A Low Power Radix-4 Booth Multiplier With Pre-Encoded Mechanism

YEN-JEN CHANG1, (Member, IEEE), YU-CHENG CHENG1, SHAO-CHI LIAO2, AND CHUN-HUO HSIAO1

1Department of Computer Science and Engineering, National Chung Hsing University, Taichung City 402204, Taiwan
2Winbond Electronics Corporation, Hsinchu County 30274, Taiwan

Corresponding author: Yu-Cheng Cheng (co7835g@gmail.com)

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Project MOST 104-2221-E-005-025- and Project 108-2221-E-005-018-.

ABSTRACT The radix-4 Booth algorithm is widely used to improve the performance of multiplier because it can reduce the number of partial products by half. However, numerous additional encoders and decoders would cause the power consumption of the Booth multiplier to be considerable. In this paper, a new radix-4 Booth pre-encoded mechanism is proposed to reduce the power consumption of the Booth multiplier. The proposed design can effectively reduce the power of the Booth multiplier dissipated in the redundant activities by disabling the Booth encoders and decoders from unnecessary working. Particularly, since the control signals are generated early at the pipeline input register before the multiplier, the performance of our design is better than the traditional Booth multiplier. Based on the TSMC 40 nm technology, the simulation results show that the proposed pre-encoded mechanism can reduce the dynamic and static power by 45% and 65%, respectively, compared to the traditional 16-bit radix-4 Booth multiplier. Compared to the previous designs, the proposed design keeps the feature of race-free and has lower power consumption. Even compared to the approximate design, the proposed design has better power efficiency and can provide the exact products.

INDEX TERMS Booth algorithm, low power multiplier, power efficiency, partial product, radix-4 Booth multiplier.

I. INTRODUCTION

Many digital signal processing (DSP) and machine learning applications are heavily dominated by multiplication [1]–[4], e.g., more than 90% convolutional neural networks (CNN) computations are occupied by multiply-accumulate (MAC) operations [5], [6]. Therefore, the multiplier is an important component in various hardware platforms. The conventional multiplication includes three major phases [3], [7]–[9]. (1) Two inputs (multiplier and multiplicand) are multiplied to generate the partial products (PPs). (2) Reducing the PPs’ matrix into two rows by partial product reduction schemes (3) The final carry propagated addition of the remaining two rows of PPs. Particularly, the second phase plays a significant role in power consumption, cost, and overall performance [7]–[9]. Then, the radix-4 Booth algorithm can improve the performance of multiplication because the radix-4 Booth multiplier can reduce the number of PP rows by half [3], [10], [11].

The authors of [11] provided a simple and intuitive encoding/decoding method to implement the radix-4 Booth algorithm; reference [12] provided a modified sign extension structure to reduce the cost and to improve performance. In [16], the author found that the traditional radix-4 Booth implementations [13]–[15] may result in unnecessary glitches of PPs. Thus, the author of [16] proposed the glitch-free Booth encoder and partial product generator to eliminate the unnecessary glitches of the radix-4 Booth multiplier. However, these traditional designs still suffer from high power consumption and high cost of Booth encoders and decoders. Thus, the authors of [17] proposed a high performance and low cost radix-4 Booth decoder. The decoder of [17] also keeps the advantage of race-free and its cost is less than the cost of [16]. Research [18] developed a neg/two/one-nf generator (encoder and decoder) to reduce the glitches in the second phase of multiplication. This neg/two/one-nf generator has less encoded signals and its signal paths are more balanced than other schemes. In [19], in order to reduce the cost of the Booth encoder and decoder, the authors proposed a
novel modified Booth encoder (NMBE) scheme that is based on the pass transistor logic (PTL).

For some error-tolerant applications, the approximate circuits can be employed to achieve low power, low circuit complexity, and high performance [9], [20]–[22]. The traditional radix-8 Booth algorithm can generate fewer PPs than the radix-4 Booth algorithm, but it needs additional adders to process the operation of odd multiples of the multiplicand. Therefore, the approximate 2-bit adder [20] was proposed to generate the triple multiplicand with no carry propagation to improve the performance. In [21], the approximate radix-4 Booth multipliers were proposed by using their approximate Booth encoders and approximate Wallace tree structure. In [22], three approximation techniques for the radix-4 Booth multiplier were proposed and these designs can reduce the logic complexity of the PP generator.

In this paper, we propose a radix-4 Booth multiplier with pre-encoded mechanism to improve the power efficiency of multiplication. One specific feature of the radix-4 Booth algorithm is that when the continuous three bits of multiplier \( Y(y_{2i+1}, y_{2i}, y_{2i−1}) \) have the same values, the corresponding PPs will be 0. This feature inspired us to find the “0X” case earlier to reduce the unnecessary switching activities of the radix-4 Booth encoders and decoders. Thus, we propose the pre-encoded scheme to detect the “0X” case before every multiplication. When the “0X” case occurs, the proposed pre-encoder will turn off the Booth encoders and decoders to save power, and set the corresponding PPs to 0 directly before the starting of multiplication. The proposed design is simulated with Taiwan Semiconductor Manufacturing Company (TSMC) 40 nm technology. Compared with the traditional design [16] and the related designs [18], [19], [22], the simulation results show that the proposed design outperforms these designs in terms of transistor count, delay, and power consumption.

The rest of this paper is organized as follows. Section II reviews the traditional radix-4 Booth multiplier and the related works. Section III describes the low power radix-4 Booth multiplier with pre-encoded mechanism in detail. Section IV shows the simulation results of the proposed design. Section V offers a brief conclusion of this paper.

### II. TRADITIONAL RADIX-4 BOOTH MULTIPLIER AND RELATED WORKS

Multiplication is a basic arithmetic operation; many DSP and machine learning applications are highly multiply-intensive [1]–[4]. Therefore, the power consumption and performance issues of the multiplier are important. However, the traditional array multiplier generates a lot of PPs \((n \times n\) multiplication has \(n\) PP rows\) and accumulates all PPs to get the final product; it consumes huge power and is not power efficient. Then, the Booth algorithm (radix-2 Booth algorithm) has been proposed to improve the performance of the multiplication [23]; the radix-4 Booth algorithm (also called modified Booth algorithm) [24] can reduce the number of PP rows by half to facilitate the multiplication. In this section, we introduce the traditional radix-4 Booth algorithm and the related works.

#### A. TRADITIONAL RADIX-4 BOOTH ALGORITHM

The radix-4 Booth algorithm is a powerful method to improve the performance of multiplication and applies to two’s complement operands. The radix-4 Booth algorithm partitions the multiplier \( Y(y_{n−1}y_{n−2} \ldots y_0) \) into overlapping groups of contiguous three bits. Each group is encoded and then decoded with multiplicand \( X(x_{n−1}x_{n−2} \ldots x_0) \) to generate the corresponding PPs. The \( n\)-bit multiplication of radix-4 Booth algorithm can be expressed as follows:

\[
X \times Y = X \times \left( -y_{n−1}2^{n−1} + \sum_{i=0}^{n−2} y_i2^i \right)
= X \times \sum_{i=0}^{(\frac{n}{2})−1} \left( -2y_{2i+1}y_{2i}y_{2i−1} \right) 2^{2i}
= X \times \sum_{i=0}^{(\frac{n}{2})−1} M_i2^{2i}X
= \sum_{i=0}^{(\frac{n}{2})−1} M_i2^{2i}X
\]

We refer to (1), the radix-4 Booth algorithm can reduce the number of PP rows by half, and one specific feature is that the corresponding PPs will be 0s when the continuous three bits of multiplier \( Y(y_{2i+1}, y_{2i}, y_{2i−1}) \), the radix-4 Booth algorithm can generate the corresponding coefficient \( M_i \). The coefficient \( M_i \) has five possible values (±1, ±2, or 0) as shown in Table I. Refer to (1), the radix-4 Booth algorithm can reduce the number of PP rows by half, and one specific feature is that the corresponding PPs will be 0s when the continuous three bits of multiplier \( Y(y_{2i+1}, y_{2i}, y_{2i−1}) \) have the same values. As shown in Table I, the continuous three bits of multiplier \( Y(y_{2i+1}, y_{2i}, y_{2i−1}) \) have the same values that indicate the “0X” case. Notice that \( y_{−1} \) is always 0 when \( i \) equals to 0.

Although the radix-4 Booth algorithm can reduce the number of PPs, the cost of the radix-4 Booth multiplier is still high. Reference [12] provided a modified sign extension structure which can reduce the number of PPs and the cost further. Fig. 1 shows the traditional radix-4 Booth multiplier with the modified sign extension structure. Each PP row has one Booth encoder and \( n+1 \) Booth decoders. Booth encoders and decoders generate the PPs; full adders and carry-propagation adder add all PPs to get the final product. However, the unnecessary glitches and switching activities occur in the traditional radix-4 Booth multiplier because of
The traditional $8 \times 8$ radix-4 Booth multiplier with the modified sign extension structure.

The unbalanced signal paths of radix-4 Booth encoders and decoders. That leads to high power consumption [16]-[18]. The overall power consumption of CMOS can be defined as

$$P = aCV_{dd}^2f + V_{dd}I_{leakage}$$  \hfill (2)

The first and second terms are dynamic and static power consumption, respectively. Where $a$ is the switching activity parameter, $C$ is the total capacitance load, $V_{dd}$ is the supply voltage, $f$ is the operating frequency, and $I_{leakage}$ is the leakage currents [25], [26]. According to (2), the power consumption can be reduced by minimizing the switching activities.

**B. RELATED WORKS**

The radix-4 Booth encoder and decoder of [11] is the most common implementation of the radix-4 Booth algorithm, and the authors of [13]-[15] provided more compact implementations. However, the author of [16] found these designs have unnecessary glitches caused on PPs. Therefore, the glitch-free Booth encoder and partial product generator (decoder) have proposed to reduce the unnecessary glitches. In [16], the propagation delay from inputs X and Y to PP is only two units (one XOR/XNOR and the complex gate for output), and all paths almost have the same propagation delay. In [17], the authors proposed a high performance and low cost radix-4 Booth decoder that also keeps the advantage of race-free. Especially, the cost of [17] is less than the cost of [16].

In [18], the authors proposed a neg/two/one-nf (nf means neg-first) generator as shown in Fig. 2, and the neg-first radix-4 Booth algorithm is shown in Table 2. The neg-first radix-4 Booth algorithm is the three-signal scheme; signals $one_i$, $two_i$, and $neg_i$ can indicate that ($y_{2i+1}$, $y_{2i}$, $y_{2i-1}$) belongs to which case. The signal $neg_i$ is for negation operation and $cor_i$ is the correction bit for negative operation. The neg-first means that the negation operation is done before the selection between “1X” and “2X”. For generating PP, the neg/two/one-nf decoder adopts the OR-AND-INV (OAI) gate, and all input signals of this OAI gate almost arrive at the same time (about 1 XNOR gate delay) as shown in Fig. 2. Therefore, the signal paths are more balanced than other schemes and the glitches can be reduced. However, in the case of “−0X”, $neg_i = y_{2i-1}$ may lead to more switching activities in the XNOR gate since signals $one_i$ and $two_i$ can set PP to 0 regardless of $neg_i$. Note that the neg-first design needs one more compact decoder to generate the first $nx$ signal and this decoder no needs to generate the PP as shown in Fig. 3. Therefore, each PP row of the neg-first design has one encoder and $n + 2$ decoders.

The NMBE scheme [19] which is based on the PTL has less cost than traditional design. But, this scheme suffers from the problems of weak 1/strong 0 and the signal paths are not balanced. Research [22] provided three approximation techniques for radix-4 Booth multipliers and one of them is called approximate Booth multipliers models 1 (ABM-M1). ABM-M1 is composed of the exact partial product generators (radix-4 Booth encoder and encoder) and the approximate

![FIGURE 1. The traditional $8 \times 8$ radix-4 Booth multiplier with the modified sign extension structure.](image-url)

**TABLE 2. The neg-first radix-4 booth algorithm [18].**

| $y_{2i+1}$ | $y_{2i}$ | $y_{2i-1}$ | Operation on X | $neg_i$ | $two_i$ | $one_i$ | $cor_i$ |
|----------|---------|-----------|----------------|--------|--------|--------|--------|
| 0        | 0       | 0         | +0X            | 0      | 0      | 0      | 0      |
| 0        | 0       | 1         | +1X            | 0      | 0      | 1      | 0      |
| 0        | 1       | 0         | +1X            | 0      | 0      | 1      | 0      |
| 0        | 1       | 1         | +2X            | 0      | 1      | 0      | 0      |
| 1        | 0       | 0         | −2X            | 1      | 1      | 0      | 1      |
| 1        | 0       | 1         | −1X            | 1      | 0      | 1      | 1      |
| 1        | 1       | 0         | −1X            | 1      | 0      | 1      | 1      |
| 1        | 1       | 1         | −0X            | 1      | 0      | 0      | 0      |
Y.-J. Chang et al.: Low Power Radix-4 Booth Multiplier With Pre-Encoded Mechanism

III. THE PROPOSED RADIX-4 BOOTH MULTIPLIER WITH PRE-ENCODED MECHANISM

In this paper, we propose a pre-encoded mechanism to reduce the power consumption of the radix-4 Booth multiplier. As mentioned above, the unnecessary switching activities make multiplier consume more power, and the PPs must be 0s in the “0X” case. For that reason, we propose a pre-encoded mechanism to find the “0X” case earlier to reduce the unnecessary switching activities of the radix-4 Booth encoders and decoders. When detects the “0X” case, the proposed pre-encoded mechanism can turn off Booth encoders and decoders immediately to save power.

The architecture and the timing chart of the proposed pre-encoded mechanism are shown in Fig. 4 and Fig. 5, respectively. As shown in Fig. 4, the proposed pre-encoded mechanism is composed of multiplicand and multiplier registers, additional proposed pre-encoders, and radix-4 Booth multiplier (includes adders, the proposed low cost Booth encoders and decoders). The signals $x_{n-1},x_{n-2},\ldots,x_0,x_{n-1},x_{n-2},\ldots,x_0$ and $y_{n-1},y_{n-2},\ldots,y_0,y_{n-1},y_{n-2},\ldots,y_0$ denote the outputs of the multiplicand and multiplier registers. As shown in Fig. 5, the multiplicand X and multiplier Y would be set on the data bus during the data setup time before the multiplication (multiplication phase). The proposed mechanism can detect the “0X” case during this setup time (denotes as pre-encode phase). In order to detect the “0X” case before the multiplication, the proposed design needs additional pre-encoders. If the “0X” case occurs, the proposed pre-encoders will immediately turn off the corresponding Booth encoders and decoders, and the corresponding PPs will be set as 0s to reduce the switching activities. In contrast, the Booth encoders and decoders will work as usual. Because the pre-encoders process the “0X” case already, the proposed Booth encoders and decoders only need to process the “±1X” and “±2X” cases, then, the cost of our design can be reduced. Accordingly, the proposed pre-encoded mechanism has less cost than the other designs.

A. THE PROPOSED PRE-ENCODER

The proposed pre-encoded mechanism needs pre-encoder to detect the “0X” case in the pre-encode phase. Table 3 shows the proposed pre-encoded radix-4 Booth algorithm. The proposed pre-encoded mechanism has three encoded signals. Signal zero, which is generated by the proposed pre-encoder can determine the three bits of multiplier Y on the data bus are the same or not. Signals neg and ot are generated
by the proposed encoder. Signal neg, is for the negation operation; signals neg, and ot, are for the remaining cases ("±1X" and "±2X"). Note that signal zero, has the highest priority, cor, is the correction bit for negative operation, and d means "don’t care".

According to Table 3, when \( y_{2i+1}, y_{2i} \), and \( y_{2i-1} \) on the data bus have the same value, the "0X" case occurs, thus, the signal \( z_{0i} \) will be 1. The equation of \( z_{0i} \) can be written as

\[
\begin{align*}
\text{zero}_i = (y_{2i+1} \cdot y_{2i} \cdot y_{2i-1})_{\text{db}} + (y_{2i+1} \cdot y_{2i} \cdot y_{2i-1})_{\text{db}} \\
\end{align*}
\]

where \( \text{db} \) means that the \( y_{2i+1, y_{2i}, y_{2i-1}} \) arrive at the data bus in the pre-encode phase. According to (3), the proposed pre-encoder can be implemented with the AND-OR-INV (AOI) gate and a latch-like circuit as shown in Fig. 6(a). Fig. 6(b) shows the proposed pre-encoder at the transistor level. In the pre-encode phase, when the multiplier \( Y \) arrives on the data bus, the pre-encoder starts to detect the "0X" case. The signal \( \text{zero}_i \) is generated to control the proposed low cost encoder and decoders (introduce in section III-B) of the \( i \)th PP row; the encoder and decoders work as usual or not in the multiplication phase according to the signal \( \text{zero}_i \).

1) "Non-0X" cases: In the pre-encode phase, if the \( y_{2i+1_{\text{db}}, y_{2i_{\text{db}}, y_{2i-1_{\text{db}}}}} \) are not the same, the signal \( \text{zero}_i \) will be set to 0 by the proposed pre-encoder. Therefore, the proposed encoder and decoders of the \( i \)th PP row will work as normal to generate the corresponding PPs in the multiplication phase.

2) "0X" case: In the pre-encode phase, if the \( y_{2i+1_{\text{db}}, y_{2i_{\text{db}}, y_{2i-1_{\text{db}}}}} \) are the same, the signal \( \text{zero}_i \) will be set to 1 by the proposed pre-encoder. Thus, the proposed encoder and decoders of the \( i \)th PP row will be powered off to reduce power consumption, and the corresponding PPs will be set to 0s directly in the pre-encode phase. In the multiplication phase, these gated encoder and decoders are no need to work since they are already turned off and the corresponding PPs are already set to 0s.

Note that the latch-like circuit shown in Fig. 6 is added to prevent the probably happened unnecessary switching activities when "0X" case changes to "non-0X" case. Fig. 7 shows the latch-like circuit operations in this special situation. As shown in Fig. 7(a), when "0X" case changes to "non-0X" case (signal \( \overline{Z}_i \) changes from 0 to 1) in the pre-encode phase, the pre-encoder without the latch-like circuit generates \( \text{zero}_i \) as 0 immediately that makes encoder and decoders turn on to do the redundant multiplication of the previous time ("0X" case). To avoid this redundant multiplication, the latch-like circuit is required. As shown in Fig. 7(b), when "0X" case changes to "non-0X" case in the pre-encode phase, \( \text{Clk} \) is 0, and N2 is turned off. The pre-encoder with the latch-like circuit makes \( \text{zero}_i \) keep the previous value at a high voltage level to turn off the encoder and decoders, then, the redundant multiplication can be avoided. However, until \( \text{Clk} \) changes from 0 to 1, N2 is turned on and the pre-encoder with latch-like circuit makes \( \text{zero}_i \) as 0 through N0 and N2 to power on the encoder and decoders; the "non-0X" multiplication starts normally in the multiplication phase.

B. THE PROPOSED LOW COST RADIX-4 BOOTH ENCODER AND DECODER

Because the "0X" case has been processed by the proposed pre-encoder, the encoder and decoder only need to process the remaining cases ("±1X" and "±2X"). For that reason, the costs of radix-4 Booth encoder and decoder can be reduced.

In this paper, we propose the low cost radix-4 Booth encoder and decoder as shown in Fig. 8 and Fig. 9 to reduce the power consumption further. The gating techniques (power gating and ground gating) and the low cost of our design can reduce the dynamic and also static power consumption effectively. In particular, the issues of reducing static power consumption become more and more important when technology progresses [26], [27]. Table 4 shows a summary of the proposed pre-encoded radix-4 Booth algorithm. Signal \( \text{zero}_i \) can be used as the control signal of the gating transistors that are added in the proposed encoder and decoder. When \( \text{zero}_i \) is 1, the proposed encoder and decoder will be gated to reduce

![FIGURE 6. The proposed pre-encoder. (a) gate level; (b) transistor level.](image)

![FIGURE 7. The difference between the proposed pre-encoder without and with the latch-like circuit. (a) without the latch-like circuit; (b) with the latch-like circuit in the pre-encode phase; (c) with the latch-like circuit in the multiplication phase.](image)
power consumption. In contrast, the proposed encoder and decoder will work as usual.

Fig. 8(a) and Fig. 8(b) show the details of the proposed encoder. Like the traditional design, the inputs \(y_{2i+1}, y_{2i},\) and \(y_{2i-1}\) of the proposed encoder are the outputs of the corresponding multiplier registers. Note that the inputs of the proposed pre-encoder are the multiplier \(Y\) on the data bus since the pre-encoder needs to detect the “0X” case earlier in the pre-encode phase. Because of this pre-encoder, the proposed encoder only needs to generate signals \(\text{neg}_i\) and \(\text{ot}_i\). According to Table 3, the expressions of \(\text{neg}_i\) and \(\text{ot}_i\) are

\[
\text{neg}_i = y_{2i+1} \quad (4)
\]
\[
\text{ot}_i = y_{2i-1} \oplus y_{2i} \quad (5)
\]

Signal \(\text{neg}_i\) is equal to \(y_{2i+1}\) and signal \(\text{ot}_i\) can be generated by an XOR gate according to (4) and (5), respectively. The XOR gate can be implemented with the CMOS logic [28]. To reduce the power consumption in the “0X” case, the XOR gate of the proposed encoder adopts the gating techniques. As shown in Fig. 8(b), the gating transistor \(P_{E0}\) is added between the power supply and the XNOR logic, and is controlled by \(\text{zero}_i\). The gating transistor \(N_{E0}\) is added between GND and the XNOR logic, and is controlled by \(\overline{\text{zero}_i}\). Signals \(\text{zero}_i\) and \(\overline{\text{zero}_i}\) are generated by the proposed pre-encoder as introduced above.

1) “Non-0X” cases: In the pre-encode phase, if the \(y_{2i+1}, y_{2i},\) and \(y_{2i-1}\) are not the same, the signal \(\text{zero}_i\) is 0. Thus, the gating transistors \(P_{E0}\) and \(N_{E0}\) of the proposed encoder will be turned off to reduce the dynamic power consumption and leakage currents. In the multiplication phase, the proposed encoder still stays in the standby mode to save power.

According to Table 3, the expression of correction bit \(\text{cor}_i\) can be written as

\[
\text{cor}_i = y_{2i+1} \odot \overline{\text{zero}_i} \quad (6)
\]

Based on (6), the correction bit can be generated by an AND gate that is simpler than the circuitry in [18]. The correction bit \(\text{cor}_i\) is 0 in the “0X” case; otherwise, the value of \(\text{cor}_i\) depends on \(y_{2i+1}\) (\(\text{neg}_i\)) as summarized in Table 4.

Fig. 9(a) and Fig. 9(b) show the details of the proposed decoder. Like the traditional design, the input \(x_j\) of the proposed decoder is connected with the output of the corresponding multiplicand register. According to the encoded signals of the proposed pre-encoder and encoder, the proposed decoder can generate the corresponding PP. As introduced before, the PP should be 0 when the “0X” case occurs. The PP of “+1X” is \(x_j\) and the PP of “-1X” is \(\overline{x}_j\). The PP of “+2X” is \(x_{j-1}\) and the PP of “-2X” is \(\overline{x}_{j-1}\). Table 4 summarizes the PP value of each case; the PP values of “+1X” and “+2X” can be defined as (7) and (8), respectively.

\[
PP_{\pm 1X} = \text{neg}_i \oplus x_j = \text{nx}_j \quad (7)
\]
\[
PP_{\pm 2X} = \text{neg}_i \oplus x_{j-1} = \text{nx}_{j-1} \quad (8)
\]

Based on Table 4, the expression of PP value can be written as

\[
PP_{ij} = \text{zero}_i \cdot 0 + \overline{\text{zero}_i} \cdot (\text{ot}_i \cdot \text{nx}_j + \overline{\text{ot}_i} \cdot \text{nx}_{j-1}) \quad (9)
\]

According to (9), the proposed decoder can be composed of an XOR gate and two multiplexers (MUXs). The XOR gate
is implemented with the CMOS logic [28] and its output is shared with the neighbor decoder. The first MUX which is controlled by \( o_{i1} \) is implemented by the transmission gates (TGs); the second MUX which is controlled by \( z_{o1} \) can be implemented easily by an NMOS. Like the proposed encoder, the proposed decoder adopts the gating techniques. As shown in Fig. 9(b), the gating transistor \( P_{D0} \) which is controlled by \( z_{o1} \) is added between the power supply and the XOR logic; the gating transistor \( N_{D0} \) which is controlled by \( \overline{z_{o1}} \) is added between GND and the XOR logic.

1) "Non-0" cases: In the pre-encode phase, the signal \( z_{o1} \) is 0 when the \( y_{2i+1}, y_{2i}, \) and \( y_{2i-1} \) are not the same. In the multiplication phase, the gating transistors \( P_{D0} \) and \( N_{D0} \) of the proposed decoder are turned on to generate the signal \( nX \) as normal. Therefore, the proposed decoder can generate the corresponding PP according to the encoded signals \( o_{i1} \) and \( z_{o1} \).

2) "0" case: In the pre-encode phase, the signal \( z_{o1} \) is 1 when the \( y_{2i+1}, y_{2i}, \) and \( y_{2i-1} \) are the same. Since \( z_{o1} \) is 1, the PP will be set to 0 directly in the pre-encode phase. In addition, the gating transistors \( P_{D0} \) and \( N_{D0} \) of the proposed encoder are turned off to reduce the dynamic power consumption and leakage currents. In the multiplication phase, the PP is 0, and the proposed decoder still stays in standby mode to save power.

As shown in Fig. 10, each PP row of the proposed pre-encoded mechanism has one pre-encoder, one encoder, and \( n + 2 \) decoders (10 decoders for \( n = 8 \)) that is similar to [18]. Clearly, the decoders account for the majority of the circuitry cost of each PP row. To minimize the cost of the proposed decoders, the additional gating transistors can be shared. As shown in Fig. 10, every five decoder shares a set of gating transistors. Take 8-bit multiplication \( (n = 8) \) for example, there are 10 decoders including the compact decoder in each PP row, and these decoders can be divided into two groups. In the first group, decoder0 to decoder3 share the gating transistors \( P_{D1} \) and \( N_{D1} \). When \( z_{o1} \) is 0, these two groups work as usual to generate the corresponding PPs. When \( z_{o1} \) is 1, these two groups will be turned off to reduce power consumption and set PPs of this row to 0s immediately.

Table 5 summarizes the comparisons for the proposed pre-encoded design and the related designs [16], [18], [19], [22]. The traditional design [16] is the four-signal scheme and have balanced signal propagation paths. But, the cost of traditional design is higher than the others. The NMBE design [19] is the three-signal scheme and based on the PTL. The cost of the NMBE design [19] is low, but the signal paths are not balanced. Moreover, the NMBE design [19] suffers from the problems of weak 1/strong 0 that makes the static power increase. In order to reduce the circuit complexity, the ABM-M1 design [22] adopted the approximate decoders (PPG-2S) in some least-significant bits. The encoder of [22] has 50 transistors which is larger than the others since this encoder needs to encode for the exact and approximate decoders. The ABM-M1 design [22] can only be used in the error-tolerant applications. Same as the neg-first design [18], the proposed design has three encoded signals and balanced signal propagation paths. However, the neg-first design [18] has the "−0X" case that may lead to more switching activities in XNOR gates of decoders. For \( n \)-bit multiplication of design [18] and the proposed design, one PP row needs one encoder and \( n + 2 \) decoders (one additional pre-encoder for the proposed design). Obviously, the proposed design has less cost than the others even though the proposed design needs the additional pre-encoder. Because of the pre-encoder, the proposed design can turn off the encoder and decoders to save power in the "0X" case. Therefore, we expect that the proposed design has the better power efficiency than the other designs.

### IV. SIMULATION RESULTS

In this paper, the related works [16], [18], [19], [22], and the proposed pre-encoded mechanism are simulated by using TSMC 40 nm CMOS technology. The supply voltage is 1.0V, the clock frequency is 100 MHz, and the simulation is done by HSPICE tool. We simulate generating one PP row of \( n \)-bit multiplication \( (n = 8 \text{ or } n = 16) \); we provide the power consumption, the performance, and transistor count (TC) to show the effectiveness of the proposed pre-encoded mechanism. We also provide the overall comparisons of multipliers to prove the superiority of the proposed design over the related works.

#### A. FUNCTIONALITY

To verify the feasibility and correctness of the proposed pre-encoded mechanism, the TSMC 40 nm technology is used to simulate two specific scenes with HSPICE. One specific scene is changing from "0X" case to "non-0X" case and the other is changing from "non-0X" case to "0X" case. Fig. 11 and Fig. 12 show the waveforms of these two specific scenes. The black solid line is the clock signal \( Clk \), the blue solid
TABLE 5. Comparisons for the radix-4 booth algorithms.

| Features of Algorithm | The traditional design [16] | The neg-first design [18] | The NMBE design [19] | The ABM-M1 design [22] | The proposed design |
|-----------------------|-------------------------------|---------------------------|----------------------|------------------------|---------------------|
|                      | 4 encoded signals: neg, X1, X2P, and ZP | 3 encoded signals: neg, two, and one neg-first case | 3 encoded signals: n, s, and z based on PTL strong 0 weak 1 | 3 encoded signals: neg, two, and zero composed of the exact and approximate PP generators | 3 encoded signals: zero, neg, and ot zero pre-encoded no “−0X” case n + 2 decoders per row adopts gating techniques |
|                      | n + 1 decoders per row did not adopt gating techniques | n + 2 decoders per row did not adopt gating techniques | n + 1 decoders per row did not adopt gating techniques | n + 1 decoders per row did not adopt gating techniques |
| Pre-encoder          | 34 transistors                | 30 transistors            | 20 transistors       | 40 transistors         | 18 transistors (with gating transistors) |
| Encoder              | 40 transistors                | 16 transistors            | 18 transistors       | 14 transistors         | 15 transistors (with gating transistors) |
| Decoder              | balance                       | balance                   | unbalance            | balance               | race-free design |
| Propagation paths    | race-free design              | race-free design          | Approximate (PPG-2S) | Approximate (PPG-2S)  |race-free design |

Fig. 11 shows the waveforms of “0X” case changing to “non-0X” case. We choose i = 1 and j = 1 to describe the waveforms in detail. Suppose that the multiplicand X is fixed at 01010101 and the continuous three bits of multiplier Y (y3, y2, y1) change from (0, 0, 0) to (1, 1, 0), that is, “0X” case changes to “−1X” case. In the beginning, y3, y2, y1 = (0, 0, 0), thus, zero1 is 1 and PP11 is 0. When y3, y2, y1 are set to 1 (y3 and y2 are still 0) in the pre-encode phase, the pre-encoder output zero1 keeps the high voltage to prevent unnecessary switching activities as introduced in section III-B until Clk changes from 0 to 1. When Clk changes from 0 to 1 (the multiplication phase), registers store the new data from the data bus, and zero1 changes from 1 to 0 to make the encoder and decoders work as normal. PP11 changes from 0 to 1 because y3 (neg1) is 1, x1 is 0, and zero1 is 0.

Fig. 12 shows the waveforms of “non-0X” case changing to “0X” case. We choose i = 1 and j = 1 to describe the waveforms in detail. Suppose that the multiplicand X is
fixed at 01010101 and the continuous three bits of multiplier \( Y (y_3, y_2, y_1) \) change from (0, 1, 1) to (0, 0, 0), that is, “+2X” case changes to “0X” case. In the beginning, \( (y_3, y_2, y_1) = (0, 1, 1) \), therefore, \( \text{zero}_0 \) is 0 and \( PP_{11} \) is 1 (\( y_3 = 0 \) and \( x_0 = 1 \)). When \( y_2 \_db \) and \( y_1 \_db \) are set to 0 (\( y_2 \) and \( y_1 \) are still 1) in the pre-encode phase, the pre-encoder output \( \text{zero}_0 \) is set to 1 directly to turn off the encoder and decoders. Notice that \( PP_{11} \) is also set to 0 immediately in the pre-encode phase because the pre-encoder detects the “0X” case. When \( Clk \) changes from 0 to 1 (the multiplication phase), registers store the new data from the data bus, and the encoder and decoders are turned off already to reduce power consumption. Fig. 11 and Fig. 12 prove the correctness and feasibility of the proposed pre-encoded design.

**B. POWER CONSUMPTION ANALYSIS**

The proposed pre-encoded design is compared with the related works [16], [18], [19], [22]; we simulate generating one PP row with 8-bit multiplication and 16-bit multiplication for each design. The continuous three bits of multiplier \( Y (y_{2i+1}, y_{2i}, y_{2i-1}) \) change from one pattern to each pattern. We simulate these pattern switches with the 100 MHz clock frequency and provide the average power consumption for each pattern. The power consumption of the proposed design is composed of the power consumption of pre-encoder, encoder, and decoders. Notice that we simulate one row of the ABM-M1 design [22] with the approximation factor \( m = 4 \) (\( m = 8 \)) for the 8-bit multiplication (16-bit multiplication). When \( m = 4 \),
Table 6. Comparisons of the radix-4 booth designs for generating one partial product row.

|                  | One PP row |                          |                          |                          |                          |                          |
|------------------|------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|                  |            | The traditional design [16] | The neg-first design [18] | The NMBE design [19]     | The ABM-M1 design [22]   | The proposed design     |
|                  |            | Average dynamic power (\(W\)) | Average static power (\(W\)) | Average dynamic power (\(W\)) | Average static power (\(W\)) | Average dynamic power (\(W\)) |
|                  |            | Pre-encoder               | Encoder                   | Decoders                  | Total                     | Encoder                   |
|                  |            | 9.95E-07                  | 6.42E-07                  | 1.17E-06                  | 6.12E-06                  | 1.28E-06                  |
|                  |            | Pre-encoder               | Encoder                   | Decoders                  | Total                     | Encoder                   |
|                  |            | 6.23E-06                  | 6.20E-06                  | 3.19E-06                  | 1.81E-06                  | 2.93E-07                  |
|                  |            | Pre-encoder               | Encoder                   | Decoders                  | Total                     | Encoder                   |
|                  |            | 7.12E-08                  | 6.03E-08                  | 1.22E-07                  | 6.94E-08                  | 1.30E-07                  |
|                  |            | Pre-encoder               | Encoder                   | Decoders                  | Total                     | Encoder                   |
|                  |            | 6.69E-07                  | 6.03E-08                  | 2.15E-07                  | 6.94E-08                  | 2.11E-07                  |
|                  | 8-bit      | Pre-encoder               | Encoder                   | Decoders                  | Total                     | Encoder                   |
|                  |            | 7.40E-07                  | 2.75E-07                  | 3.55E-07                  | 2.11E-07                  | 0.0774                    |
|                  |            | Delay (ns)                | 0.0795                    | 0.0791                    | 0.1087                    | Exact: 0.0774             |
|                  |            | Transistor count (TC)     | 394                       | 182                       | 182                       | 196                       |
|                  |            | 16-bit                   | Pre-encoder               | Encoder                   | Decoders                  | Total                     | Encoder                   |
|                  |            | 8.34E-08                  | 6.98E-08                  | 1.22E-05                  | 7.28E-08                  | 1.21E-05                  |
|                  |            | Pre-encoder               | Encoder                   | Decoders                  | Total                     | Encoder                   |
|                  |            | 1.31E-06                  | 9.68E-07                  | 2.01E-07                  | 9.68E-08                  | 2.69E-07                  |
|                  |            | Pre-encoder               | Encoder                   | Decoders                  | Total                     | Encoder                   |
|                  |            | 1.39E-06                  | 4.64E-07                  | 2.16E-06                  | 6.08E-07                  | 3.37E-07                  |
|                  |            | Delay (ns)                | 0.1142                    | 0.0944                    | 0.1399                    | Exact: 0.0980             |
|                  |            | Transistor count (TC)     | 714                       | 310                       | 326                       | 332                       |

\(m = 8\), one row of the ABM-M1 design [22] is composed of one encoder, five (nine) exact decoders, and four (eight) PPG-2Ss.

Fig. 13 shows the average dynamic power consumption for each pattern switch. For instance, “000” in Fig. 13 represents the average power consumption for changing from each pattern to pattern “000”. Obviously, the dynamic power consumption of the proposed design is less than the power consumption of the other designs for each pattern switch because the proposed design has the advantages of less cost and race-free. The dynamic power consumption of the NMBE design [19] is larger than the traditional design [16] in some cases (especially “000”) because of the drawbacks of the poor voltage level. Table 6 summarizes the average static power consumption of the eight patterns for one PP row. The NMBE design [19] still has the largest static power consumption. For 8-bit (16-bit) case, compared to the traditional design [16] and the neg-first design [18], the proposed design can reduce the static power consumption by 71.5% (75.8%) and 23.3% (27.4%), respectively; compared to the approximate design [22], the proposed design can reduce the static power consumption by 40.6% (44.6%).

### C. PERFORMANCE AND COST ANALYSES

The performance and cost analyses for one PP row are summarized in Table 6. The worst case delay is the metric to evaluate the performance. For example, the worst case delay of the neg-first design [18] will occur when signal \(\text{two}_{i}\) changes from 0 to 1 or 1 to 0. The worst case delay of the proposed design will occur when the encoded signals change from “0X” to “non-0X” case. As shown in Table 6, the NMBE design [19] has the longest delay due to the weak drivability of signals. For 8-bit (16-bit) case, compared to the traditional design [16] and the neg-first design [18], the delay reductions of the proposed design are 20.9% (34.4%) and 20.5% (20.7%), respectively. As shown in Table 6, the worst case delay of the ABM-M1 design [22] is contributed by the exact decoder. Compared to the ABM-M1 design [22], the delay reduction of the proposed design is 18.7% (23.6%).
TABLE 7. Comparisons of the n-bit radix-4 booth multipliers.

|                      | Average dynamic power (W) | Average static power (W) | Delay (ns) | PDP (μJ) | Transistor count (TC) |
|----------------------|---------------------------|--------------------------|------------|----------|-----------------------|
| **8-bit Radix-4 Booth Multiplier** |                           |                          |            |          |                       |
| The traditional design [16] | 4.24E-05                  | 4.63E-06                 | 0.4135     | 17.53    | 2772                  |
| The neg-first design [18] | 2.53E-05                  | 2.41E-06                 | 0.4131     | 10.45    | 1924                  |
| The NMBE design [19]      | 3.92E-05                  | 1.28E-05                 | 0.4427     | 17.35    | 1924                  |
| The ABM-M1 design [22]    | 3.24E-05                  | 2.90E-06                 | 0.4114     | 13.33    | 2040                  |
| The proposed design       | 2.38E-05                  | 1.78E-06                 | 0.3969     | 9.45     | 1852                  |
| **16-bit Radix-4 Booth Multiplier** |                           |                          |            |          |                       |
| The traditional design [16] | 1.96E-04                  | 1.68E-05                 | 0.7818     | 153.23   | 9972                  |
| The neg-first design [18] | 1.16E-04                  | 8.27E-06                 | 0.7620     | 88.39    | 6740                  |
| The NMBE design [19]      | 1.86E-04                  | 2.94E-05                 | 0.8075     | 150.20   | 6868                  |
| The ABM-M1 design [22]    | 1.50E-04                  | 9.92E-06                 | 0.7656     | 114.84   | 7180                  |
| The proposed design       | 1.08E-04                  | 5.84E-06                 | 0.7424     | 80.18    | 6436                  |

Table 6 also provides the transistor count (TC) to evaluate the costs of each design. For one n-bit PP row, different from the other designs, both of the neg-first design [18] and the proposed design need n+2 decoders (one of them is the compact decoder). Take 8-bit for example, the neg-first design [18] needs 182 transistors for one PP row which is composed of one 30T encoder, one 8T compact decoder, and nine 16T decoders. The proposed design needs 164 transistors for one PP row which is composed of one 17T pre-encoder, one 18T encoder, one 8T compact decoder, nine 13T decoders, and four shared gating transistors (two groups). As shown in Table 6, the traditional design has the largest TC; the other designs can effectively reduce the costs and the proposed design has the least TC.

D. COMPARISON OF RADIX-4 BOOTH MULTIPLIERS

In order to show the effectiveness of our design, we also provide the comparisons of the radix-4 Booth multipliers as shown in Table 7. For a fair comparison, each design adopts the same adder array for PP accumulation as shown in Fig. 1. Only the multiplier of the ABM-M1 design [22] is the approximate multiplier, and the approximation factor \( m = 4 \) for 8-bit multiplication \( (m = 8 \) for 16-bit multiplication). Take 16-bit multiplication with \( m = 8 \) for example, the PPs with a significance less than 8 are generated by PPG-2Ss and the remaining PPs are generated by the exact decoders.

Compared to the traditional design [16], the other designs can reduce the dynamic power consumption and TC. However, the NMBE design [19] is the worst in terms of static power consumption and delay because of the drawbacks of the poor voltage level that increases the leakage currents and delay. Because the proposed pre-encoded design has the advantages of race-free, conditional power gating, and low cost, the proposed design is the best design in terms of dynamic power, static power, delay, power-delay product (PDP), and TC as shown in Table 7. For 8-bit (16-bit) case, compared to the traditional design [16], the proposed design can reduce the power consumption by 43.9% (44.9%) and 61.6% (65.2%), respectively, with 4.0% (5.0%) performance improvement. Compared to the neg-first design [18], the proposed design can reduce the dynamic power and static power by 5.9% (6.9%) and 26.1% (29.4%), respectively. Compared to the approximate design [22], the proposed design can provide the exact results, and can reduce the dynamic power and static power by 26.5% (28.0%) and 38.6% (41.1%), respectively. Obviously, the proposed design has the lowest PDP results. Fig. 13, Fig.14, Table 6, and Table 7 prove the superiority of the proposed design compared to the other designs.

V. CONCLUSION

In this paper, a low power radix-4 Booth pre-encoded mechanism has been proposed to reduce the unnecessary switching activities of encoders and decoders in the “0X” case. The proposed pre-encoded mechanism can detect the “0X” case earlier and adopts the gating techniques. When the “0X” case occurs, the encoder and decoders will be turned off immediately by the proposed pre-encoder to reduce the power consumption and leakage currents. The simulations are done by HSPICE with the TSMC 40 nm technology and the results show that the proposed design can provide significant reductions in power consumption, delay, and transistor count compared with the state-of-the-art encoded designs. For 16-bit multiplication, compared to the traditional radix-4 Booth multiplier, the proposed pre-encoded mechanism has 35% reduction in transistor count, 5% improvement in performance, and can reduce dynamic and static power consumption by 45% and 65%, respectively. Compared to the neg-first design and the NMBE design, the proposed design has better performance, less transistor count, and lower power consumption. Even compared to the approximate design, the proposed design can provide precise products, and can achieve 28% dynamic power reduction and 41% static power reduction.

REFERENCES

[1] G.-N. Sung, Y.-J. Ciou, and C.-C. Wang, “A power-aware 2-dimensional bypassing multiplier using cell-based design flow,” in Proc. IEEE Int. Symp. Circuits Syst., May 2008, pp. 3338–3341.

[2] G.-L. Jiang, T.-C. Wu, and Y.-J. Chang, “Low power multiplier with alternative bypassing implementation,” in Proc. Int. Conf. Embedded Syst. Appl. (ESA), Jul. 2012, pp. 77–82.
[3] X. Cui, W. Liu, X. Chen, E. E. Swartzlander, and F. Lombardi, “A modified partial product generator for redundant binary multipliers,” IEEE Trans. Comput., vol. 65, no. 4, pp. 1165–1171, Apr. 2016.

[4] J. M. D. Moss, D. Boland, and H. W. P. Leong, “A two-speed, radix-4, serial-parallel multiplier,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 27, no. 4, pp. 769–777, Dec. 2018.

[5] B. Moons and M. Verhelst, “An energy-efficient precision-scalable ConVNet processor in 40-nm CMOS,” IEEE J. Solid-State Circuits, vol. 52, no. 4, pp. 903–914, Apr. 2017.

[6] J. Jo, S. Kim, and I.-C. Park, “Energy-efficient convolution architecture based on rescheduled dataflow,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 65, no. 12, pp. 4196–4207, Dec. 2018.

[7] C.-H. Chang, J. Gu, and M. Zhang, “Ultra low-voltage low-power CMOS 4-2 and 5-2 compressors for fast arithmetic circuits,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 51, no. 10, pp. 1985–1997, Oct. 2004.

[8] P. Aliparast, Z. D. Koozehkanani, and F. Nazari, “An ultra high speed digital 4-2 compressor in 65-nm CMOS,” Int. J. Comput. Theory Eng., vol. 5, no. 4, pp. 593–597, Aug. 2013.

[9] Y.-J. Chang, Y.-C. Cheng, Y.-F. Lin, S.-C. Liao, C.-H. Lai, and T.-C. Wu, “Imprecise 4–2 compressor design used in image processing applications,” IET Circuits, Devices Syst., vol. 13, no. 6, pp. 848–856, Sep. 2019.

[10] J. Park, S. Kim, and Y.-S. Lee, “A low-power booth multiplier using novel data partition method,” in Proc. IEEE Asia-Pacific Conf. Adv. Syst. Integ. Circuits, Aug. 2004, pp. 54–57.

[11] I. S. Abu-Khater, A. Bellaouar, and M. I. Elmasry, “Circuit techniques for CMOS low-power high-performance multipliers,” IEEE J. Solid-State Circuits, vol. 31, no. 10, pp. 1535–1546, Oct. 1996.

[12] M. D. Ercegovac and T. Lang, Digital Arithmetic, 1st ed. Morgan Kaufmann, 2003.

[13] P. J. Song and G. De Micheli, “Circuit and architecture trade-offs for CMOS low-power high-performance multipliers,” IEEE J. Solid-State Circuits, vol. 52, no. 9, pp. 1184–1198, Sep. 1991.

[14] G. Goto, T. Sato, M. Nakajima, and T. Sukemura, “A 54 × 54-b regularly structured tree multiplier,” IEEE J. Solid-State Circuits, vol. 27, no. 9, pp. 1229–1235, Sep. 1992.

[15] N. Ohkubo, M. Suzuki, T. Shinbo, T. Yamanaka, A. Shimizu, K. Sasalu, and Y. Nakagome, “A 4.4 ns CMOS 54 × 54-b multiplier using pass-transistor multiplexer,” IEEE J. Solid-State Circuits, vol. 30, no. 3, pp. 251–257, Mar. 1995.

[16] R. Fried, “Minimizing energy dissipation in high-speed multipliers,” in Proc. Int. Symp. Low Power Electron. Design, Aug. 1997, pp. 214–219.

[17] W.-C. Yeh and C.-W. Jen, “High-speed booth encoded parallel multiplier design,” IEEE Trans. Comput., vol. 49, no. 7, pp. 692–701, Jul. 2000.

[18] Z. Huang and M. D. Ercegovac, “High-performance low-power left-to-right array multiplier design,” IEEE Trans. Comput., vol. 54, no. 3, pp. 272–283, Mar. 2005.

[19] P. R. Rajput and M. N. S. Swamy, “High speed, efficient area, low power novel modified booth encoder multiplier for signed-unsigned number,” Adv. Intell. Syst. Comput., vol. 464, pp. 321–333, Apr. 2016.

[20] H. Jiang, J. Han, F. Qiao, and F. Lombardi, “Approximate radix-8 Booth multipliers for low-power and high-performance operation,” IEEE Trans. Comput., vol. 65, no. 8, pp. 2638–2644, Aug. 2016.

[21] W. Liu, L. Qian, C. Wang, H. Jiang, J. Han, and F. Lombardi, “Design of approximate radix-4 Booth multipliers for error-tolerant computing,” IEEE Trans. Comput., vol. 66, no. 8, pp. 1435–1444, Aug. 2017.

[22] S. Venkatachalam, E. Adams, H. J. Lee, and S.-B. Ko, “Design and analysis of area and power efficient approximate booth multipliers,” IEEE Trans. Comput., vol. 68, no. 11, pp. 1679–1703, Nov. 2019.

[23] A. D. Booth, “A signed binary multiplication technique,” Quart. J. Mech. Appl. Math., vol. 4, no. 2, pp. 236–240, 1951.

[24] O. Macorley, “High-speed arithmetic in binary computers,” Proc. IRE, vol. 49, no. 1, pp. 67–91, Jan. 1961.

[25] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, Digital Integrated Circuits, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.

[26] N. S. Kim, T. Austin, D. Blaauw, T. Mudge, K. Flautner, J. S. Hu, M. J. Irwin, M. Kandemir, and V. Narayanan, “Leakage current: Moore’s law meets static power,” Computer, vol. 36, no. 12, pp. 68–75, Dec. 2003.

[27] Y.-J. Chang, Y.-C. Cheng, Y.-F. Lin, S.-C. Liao, C.-H. Lai, and T.-C. Wu, “An approximate radix-4 booth multiplier for error-tolerant computing,” IEEE Transactions on Computers, vol. 66, no. 8, pp. 1435–1444, Aug. 2017.

[28] S. Nishizawa, T. Ishihara, and H. Onodera, “Analysis and comparison of XOR cell structures for low voltage circuit design,” in Proc. Int. Symp. Qual. Electron. Design (ISQED), Mar. 2013, pp. 703–708.