996.
- [12] E. J. King and E. E. Swartzlander, Jr., “Data-dependent truncation scheme for parallel multipliers,” in *Proc. 31st Asilomar Conference on Signals, Systems, and Computers*, 1997, vol. 2, pp. 1178-1182, Pacific Grove, CA.
- [13] E. E. Swartzlander, Jr., “Truncated multiplication with approximate rounding,” in *Proc. 33rd Asilomar Conference on Signals, Systems, and Computers*, 1999, vol. 2, pp. 1480-1483.
- [14] J. M. Jou, S. R. Kuang, and R. D. Chen, “Design of low-error fixed-width multiplier for DSP applications,” *IEEE Trans. Circuits Syst. II*, vol. CAS-46, no. 6, pp. 836-842, Jun. 1999.
- [15] L. D. Van, S. S. Wang, and W. S. Feng, “Design of the lower-error fixed-width multiplier and its application,” *IEEE Trans. Circuits Syst. II*, vol. 47, pp. 1112-1118, Oct. 2000.
- [16] S. J. Jou, M. H. Tsai and Y. L. Tsao, “Low-Error Reduced-width Booth multiplier for DSP application,” *IEEE Trans. Circuits Syst. I*, vol. 50, pp. 1470-1474, Nov. 2003.
- [17] K. J. Cho, K. C Lee, J.G Chung and K. K. Parhi, “Design low-error fixed-width modified Booth multiplier,” *IEEE Trans. VLSI*, vol.12, pp. 522-531, No 5, May 2004.
- [18] T. B. Juang, S. F. Hsiao, ”Low-error carry-free fixed-width multiplies with low-cost compensation circuits,” *IEEE Trans. Circuits Syst. II*, vol. 52, pp. 299-303, June 2005.
- [19] L. D. Van and C. C. Yang, “Generalized low-error area-efficient fixed-width multipliers,” *IEEE Trans. Circuits Syst. I*, vol. 52, pp. 1608-1619, Aug. 2005.
- [20] M. A. Song, L. D. Van, and S. Y. Kuo, “Adaptive low-error fixed-width Booth

- multipliers,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E90-A, no. 6, pp. 1180-1187, Jun. 2007.
- [21] S. Krithivasan and M. J. Schulte, “Multiplier architectures for media processing,” in *Proc. IEEE Asilomar Conference on Signals, Systems, and Computers*, Nov. 2003, vol. 2, pp. 2193-2197.
- [22] Y.-H. Huang, H.-P. Ma, M.-L. Liou, and T.-D. Chiueh, “A 1.1 G MAC/s sub-word-parallel digital signal processor for wireless communication applications,” *IEEE Journal of Solid-State Circuits*, Vol.39, pp.169-183, Jan. 2004.
- [23] S. Krithivasan, M. J. Schulte, and J. Glossner, ”A subword-parallel multiplication and sum-of-squares unit,” *IEEE Computer society Annual Symposium on VLSI*, pp. 273-274, Feb. 2004.
- [24] Y.-L. Tsao, W.-H. Chen, M.-H. Tan, M.-C. Lin, and S.-J. Jou, “Low-power embedded DSP core for communication systems,” *EURASIP Journal on Applied Signal Processing*, pp.1355-1370, Jan. 2003.
- [25] D. Tan, A. Danysh and M. Liebelt, ”Multiple-precision fixed-point vector multiply-accumulator using shared segmentation,” in *Proc. IEEE Symposium on Computer Arithmetic*, pp. 12-19, Jun. 2003.
- [26] C. L. Wey and J. F. Li, ”Design of reconfigurable array multipliers and multiplier-accumulators,” in *Proc. IEEE Asia-Pacific Conference on Circuits and Systems*, Dec. 2004, pp. 37-40.
- [27] R. Lin, “Reconfigurable parallel inner product processor architecture,” *IEEE Trans. VLSI Syst.*, vol. 9, pp. 261-272, Apr. 2001.
- [28] K. Tatas, G. Koutroumpetis, D. Soudris, A. Thanailakis, "Architecture design of a coarse-grain reconfigurable multiply-accumulate unit for data-intensive applications," *Integration The VLSI Journal*, vol. 40, pp. 74-93, Feb. 2007.
- [29] S. D. Haynes and P. Y. K. Cheung, “Configurable multiplier blocks for embedding in

- FPGAs,” *Electronics Letter*, vol. 34, no. 7, pp. 638-639, Apr. 1998.
- [30] J. Di and J. S. Yuan, “Run-time reconfigurable power-aware pipelined signed array multiplier design,” in *Proc. IEEE International Symposium on Signals, Circuits, and Systems*, July 2003, vol. 2, pp. 405-406.
- [31] M. Sjalander, M. Drazdziulis, P. Larsson-Edefors, and H. Eriksson, “A low-leakage twin-precision multiplier using reconfigurable power gating,” in *Proc. IEEE International Symposium on Circuits, and Systems*, May 2005, vol. 2, pp. 1654-1657.
- [32] S.-R. Kuang and J.-P. Wang, “Design of power-efficient pipelined truncated multipliers with various output precision,” *IET Computers & Digital Techniques*, vol. 1, pp. 129-136, Mar. 2007.
- [33] C. J. Nicol, P. Larsson, K. Azadet, and J. H. O’Neill, “A low power 128-tap digital adaptive equalizer for broadband modems,” *IEEE J. Solid-State Circuits*, vol. 32, no. 11, pp. 1777–1789, Nov. 1997.
- [34] C. Henning, R. Schwann, V. Gierenz, and T. G. Noll, “A low power reconfigurable 12-tap FIR interpolation filter with fixed coefficient sets,” in *Proc. IEEE 26th European Solid-State Circuits Conference*, Sep. 2000, pp. 81-84.
- [35] K.-H. Chen and T.-D. Chiueh, “A low-power digit-based reconfigurable FIR filter,” *IEEE Trans. Circuits Syst. II*, vol. 53, pp. 617-621, Aug. 2006.

# Biography

---

**Jin-Hao Tu** was born in Taipei, Taiwan, R.O.C., in 1984. He received the B.S. degree from National Changhua University of Education, Changhua, Taiwan, in 2006, and the M.S. degree from National Chiao Tung University (NCTU), Hsinchu, Taiwan, in 2008, all in computer science. His research interests are computer arithmetic and 3D graphics system design.

