A Cycle-Based Synthesis algorithm for Reversible Logic

Zahra Sasanian*, Mehdi Saeedi, Mehdi Sedighi, Morteza Saheb Zamani
{sasanian, msaedeei, msedighi, szamani}@aut.ac.ir
Quantum Design Automation Lab, Computer Engineering Department
Amirkabir University of Technology
Tehran, Iran

ASP-DAC 2009
Outline

- Introduction
- Basic Concepts
- Previous Work
- Synthesis Algorithm
- Experimental Results
- Future Works
- Conclusions
Introduction

- Reversible Logic
  - Equal number of inputs and outputs
  - Injective mapping
  - Example: \( f = \{0, 1, 3, 5, 2, 6, 7, 4\} \)
    
    \[
    \begin{align*}
    f: \text{input} & \rightarrow \text{Output} \\
    0 & \rightarrow 0 \\
    1 & \rightarrow 1 \\
    2 & \rightarrow 3 \\
    3 & \rightarrow 5 \\
    4 & \rightarrow 2 \\
    5 & \rightarrow 6 \\
    6 & \rightarrow 7 \\
    7 & \rightarrow 4
    \end{align*}
    \]

<table>
<thead>
<tr>
<th>( i_1 )</th>
<th>( i_2 )</th>
<th>( i_3 )</th>
<th>( f_1 )</th>
<th>( f_2 )</th>
<th>( f_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Power Dissipation

- **Rolf Landauer (1961)**
  - Every lost bit causes an energy loss
  - When a computer erases a bit of information, the amount of energy dissipated into the environment is at least $kT \times \ln 2$

- **Charles Bennett (1973)**
  - To avoid power dissipation in a circuit, the circuit must be built with reversible gates
Motivation

- Decrease in power dissipation
- Application in
  - Low power CMOS design
  - Optical computing
  - Nanotechnology
  - DNA computing
  - Quantum computing
    - Each unitary quantum gate is intrinsically reversible
Basic Concepts

Reversible gates

- NOT

\[ x \oplus \bar{x} \]

- CNOT

\[ x \oplus y \]

- C^2\text{NOT} (Toffoli)

\[ a \oplus b \oplus ab \]

Generalized Toffoli gate
Reversible Circuits

inputs       reversible gates       outputs

in1           out 1
in 2          out 2
in 3          out 3
in 4          out 4

2 in 1
2 out
3 in
2 out
4 in
2 out
Basic Concepts

Transposition  
\[ f = (a, b) \]

K-Cycle  
\[ f = (a_1, a_2, \ldots, a_k) \]
\[ f(a_1) = a_2, f(a_2) = a_3, \ldots, f(a_k) = a_1 \]

Disjoint Cycles

Cycles \( f \) and \( g \) are called disjoint if they have no common members, i.e.  
\[ \forall a \in f, a \notin g \text{ and vice versa} \]
Canonical Cycle Form (CCF)

- Every permutation function can be written uniquely, except for the order, as a product of disjoint cycles

\[ f = (a_1, a_2, \ldots, a_k)(b_1, b_2, \ldots, b_j)(c_1, c_2, \ldots, c_j) \]
Previous Work

Synthesis Algorithm of [6]
- Uses NCT (NOT, CNOT, Toffoli) library
- Decomposes every cycle with length larger than two in the CCF of the permutation function to a set of pairs of disjoint transpositions

\[(x_0, x_1, x_2, \ldots, x_k) = (x_0, x_1)(x_{k-1}, x_k)(x_0, x_2, x_3, \ldots, x_{k-1})\]
Previous Work

- Synthesis Algorithm of [6]
  - Synthesizes each disjoint transposition pair $(a, b)(c, d)$ using $\pi k_0 \pi^{-1}$ circuit
Cycle-Based Synthesis Algorithm

Goal:
- To show the effect of synthesizing larger cycles directly
- To avoid redundant term synthesis
  - Each term is synthesized once and is fixed in next steps
Cycle-Based Synthesis Algorithm

Direct Synthesis of 3-Cycles

\[ \pi_1, \pi_2, \pi_1^{-1} \]

Inputs: \( a, b, c \)

Initial Terms:
- \( a: 2^{n+1}+1 \)
- \( b: 2^{n}-1 \)
- \( c: 2^{k}-1 \)

Intermediate Terms:
- \( \pi_1: 2^{n}-2^{k-1}-1 \)
- \( \pi_2: 2^{n}-2^{k-1}-1 \)

Intermediate Terms:
- \( K_{0(3)}: 2^{n-1}-1 \)
- \( \pi_2^{-1}: 2^{n+1}+1 \)
- \( \pi_1^{-1}: 2^{n}-1 \)

Outputs: \( c, a, b \)
Cycle-Based Synthesis Algorithm

- 3-Cycle generator $k_{0(3)}$

\[ k_{0(3)} = (2^n - 2^{k-1} - 1, 2^n - 1, 2^{n-1} - 1) \]

\[ k = \lfloor n/2 \rfloor \]
Cycle-Based Synthesis Algorithm

- \( \pi_2 \) Circuit for every 3-cycle

\[ k = \lfloor n/2 \rfloor \]
Example

- $f = (73, 63, 13), \ n=7$
- Using Toffoli gates
## Building Blocks

<table>
<thead>
<tr>
<th>Primitive Cycles</th>
<th>Initial terms</th>
<th>$\pi_2$ Circuit</th>
<th>Intermediate terms</th>
<th>$K_0$ Circuit $(k=\lceil n/2 \rceil)$</th>
<th># of gates (([6]/ours))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(2-Cycle)(2-Cycle)</td>
<td>$(2^{n-1}+4, 2^{n-1}+1)(2^{n-1}+2, 2^{n-1}+7)$</td>
<td>T(0,$n-1,2$) T(1,$n-1,2$) T(2,$n-1,[3, \ldots, n-2]$)</td>
<td>$(2^{n-4}2^{n-3})(2^{n-2},2^{n-1})$</td>
<td>$C^{n-1}$\text{NOT}(n-1,n-2,\ldots,2,0)</td>
<td>18n-44/18n-44</td>
</tr>
<tr>
<td>(3-Cycle)</td>
<td>$(2^{n-1}-1,2^{n-1}-1,2^{n-1}-1)$</td>
<td>T(0,$n-1,k-1$) T(0,$k-1,[1\ldots k-2,k\ldots n-2]$) T(0,$n-1,k-1$)</td>
<td>$(2^{n-2k-1}2^{n-1}-1,2^{n-1}-1)$</td>
<td>$C^{n-k}$\text{NOT}(n-1,\ldots,k,k-1) $C^{k}$\text{NOT}(k-1,\ldots,0,n-1)</td>
<td>36n-88/16n-34</td>
</tr>
<tr>
<td>(3-Cycle)(3-Cycle)</td>
<td>$(2^{k-1}-1,2^{n-1}-1,2^{n-1}-1,2^{n-2}-2) (2^{k-1}-1,2^{n-1}-2,2^{n-2})$</td>
<td>T(0,$n-1,n-2$) T(1,$n-1,n-2$) T(0,$k-1,n-1$) T(0,$k-1,n-2$) T(1,$k-1,n-1$) T(1,$k-1,n-2$) T(0,$n-2,1$) T(1,$n-2,[2,\ldots,n-3]$)</td>
<td>$(2^{n-2k-1}2^{n-1}-1,2^{n-1}-1) (2^{n-2-k-1}2^{n-2},2^{n-1}-2)$</td>
<td>$C^{n-k}$\text{NOT}(n-1,\ldots,k,k-1) $C^{k-1}$\text{NOT}(k-1,\ldots,1,n-1) $C^{n-k}$\text{NOT}(n-1,\ldots,k,k-1) $C^{k-1}$\text{NOT}(k-1,\ldots,1,n-1)</td>
<td>36n-88/22n-34</td>
</tr>
</tbody>
</table>
Partitioning

\[ f = (a, b, c, d, e, f, g, h, i, j, k, l, m) \]
\[ = (a, b, c) (d, e, f) (g, h, i) (j, k, l) (m, a, d, g, j) \]
\[ = (a, b, c) (d, e, f) (g, h, i) (j, k, l) (m, a, d) (g, j, m) \]

Number of Gates (ours) = 2(22n-34)+2(16n-34) = 76n-136
Number of Gates ([6]) = 12(18n-44) = 108n-264
## Experimental Results

<table>
<thead>
<tr>
<th>Circuits</th>
<th># of Inputs</th>
<th>Elapsed Time (ms)</th>
<th>Number of gates</th>
<th>Imp. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>18450</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>4</td>
<td>6</td>
<td>7512</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>9</td>
<td>9</td>
<td>42304</td>
</tr>
<tr>
<td>4</td>
<td>11</td>
<td>24</td>
<td>23</td>
<td>55156</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>10</td>
<td>8</td>
<td>12944</td>
</tr>
<tr>
<td>6</td>
<td>15</td>
<td>16</td>
<td>15</td>
<td>18784</td>
</tr>
<tr>
<td>7</td>
<td>16</td>
<td>32</td>
<td>30</td>
<td>33798</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
<td>19</td>
<td>20</td>
<td>48848</td>
</tr>
<tr>
<td>9</td>
<td>19</td>
<td>21</td>
<td>15</td>
<td>44876</td>
</tr>
<tr>
<td>10</td>
<td>23</td>
<td>15</td>
<td>7</td>
<td>32172</td>
</tr>
</tbody>
</table>
Experimental Results (Cont.)

<table>
<thead>
<tr>
<th>Circuits</th>
<th># of Inputs</th>
<th>Elapsed Time (ms)</th>
<th>Number of gates</th>
<th>Imp. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>22</td>
<td>9</td>
<td>5</td>
<td>20036</td>
</tr>
<tr>
<td>12</td>
<td>24</td>
<td>10</td>
<td>9</td>
<td>21800</td>
</tr>
<tr>
<td>13</td>
<td>25</td>
<td>12</td>
<td>11</td>
<td>23646</td>
</tr>
<tr>
<td>14</td>
<td>23</td>
<td>7</td>
<td>6</td>
<td>15468</td>
</tr>
<tr>
<td>15</td>
<td>25</td>
<td>16</td>
<td>15</td>
<td>31184</td>
</tr>
<tr>
<td>16</td>
<td>28</td>
<td>17</td>
<td>19</td>
<td>30568</td>
</tr>
<tr>
<td>17</td>
<td>30</td>
<td>31</td>
<td>19</td>
<td>30308</td>
</tr>
<tr>
<td>18</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>24728</td>
</tr>
<tr>
<td>19</td>
<td>10</td>
<td>15</td>
<td>10</td>
<td>41452</td>
</tr>
<tr>
<td>20</td>
<td>10</td>
<td>18</td>
<td>21</td>
<td>46224</td>
</tr>
<tr>
<td>Average</td>
<td>17.15</td>
<td>15.2</td>
<td>13.35</td>
<td>26600</td>
</tr>
</tbody>
</table>
Future Directions

- Generalization of the Cycle-Based Algorithm
  - ex. $K_0$ circuit for the $2^m$-cycles
  - Is there an optimum cycle length?
Conclusions

- A new synthesis algorithm was proposed using direct synthesis of cycles
- The proposed algorithm uses simple NCT gates with no extra garbage bits
- The run time of the proposed synthesis algorithm is negligible
- The results show 34% improvement in number of generated gates over the existing algorithm of [6]
Thanks