CHAPTER 3

DESIGN OF HIGH SPEED HARDWARE EFFICIENT CARRY SKIP ADDER AND CARRY SKIP BCD ADDER USING REVERSIBLE LOGIC

3.1 INTRODUCTION

Adders are the basic building blocks in many computational units. There are different types of adders and the most commonly used adders are Ripple Carry Adder (RCA), CSK adder and Carry Look-Ahead (CLA) adder. In RCA each adder stage incurs one carry propagation delay and this increases the total delay of addition (Morris Mano 2001). CLA adders on the other hand generate block carry in advance using a separate logic to reduce the delay due to carry propagation. However, the additional gates incur high hardware complexity in CLA adder (Islam et al 2009). As stated by Gu and Elmsary (1996) the leakage power dissipation increases with the number of gates. Thus a trade of between hardware complexity, power dissipation and critical delay exists between CLA adder and RCA. So, efficient adder circuits for binary and BCD additions have been investigated for several decades by algorithmic and circuit level changes. As the delay, static power dissipation and energy dissipation of reversible logic circuit tend to be significantly lower compared to its irreversible counterpart, the suitability of reversible logic for the design of circuits for multi bit addition is investigated. Thus
novel design for multi bit binary addition using reversible logic viz., CSK adder and its implementation for decimal addition viz., reversible CSK BCD adder are examined at gate level in this chapter.

### 3.2 PROPOSED REVERSIBLE CSK ADDER

The proposed CSK adder shown in Figure 3.1 consists of Modified TS Gate, PG and FRG shown in Figure 3.2, Figure 3.3 and Figure 3.4 respectively. MTSG is used for addition of input bits and a carry, to generate sum bits at appropriate positions. The input carry being the output of MTSG at previous bit position. To generate block propagate carry Z, PGs are used where \( Z = p_0 \cdot p_1 \cdot p_2 \cdot p_3 \). Signals \( p_0 = A_0 \ XOR \ B_0, \ p_1 = A_1 \ XOR \ B_1, \ p_2 = A_2 \ XOR \ B_2 \) and \( p_3 = A_3 \ XOR \ B_3 \) are generated by MTSGs at corresponding bit position. The output of MTSGs at first and second bit positions \( p_0 \) and \( p_1 \) are fed to a PG to produce the product \( p_0 \cdot p_1 \) and the output of MTSGs at third and fourth bit positions \( p_2 \) and \( p_3 \) are fed to another PG to produce the product \( p_2 \cdot p_3 \). The PG outputs \( p_0 \cdot p_1 \) and \( p_2 \cdot p_3 \) are fed to third PG to produce \( Z \). Though Toffoli gate does the same function, the PGs are preferred in the proposed design because of lower quantum cost (Biswas et al. 2008b).

The computation of output carry \( (C_{out}) \) involves block propagate signal \( Z \), input carry \( (C_{in}) \) and the most significant full adder carry output \( (C_4) \). i.e., \( C_{out} = Z \cdot C_{in} + C_4 \) and is realized using FRG. The carry skip logic can reduce carry propagation delay if the block propagate signal \( Z \) is equal to one.
Figure 3.1 Proposed carry skip adder

When \( Z = 1 \), the carry input \( C_{\text{in}} \) propagates to \( C_{\text{out}} \) regardless of the carry \( C_4 \), else the carry bit \( C_4 \) propagates to \( C_{\text{out}} \). In cases where \( Z = 0 \), the generation of \( C_{\text{out}} \) will incur 4 MTSG and a FRG delay. The algorithm for the design of binary adder based on the above methodology is as follows.

**Algorithm:**

Reversible Carry-skip-adder \((A,B,C_i)\)

**Input:**

Input Vectors: \( A = (A_3, A_2, A_1, A_0) \) and

\( B = (B_3, B_2, B_1, B_0) \)

Carry input: \( C_{\text{in}} \)
**Output:**

Output Vectors: $S = (C_{out}, S_3, S_2, S_1, S_0)$

**begin**

**Step 1:**

Compute propagate bit $P_i$ for each adder block.

for $(i = 0 \text{ to } 3)$

{
    $p_i = A_i \oplus B_i$
}

$Z = p_0 \text{ AND } p_1 \text{ AND } p_2 \text{ AND } p_3$

**Step 2:**

Compute $S_i = A_i \oplus B_i \oplus C_i$ for every adder block using the MTSG gate and $C_i = (A_i \oplus B_i)$. $C_{in} \oplus A_i \oplus B_i \oplus D$ is generated for each adder block where $D = 0$.

**Step 3:**

Evaluate the final carry output, $C_{out} = ZC_{in} + C_4$

**End**

**a) MTS Gate**

![MTSG Diagram](a)
In case of 4 bit addition with inputs \( A(A_3-A_0) \) and \( B(B_3-B_0) \), for \( i=0 \) to 3 where ‘i’ denotes the bit position, the inputs to the MTSG are \( A=A_i \), \( B=B_i \), \( C= \text{carry in} \ (C_{in}, C_1, C_2, C_3) \), \( D=0 \) and the outputs of MTSG are \( P= \text{garbage output} \), \( Q= \text{signal} \ p_i \), \( R= \text{sum} \ S_i \) and \( S= \text{carry output} \ (C_{i+1}) \).

b) Peres Gate

![Peres Gate Diagram]
Figure 3.3 PG (a) Logic diagram (b) Reversible implementation
(c) Quantum representation

For bit position $i$ (= 0-3) the corresponding signals $p_i$ generated by MTSG are $p_0$, $p_1$, $p_2$ and $p_3$. These form inputs to PG and thus the inputs and outputs of PG are defined as

$$A = p_{2k}, \quad B = p_{2k+1}, \quad C = 0, \quad P, \quad Q = \text{Garbage outputs}; \quad R_k = p_{2k} p_{2k+1}, \quad k = 0, 1 \text{ for first stage}$$

$$A = R_0, \quad B = R_1, \quad C = 0, \quad P, \quad Q = \text{Garbage outputs}; \quad R = p_3 p_2 p_1 p_0 \text{ for second stage.}$$
c) FRedkin Gate

FRG performs logical AND-OR operation. The inputs and outputs of FRG are defined as A= Z, B= C_{in}, C= C_4; P, Q = garbage outputs and R= C_{out}.

Figure 3.4 FRG (a) Logic diagram (b) Reversible implementation (c) Quantum representation
3.3 EXPERIMENTAL RESULTS

The VHDL netlist description of proposed reversible CSK adder is designed according to the methodology presented in previous section. Reversible adder designs by Bruce et al (2002), Islam et al (2009) and Lala et al (2010) are used for comparison. The schemes used for comparison are also designed using structural VHDL to produce gate level netlist and synthesized using Xilinx software. Experimental evaluation of the proposed CSK adder and designs used for comparison in terms of No. of reversible gates, No. of constant inputs, No. of garbage outputs and average delay are shown in Table 3.1. It is seen from Table 3.1 that the No. of gates in the proposed CSK adder decreases by 16, 6 and 14 gates with the percentage area reduction to be 66.7%, 42.9% and 63.6% when compared to Bruce et al(2002), Islam et al(2009) and Lala et al (2010) reversible designs respectively. This is due to area efficient MTSG which realizes first stage addition with fewer gates.

Table 3.1 Performance comparison of proposed and previous CSK adder designs

<table>
<thead>
<tr>
<th>Designs</th>
<th>Gate count</th>
<th>Garbage outputs</th>
<th>Constant Inputs</th>
<th>Average Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lala et al (2010)</td>
<td>22</td>
<td>27</td>
<td>22</td>
<td>10.5</td>
</tr>
<tr>
<td>Proposed</td>
<td>8</td>
<td>11</td>
<td>7</td>
<td>3.5</td>
</tr>
</tbody>
</table>
Note that the proposed reversible CSK adder exhibit better reduction in number of garbage outputs by 11, 8 and 16 with respective percentage reduction to be 50%, 42.1% and 59.2% and in terms of number of constant inputs by 14, 8 and 15 with the corresponding reduction in percentages to be 66.7%, 53.3% and 68.1% when compared with Bruce et al (2002), Islam et al (2009) and Lala et al (2010) reversible designs respectively.

This is due to the use of reversible Peres gate and FRG in the proposed adder design. A plot of constant inputs and garbage outputs of proposed CSK adder and previous approaches is shown in Figure 3.5 and Figure 3.6 respectively. From the delay values shown in Table 3.1 it is seen that the proposed design demonstrate 69.6%, 30% and 66.7% delay reduction compared to Bruce et al (2002), Islam et al (2009) and Lala et al (2010) reversible designs respectively.

![Figure 3.5 Constant inputs of proposed reversible CSK adder and previous approaches](image_url)
3.4 IMPLEMENTATION OF THE PROPOSED CSK ADDER FOR DECIMAL ADDITION

To verify the functionality and measure its driving capability the proposed CSK adder is implemented in a BCD adder and is shown in Figure 3.7. The CSK BCD adder uses the proposed reversible CSK adder discussed in section 3.2 for first stage addition and a BCD overflow detection and correction circuit to adjust the sum outputs of first stage to equivalent decimal. BCD overflow detection and correction is realized using MPS gate shown in Figure 2.4. The outputs $C_{out}$ and intermediate sum bits ($S_3$, $S_2$, $S_1$, $S_0$) of the first stage adder are fed to the MPS gate to check for the BCD overflow and to produce the equivalent decimal output ($C_{dout}$, $F_3$, $F_2$, $F_1$ and $F_0$). The inputs and outputs of MPS gate are defined as $A = C_{out}$, $B = S_3$; $C = S_2$; $D = S_1$; $E = S_0$; $P = C_{dout}$; $Q = F_3 = S_3$; $R = F_2 = S_2$; $S = F_1 = S_1$; $T = F_0 = S_0$. 

Figure 3.6 Garbage outputs of proposed reversible CSK adder and previous approaches
The implementation results of reversible CSK adder for BCD addition and state of the art reversible BCD designs are shown in Table 3.2. The BCD adder implementation results reported in Table 3.2 demonstrates that gate count of proposed CSK adder decimal system is significantly reduced compared to the reversible BCD designs used for comparison. For example the gate count of the CSK adder implemented BCD design decreases by 4 gates compared to the best of designs used for comparison. A plot of gate count of proposed CSK adder implemented BCD design and previous approaches are shown in Figure 3.8. Also note that the proposed CSK adder implemented BCD design highlights 30% reduction in constant inputs and 21.4% reduction in garbage outputs compared to the best of
designs used for comparison. In addition the delay parameters shown in Table 3.2 demonstrate that the proposed CSK implemented BCD design show a delay reduction of 50%, 47%, 50%, 68.9% and 43.8% when compared to Biswas et al (2008b), Thapliyal et al (2006), Islam et al (2009), Islam & Begum (2008) and Bhagyalakshmi & Venkatesha (2011) designs respectively.

Table 3.2 Performance comparison of proposed CSK adder implemented BCD design and state-of-the-art approaches

<table>
<thead>
<tr>
<th>Designs</th>
<th>Gate count</th>
<th>Garbage outputs</th>
<th>Constant Inputs</th>
<th>Average Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biswas et al (2008b)</td>
<td>15</td>
<td>14</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>Bhagyalakshmi &amp; Venkatesha (2011)</td>
<td>13</td>
<td>14</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>Proposed</td>
<td>9</td>
<td>11</td>
<td>7</td>
<td>4.5</td>
</tr>
</tbody>
</table>

A plot of average delay of proposed CSK adder implemented BCD design and previous approaches are shown in Figure 3.9. The metrics expressed in Table 3.2 presents clearly the potential benefits of proposed reversible adder design and their suitability for real time portable applications.
Figure 3.8 Gate counts of proposed CSK BCD adder and previous approaches

Figure 3.9 Average delays of proposed CSK BCD adder and previous approaches
3.5 CONCLUSION

In this chapter the design of an optimized carry skip adder and its suitability for decimal addition in reversible logic are examined. The proposed reversible adder design use cost efficient gates for ripple carry addition and carry skip logic input generation. This improves the performance of the proposed designs significantly and they are functionally verified at the logical level by using the structural VHDL code and HDL simulators. The evaluation results revealed that both circuits are highly optimized in terms of gate count, constant input and garbage output when compared with the existing designs. For example the gate count, garbage output and constant input of proposed CSK adder show atleast 6 gate, 8 outputs and 8 inputs reduction with the respective percentages to be 42.9%, 42.1% and 53.3% for 4 bit input, and its BCD counterpart show the corresponding reductions to be atleast 4 gates, 3 outputs and 3 constant inputs with respective percentages to be 30.7%, 21.4% and 30% for 1 digit addition. As a result, the proposed reversible binary and decimal adders are applicable in portable quantum computing to reduce cost and area.