EFFICIENT REALIZATION OF XOR-INTENSIVE FUNCTIONS IN FPGAs

1. INTRODUCTION

- Emerging XOR Intensive Applications
  - error detecting/correcting
  - data encryption/decryption
  - arithmetic circuits

✓ by AND/OR-based EDA tools?
Efficient Realization of XOR-intensive Functions in FPGAs

**AND/OR EXPRESSION & REALIZATION**

<table>
<thead>
<tr>
<th>xy</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>zw</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ f = (xzw) + (xzw) + (yzw) + (xzw) + (yzw) \]

**AND/XOR EXPRESSION & REALIZATION**

<table>
<thead>
<tr>
<th>xy</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>zw</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ f = \overline{y} \overline{w} \oplus z \oplus \overline{x} \overline{y} \overline{z} \oplus \overline{x} y w \]
2. TECHNOLOGY MAPPING FOR AND/XOR EXPRESSIONS

- Shannon’s & Davio’s Expansion
- Example
- Proposed Technology Mapping Method

SHANNON’S EXPANSION

- An arbitrary logic function $f(x_1, x_2, ..., x_n)$ can be expanded as

$$ f = \bar{x}_i f_i^0 \oplus x_i f_i^1 \quad (1) $$

where $f_i^0 = f(x_1, x_2, ..., x_i = 0, ..., x_n)$, and

$$ f_i^1 = f(x_1, x_2, ..., x_i = 1, ..., x_n) $$
**Positive Davio’s Expansion**

An arbitrary logic function \( f(x_1, x_2, \ldots, x_n) \) can be expanded as

\[
f = f_i^0 \oplus x_if_i^2 \tag{2}
\]

where \( f_i^0 = f(x_1, x_2, \ldots, x_i = 0, \ldots, x_n) \),
\( f_i^1 = f(x_1, x_2, \ldots, x_i = 1, \ldots, x_n) \), and
\( f_i^2 = f_i^0 \oplus f_i^1 \)

---

**Negative Davio’s Expansion**

An arbitrary logic function \( f(x_1, x_2, \ldots, x_n) \) can be expanded as

\[
f = f_i^1 \oplus \bar{x}_if_i^2 \tag{3}
\]

where \( f_i^0 = f(x_1, x_2, \ldots, x_i = 0, \ldots, x_n) \),
\( f_i^1 = f(x_1, x_2, \ldots, x_i = 1, \ldots, x_n) \), and
\( f_i^2 = f_i^0 \oplus f_i^1 \)
**Example of Shannon / Davio Expansion**

\[ f = x_1 x_2 x_3 x_4 x_5 \oplus x_1 x_3 x_4 x_5 \oplus x_2 x_3 x_4 x_5 \oplus x_1 x_3 x_4 x_5 \]

\[ f_0 = x_3 x_4 x_5 \oplus x_2 x_3 x_4 x_5 \]

\[ f_1 = x_2 x_3 x_4 x_5 \oplus x_2 x_3 x_4 x_5 \oplus x_3 x_4 x_5 \]

\[ f_2 = x_3 x_4 x_5 \oplus x_2 x_3 x_4 x_5 \oplus x_3 x_4 x_5 \]

\[ f_s = \overline{x_1} (x_3 x_4 x_5 \oplus x_2 x_3 x_4 x_5) \oplus \]

\[ x_1 (x_2 x_3 x_4 x_5 \oplus x_2 x_3 x_4 x_5 \oplus x_3 x_4 x_5) \]

\[ f_{+d} = (x_3 x_4 x_5 \oplus x_2 x_3 x_4 x_5) \oplus \\
\]

\[ x_1 (x_3 x_4 x_5 \oplus x_2 x_3 x_4 x_5 \oplus x_3 x_4 x_5) \]

\[ f_d = (x_2 x_3 x_4 x_5 \oplus x_2 x_3 x_4 x_5 \oplus x_3 x_4 x_5) \oplus \\
\]

\[ \overline{x_1} (x_3 x_4 x_5 \oplus x_2 x_3 x_4 x_5 \oplus x_3 x_4 x_5) \]

---

**Proposed Technology Mapping Method**

1. if (# of input variables of f) \( \leq 5 \)
   - exit; f can be fit into a CLB perfectly
   - else go to step 2
2. Pick the LFU input variable in f and let it be \( x_i \)
3. Decompose f into \( f = f_0 \oplus x_i f_i^2 \)
4. if (# of input variables in \( f_i^2 \)) > 5
   - replace f with new \( f_i^2 \) and go to step 2
   - else go to step 5
5. if (# of input variables in \( f_i^2 \)) > 5
   - replace f with new \( f_i^2 \) and go to step 2
   - else go to step 6
6. Update f with the new \( f_i^2 \) and \( f_i \)
7. Exit
3. Example
Preprocessing for inc_p

1. inc.pla; original format
2. inc.vhd; VHDL format
3. inc_p_t.vhd; parity prediction circuit
4. minimizer(p); Logic Optimization
5. inc_p.vhd; optimized parity circuit
6. Technology Mapping (next slides)

Example (inc_p) of Technology Mapping

Step 1: (# of input variables in inc_p) = 7
f = \( \text{inc}_p \oplus \text{inc}_p \oplus \text{inc}_p \oplus \text{inc}_p \oplus \text{inc}_p \oplus \text{inc}_p \oplus \text{inc}_p \)

Step 2: \( i = 4 \)

Step 3:
\[ f' = \text{inc}_p \oplus \text{inc}_p \oplus \text{inc}_p \oplus \text{inc}_p \oplus \text{inc}_p \oplus \text{inc}_p \oplus \text{inc}_p \]
\( f = f' \oplus \text{inc}_p \oplus \text{inc}_p \).

\[ f = f' \oplus \text{inc}_p \oplus \text{inc}_p \]

Efficient Realization of XOR-intensive Functions in FPGAs 12 of 29
**Step 4:** Go to Step 2 since # of input variables in $f_i$ is six; let $f = f_i$

**Step 2:** $i = 6$

**Step 3:**

\[ f_i = -01-235 \oplus 2-35 \oplus 0-1-2-3 \oplus 012-35 \oplus 0-1-3 \oplus 0135 \oplus -0235 \oplus -02 \oplus 01-2 \oplus -1-2-3 \]

\[ f_i = -01-235 \oplus 2-35 \oplus 012-35 \oplus 0-1-3 \oplus 012-35 \oplus -02 \oplus 01-2 \oplus -1-2-3 \]

\[ f_i = 0-1-2-3 \oplus -0135 \oplus -0235 \oplus -01-2 \oplus -012-3-5 \]

\[ f = f_i \oplus f_i \]

\[ f = (-01-235 \oplus 2-35 \oplus 0-1-2-3 \oplus 012-35 \oplus 0-1-3 \oplus 0135 \oplus -0235 \oplus -02 \oplus 01-2 \oplus -1-2-3) \]

\[ \oplus 6 (0-1-2-3 \oplus -0135 \oplus -0235 \oplus -01-2 \oplus -012-3-5) \]

---

**Step 4:** Go to Step 5 since # of input variables in $f_i$ is five

**Step 5:** Go to Step 2 since # of input variables in $f_i$ is six; let $f = f_i$

**Step 2:** $i = 5$

**Step 3:**

\[ f_i = 0-1-2-3-6 \oplus -012-36 \oplus 0-1-2-3 \]

\[ f_i = 0-1-2-3-6 \oplus -012-36 \oplus 0-1-2-3 \]

\[ f_i = -012-36 \]

\[ f = f_i \oplus 5 f_i \]

\[ f = (0-1-2-3-6 \oplus -012-36 \oplus 0-1-2-3) \oplus 5(-012-36) \]

---
**Step 4:** Go to Step 5 since # of input variables in \( f_i = 5 \)

**Step 5:** Go to Step 6 since # of input variables in \( f_i = 5 \)

**Step 6:**

\[
f = (\oplus 01\oplus 235 \oplus 235 \oplus 01235 \oplus 01235 \oplus 0135 \oplus 0235 \\
\oplus 0123 \oplus 0123 \oplus 0123 \oplus 0123 \oplus 0123 \oplus 0123 \\
\oplus 0123 \oplus 0123 \oplus 0123 \oplus 0123 \oplus 0123 \oplus 0123)
\]

**Step 7:** Exit
4. RESULTS

- MCNC Benchmark Circuits
- Xilinx’s XC4010 (400 CLBs)
- Xilinx’s Foundation Software
- Microsoft Visual C++ 6.0
Efficient Realization of XOR-intensive Functions in FPGAs

**Design Flow**

1. MCNC Benchmark
2. Convert to VHDL
3. Parity Prediction Circuit (VHDL)
4. Convert to AND/XOR (VHDL)
5. Xilinx Foundation FPGA
6. Direct Approach
7. AND/XOR Direct Approach
8. Shannon Approach
9. Proposed Davio Approach

**Number of CLBs (Parity Prediction Circuits)**

<table>
<thead>
<tr>
<th>Stats. Ex.</th>
<th>In</th>
<th>Out</th>
<th>Prod. Terms</th>
<th>Original</th>
<th>Parity Prediction</th>
<th>Circuits</th>
</tr>
</thead>
<tbody>
<tr>
<td>misex1</td>
<td>8</td>
<td>7</td>
<td>32</td>
<td>A^+ B^+</td>
<td>A^+ B^+</td>
<td>A^+ B^+</td>
</tr>
<tr>
<td>sqrt8</td>
<td>8</td>
<td>4</td>
<td>40</td>
<td>11 10</td>
<td>7 5</td>
<td>3 5</td>
</tr>
<tr>
<td>5xp1</td>
<td>7</td>
<td>10</td>
<td>75</td>
<td>18 15</td>
<td>33 28</td>
<td>17 17</td>
</tr>
<tr>
<td>inc</td>
<td>7</td>
<td>9</td>
<td>34</td>
<td>22 20</td>
<td>15 8</td>
<td>10 8</td>
</tr>
<tr>
<td>con1</td>
<td>7</td>
<td>2</td>
<td>9</td>
<td>3 2</td>
<td>2 3</td>
<td>2 3</td>
</tr>
<tr>
<td>bw</td>
<td>5</td>
<td>28</td>
<td>87</td>
<td>32 31</td>
<td>1 1</td>
<td>1 1</td>
</tr>
<tr>
<td>rd35</td>
<td>5</td>
<td>3</td>
<td>32</td>
<td>3 3</td>
<td>1 1</td>
<td>1 1</td>
</tr>
<tr>
<td>squar5</td>
<td>5</td>
<td>8</td>
<td>32</td>
<td>6 6</td>
<td>1 1</td>
<td>1 1</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>52</td>
<td>71</td>
<td>341</td>
<td>108 97</td>
<td>68 52</td>
<td>41 43</td>
</tr>
<tr>
<td><strong>Avg.</strong></td>
<td>6.5</td>
<td>8.9</td>
<td>42.6</td>
<td>13.5 12.1</td>
<td>8.5 6.5</td>
<td>5.1 5.4</td>
</tr>
</tbody>
</table>

A: # of CLBs when optimized for speed
B: # of CLBs when optimized for area in Xilinx Foundation
### Total Equivalent Gate Counts
(Parity Prediction Circuits)

<table>
<thead>
<tr>
<th>Stats. Ex.</th>
<th># In</th>
<th># Out</th>
<th># Prod. Terms</th>
<th>Direct Approach</th>
<th>AND/XOR Direct</th>
<th>Proposed Davio Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>misex1</td>
<td>8</td>
<td>7</td>
<td>32</td>
<td>75</td>
<td>34</td>
<td>33</td>
</tr>
<tr>
<td>sqrt8</td>
<td>8</td>
<td>4</td>
<td>40</td>
<td>82</td>
<td>63</td>
<td>43</td>
</tr>
<tr>
<td>5xp1</td>
<td>7</td>
<td>10</td>
<td>75</td>
<td>453</td>
<td>208</td>
<td>72</td>
</tr>
<tr>
<td>inc</td>
<td>7</td>
<td>9</td>
<td>34</td>
<td>178</td>
<td>129</td>
<td>66</td>
</tr>
<tr>
<td>con1</td>
<td>7</td>
<td>2</td>
<td>9</td>
<td>30</td>
<td>36</td>
<td>37</td>
</tr>
<tr>
<td>bw</td>
<td>5</td>
<td>28</td>
<td>87</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>rd53</td>
<td>5</td>
<td>3</td>
<td>32</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>squar5</td>
<td>5</td>
<td>8</td>
<td>32</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>Total</td>
<td>52</td>
<td>71</td>
<td>341</td>
<td>866</td>
<td>518</td>
<td>299</td>
</tr>
<tr>
<td>Average</td>
<td>6.5</td>
<td>8.9</td>
<td>42.6</td>
<td>108.25</td>
<td>64.75</td>
<td>37.38</td>
</tr>
</tbody>
</table>

### Max Combinational Path Delay & Max Net Delay (Parity Prediction Circuits)

<table>
<thead>
<tr>
<th>Stats. Ex.</th>
<th>Direct Approach</th>
<th>AND/XOR Direct</th>
<th>Proposed Davio Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(A^*)</td>
<td>(B^*)</td>
<td>(A^*)</td>
</tr>
<tr>
<td>misex1</td>
<td>28.576</td>
<td>22.594</td>
<td>18.824</td>
</tr>
<tr>
<td>sqrt8</td>
<td>25.173</td>
<td>21.559</td>
<td>30.782</td>
</tr>
<tr>
<td>5xp1</td>
<td>49.317</td>
<td>27.354</td>
<td>35.672</td>
</tr>
<tr>
<td>inc</td>
<td>35.452</td>
<td>28.762</td>
<td>25.868</td>
</tr>
<tr>
<td>con1</td>
<td>21.634</td>
<td>25.334</td>
<td>23.494</td>
</tr>
<tr>
<td>bw</td>
<td>14.123</td>
<td>15.642</td>
<td>15.656</td>
</tr>
<tr>
<td>rd53</td>
<td>14.455</td>
<td>15.713</td>
<td>15.713</td>
</tr>
<tr>
<td>Total</td>
<td>205.566</td>
<td>174.068</td>
<td>182.845</td>
</tr>
</tbody>
</table>
## NUMBER OF CLBs
(MCNC BENCHMARK CIRCUITS)

<table>
<thead>
<tr>
<th>Ex.</th>
<th># In</th>
<th># Out</th>
<th># Prod. Terms</th>
<th>Direct Approach A</th>
<th>Direct Approach B</th>
<th>AND/XOR Direct A</th>
<th>AND/XOR Direct B</th>
<th>Proposed Davio Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>misex1</td>
<td>8</td>
<td>7</td>
<td>32</td>
<td>11</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>sqrt8</td>
<td>8</td>
<td>4</td>
<td>40</td>
<td>13</td>
<td>10</td>
<td>13</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>5xp1</td>
<td>7</td>
<td>10</td>
<td>75</td>
<td>18</td>
<td>15</td>
<td>38</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>inc</td>
<td>7</td>
<td>9</td>
<td>34</td>
<td>22</td>
<td>20</td>
<td>29</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>con1</td>
<td>7</td>
<td>2</td>
<td>9</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>bw</td>
<td>5</td>
<td>28</td>
<td>87</td>
<td>32</td>
<td>31</td>
<td>37</td>
<td>28</td>
<td>28</td>
</tr>
<tr>
<td>rd53</td>
<td>5</td>
<td>3</td>
<td>32</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>squar5</td>
<td>5</td>
<td>8</td>
<td>32</td>
<td>6</td>
<td>6</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Total</td>
<td>52</td>
<td>71</td>
<td>341</td>
<td>108</td>
<td>97</td>
<td>140</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Average</td>
<td>6.5</td>
<td>8.9</td>
<td>42.6</td>
<td>13.5</td>
<td>12.13</td>
<td>17.5</td>
<td>12.5</td>
<td></td>
</tr>
</tbody>
</table>

A: # of CLBs when optimized for speed in Xilinx Foundation 2.1i
B: # of CLBs when optimized for area in Xilinx Foundation 2.1i

### 5. CONCLUSIONS

- **Investigated AND/OR and AND/XOR technology mapping methods in FPGA environment**
- **Proposed a Davio’s Expansion-based technology mapping method**
- **Conducted experiments using MCNC benchmark circuits considering Direct Approach, AND/XOR Direct, and Proposed Davio Approach**
Superior for XOR intensive functions
- # of CLBs: reduced by 67.6%\(^a\) and 57.7%\(^b\)
- Gate Counts: reduced by 65.5%
- Max. Comb. Path Delay: reduced by 56.7%
- Max. Net Delay: reduced by 80.5%

Competitive for non XOR intensive functions
- # of CLBs: 12.5 vs 13.5\(^a\)/12.1\(^b\)

\(^a\): speed optimized Direct Approach
\(^b\): area optimized Direct Approach

CURRENT WORK

Area and Delay Results of MCNC Benchmarks’ Parity Prediction Circuits
CURRENT WORK cont.

Area-Delay Product Results of MCNC Benchmarks’ Parity Prediction Circuits

CURRENT WORK cont.

Area and Delay Results of MCNC Benchmark Circuits
FUTURE WORK

- Error correcting / detecting circuits
- Data encryption / decryption
- Computer arithmetic circuits
- Floating Point Unit and its applications