Algebraic and Boolean Optimization Methods for AQFP Superconducting Circuits

Eleonora Testa, Siang-Yun Lee, Heinz Riener and Giovanni De Micheli

Contact: testa.eleonora91@gmail.com

January 2021

26th Asia and South Pacific Design Automation Conference
Introduction

SuperConducting Electronics (SCE)

- Scalable computing at low-power
- Replacement to CMOS
- Rapid-Single-Flux Quantum (RSFQ) circuits and Adiabatic-Quantum Flux Parametron (AQFP)

AQFP $\rightarrow$ Low energy consumption (AC biased)

Electronic Design Automation (EDA) for SCE

- Lack of complete flow
- Different constraints (e.g., inputs present in specific timeframes) from standard CMOS

Our work: new logic synthesis flow for AQFP circuits

AQFP Technology and Library

• Basic block of the library: buffer
• 3 buffers in parallel → majority gate
• Biased buffers → AND and OR
• Inversion: easy to realize and free of cost

• Each gate inputs have to be present in specific timing windows (path balancing)
• Splitters insertion to deal with fanout > 1
**Logic Synthesis for AQFP**

**AQFP Technology**

- Majority-based + inversions
- AND and OR same cost as MAJ gate
- Splitters for each fanout > 1, splitters are buffers
- Path balancing to have inputs at specific timing frames

**New Logic Synthesis Flow**

- Use Majority-Inverter Graphs (MIGs, [3]) to represent and optimize functions
- Logic optimizations that maximize the use of majority gates
- Limit fanout increase during optimization (each splitter has delay 1)
- Optimize depth as first target metric (use fewer buffers in path balancing)

---


(c) Eleonora Testa
Novel Logic Synthesis Flow for AQFP

1. Description of the circuit (verilog, aiger)
2. Create the initial MIG
3. Depth optimization
4. Area optimization with resubstitution and refactoring
5. Splitters and buffers insertion
6. Map into AQFP technology

Remarks:
- Depth optimization uses algebraic rules of MIGs
- Boolean methods to optimize area
- Fanout increase accounted for during all steps
- Final step allows correct functionality when mapped into AQFP
Step1: Create the Initial MIG

Create an MIG, that (i) maximizes the use of majority (over AND and OR) and (ii) has fanout limited to 16

- **Rewriting [4]:** replace subgraphs with new graphs (smaller) from a database
- **Fanout limitation [5]:** duplication of nodes with fanout > 16

---

Step 2: Depth Optimization

Use algebraic rules of MIGs to reduce the depth of the critical path

- Algebraic rules used:
  
  **Associativity** $\langle xu(yuz) \rangle = \langle zu(yux) \rangle$
  
  **Distributivity** $\langle xy(uvz) \rangle = \langle \langle xyu(xyz) \rangle v \rangle$
  
  **Complementary Assoc.** $\langle xu(y\bar{u}z) \rangle = \langle xu(yxz) \rangle$

- Nodes have different depth according to their fanout (e.g., fanout = 2, depth = 2)

- Algebraic rules may not increment the fanout size of some nodes (e.g., increase from 4 to 5 results in more splitters, etc.)
Step2: Depth Optimization

Use algebraic rules of MIGs to reduce the depth of the critical path

• Depth consider the splitters

• Works over the critical path

• Apply algebraic rules that consider fanout increase

• Distributivity tried as last resource -> area increase

---

**Input:** MIG $M$, allow_area_increase

**Output:** Depth Optimized MIG $M$

\[
M \leftarrow \text{countSplittersDelay}(M);
\]

\[
\text{maxDepth} \leftarrow \text{depth}(M);
\]

**foreach** output po in $M$ **do**

**if** level(po) < max_depth **then** continue;

**foreach** node n in topological order in TFI po **do**

\[
\text{children}_2 \leftarrow \text{child of } n \text{ with largest level};
\]

**if** fanout_size(children_2) > 1 **then** continue;

**if** try_associativity(n) **then** continue;

**if** try_complAssociativity(n) **then** continue;

**if** allow_area_increase **then**

**if** try_distr(n) **then** continue;

**network-cleanup-and-sweeping**(M);
Step 3: Area Opt. with Resubstitution and Refactoring

Use Boolean resubstitution and refactoring to reduce the size of the MIG

- **Resubstitution**: express node \( n \) using other nodes (divisors) in the network
  - Check that fanout of nodes is not increased
  - Consider divisors of smaller depth to not increase the depth of the MIG
- **Relevance resub**:
  - \( \text{Relevance} - \langle xyz \rangle = \langle x_y \bar{z} y z \rangle \)
- **Refactoring**: resynthesizes subnetworks using [6]

```plaintext
Input: MIG \( M \), cut_size, max_divs
Output: Size Optimized MIG \( M \)
1 \( M \leftarrow \text{count_splits}_\text{delay}(M); \)
2 \( \text{list} \leftarrow \text{topological_sort_network}(M); \)
3 \textbf{foreach} node \( n \) in \text{list} \textbf{do}
4 \hspace{1em} \text{max_depth} \leftarrow \text{level}(n) - 1;
5 \hspace{1em} \text{cut} \leftarrow \text{find_reconvergent_cut}(n, \text{cut_size});
6 \hspace{1em} \text{m}f\text{c} \leftarrow \text{computeMFFC}(n);
7 \hspace{1em} \textbf{if} |\text{m}f\text{c}| > 0 \textbf{then}
8 \hspace{2em} \text{div} \leftarrow \text{divisors(}\text{list, } n, \text{max_div, max_depth});
9 \hspace{2em} \text{truth_tables(}\text{cut});
10 \hspace{2em} \textbf{if} \text{try}_\text{const}-\text{resub(}\text{list, } n, \text{div}) \textbf{then} \text{continue};
11 \hspace{2em} \textbf{if} \text{try}_0-\text{resub(}\text{list, } n, \text{div}) \textbf{then} \text{continue};
12 \hspace{2em} \textbf{if} \text{try}_\text{relevance}-\text{resub(}\text{list, } n, \text{div}) \textbf{then} \text{continue};
13 \hspace{2em} \textbf{if} \text{try}_1-\text{resub(}\text{list, } n, \text{div, m}f\text{c}) \textbf{then} \text{continue};
14 \hspace{1em} \text{network-cleanup-and-sweeping}(M);
```

Step 4: Splitters and Buffers Insertion

Splitters are used for fanout > 1, and buffers are needed to balance the path

- First, inserts splitters and splitter trees for all fanout > 1
- Max. fanout is 16, thus max. 5 splitters in 2 levels
- Followed by buffer insertion for all nodes and POs
- Buffers: all input arrival times balanced for each node
- Buffers can be shared
Experimental Results

- All implemented using EPFL logic synthesis libraries [7]
- Equivalence checking done with ABC
- Two sets of experiments:
  1. MIG improvement
  2. AQFP mapping results

AQFP library for mapping:

<table>
<thead>
<tr>
<th></th>
<th>Area (# of JJs)</th>
<th>Delay (Levels of JJs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-input MAJ</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>2-input AND</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>2-input OR</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>1:4 Splitter</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Buffer</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Technology-Independent Results

Original MIG: obtained with Step1

Optimized MIG: after depth optimization, resub and refactoring

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Original MIG</th>
<th>Optimized MIG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Size  Depth</td>
<td>Max. F.</td>
</tr>
<tr>
<td>5xp1</td>
<td>116    10</td>
<td>5</td>
</tr>
<tr>
<td>c1908</td>
<td>381    38</td>
<td>9</td>
</tr>
<tr>
<td>c432</td>
<td>178    44</td>
<td>16</td>
</tr>
<tr>
<td>c5315</td>
<td>1270   33</td>
<td>16</td>
</tr>
<tr>
<td>c880</td>
<td>300    28</td>
<td>10</td>
</tr>
<tr>
<td>chkn</td>
<td>421    28</td>
<td>9</td>
</tr>
<tr>
<td>count</td>
<td>119    18</td>
<td>5</td>
</tr>
<tr>
<td>dist</td>
<td>535    16</td>
<td>7</td>
</tr>
<tr>
<td>in5</td>
<td>443    19</td>
<td>14</td>
</tr>
<tr>
<td>in6</td>
<td>370    17</td>
<td>6</td>
</tr>
<tr>
<td>k2</td>
<td>1957   25</td>
<td>16</td>
</tr>
<tr>
<td>m3</td>
<td>411    13</td>
<td>7</td>
</tr>
<tr>
<td>max512</td>
<td>713    17</td>
<td>9</td>
</tr>
<tr>
<td>misex3</td>
<td>1533   24</td>
<td>16</td>
</tr>
<tr>
<td>mlp4</td>
<td>462    16</td>
<td>5</td>
</tr>
<tr>
<td>prom2</td>
<td>3484   22</td>
<td>16</td>
</tr>
<tr>
<td>sqr6</td>
<td>138    13</td>
<td>3</td>
</tr>
<tr>
<td>x1d1n</td>
<td>152    14</td>
<td>6</td>
</tr>
<tr>
<td>Averages</td>
<td></td>
<td>2.2</td>
</tr>
</tbody>
</table>

- **Goal:** depth optimization (21.4% on average)
- **Max. F:** maximum fanout. Such to not increase the splitters, and never > 16
- **Size is also optimized (up to 8.7%)**
### AQFP-Mapped Results

**Original MIG:** obtained with Step1 + buffers and splitters

**Optimized MIG:** after depth optimization, resub and refactoring + buffers and splitters

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Original MIG</th>
<th>Optimal MIG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>B. &amp; S. Area (# JJs)</td>
<td>Delay (JJs Levels)</td>
</tr>
<tr>
<td>5xp1</td>
<td>139 974</td>
<td>13 76</td>
</tr>
<tr>
<td>c1908</td>
<td>1683 5652</td>
<td>59 1481</td>
</tr>
<tr>
<td>c432</td>
<td>780 2628</td>
<td>64 732</td>
</tr>
<tr>
<td>c5315</td>
<td>6109 19838</td>
<td>53 5674</td>
</tr>
<tr>
<td>c880</td>
<td>1518 4836</td>
<td>42 1354</td>
</tr>
<tr>
<td>chkn</td>
<td>831 4188</td>
<td>33 751</td>
</tr>
<tr>
<td>count</td>
<td>464 1642</td>
<td>27 356</td>
</tr>
<tr>
<td>dist</td>
<td>680 4570</td>
<td>23 562</td>
</tr>
<tr>
<td>in5</td>
<td>907 4472</td>
<td>26 821</td>
</tr>
<tr>
<td>in6</td>
<td>694 3608</td>
<td>20 649</td>
</tr>
<tr>
<td>k2</td>
<td>3646 19034</td>
<td>36 3347</td>
</tr>
<tr>
<td>m3</td>
<td>516 3498</td>
<td>18 387</td>
</tr>
<tr>
<td>max512</td>
<td>879 6036</td>
<td>23 752</td>
</tr>
<tr>
<td>misex3</td>
<td>3108 15144</td>
<td>33 2946</td>
</tr>
<tr>
<td>mlp4</td>
<td>559 3890</td>
<td>22 506</td>
</tr>
<tr>
<td>prom2</td>
<td>4881 30666</td>
<td>27 4008</td>
</tr>
<tr>
<td>sqr6</td>
<td>241 1310</td>
<td>16 129</td>
</tr>
<tr>
<td>x1dn</td>
<td>206 1324</td>
<td>19 189</td>
</tr>
</tbody>
</table>

| Averages  | 15.7 | 7.4  | 20.9 |

(c) Eleonora Testa
Summary

- Novel logic synthesis flow for SCE and AQFP
- Use of Majority-Inverter Graphs (MIGs)
- Fanout limitations and constraints to account for splitters
- Depth optimization as first target metric (to reduce number of buffers need to path balancing)
- Algebraic and Boolean methods
- Results on AQFP, using number of JJs as main cost
Algebraic and Boolean Optimization Methods for AQFP Superconducting Circuits

Eleonora Testa, Siang-Yun Lee, Heinz Riener and Giovanni De Micheli

January 2021

26th Asia and South Pacific Design Automation Conference