Testability Improvement during High-Level Synthesis

Saeed Safari
CE Department, Sharif
University of Technology, Tehran, IRAN
safari@mehr.sharif.edu

Hadi Esmaeilzadeh
ECE Department,
University of Tehran,
Tehran, IRAN
hadi@cad.ece.ut.ac.ir

Amir-Hossein Jahangir
CE Department, Sharif
University of Technology,
Tehran, IRAN
jahangir@sharif.edu

Abstract

Improving testability during the early stages of High-Level Synthesis (HLS) reduces test hardware overheads, test costs, design iterations, and also improves fault coverage [1]. In this paper, we present a novel register allocation algorithm which is based on weighted graph coloring, targeting testability improvement.

1. Proposed testability model

In our register allocation method, we construct an extended conflict graph (ECG) whose vertices are variables and the weight of each edge shows the cost of assigning two endpoint variables of the edge to a register. There are three types of edges in the ECG, an edge with $+\infty$ weight which shows that the corresponding variables cannot be assigned to a single register, positive weight shows that we prefer not to assign these variables to a register, and negative weight shows the preference to assign these variables to a register.

**CY/OY Enhancement:** We place an edge between each primary input (primary output) variable and its compatible intermediate variable, with weight $-w_{co}$ ($-w_{ob}$).

**Sequential Depth and Loop Reduction:** For all compatible variable pairs $(v_i, v_j)$, which are assigned to the register pair $(R_i, R_j)$, merge the two registers and evaluate $P_{ij}$ (the maximum length of the shortest path between primary inputs and outputs), $C_{ij}$ (the length of the longest cycle), and $n_{ij}$ (the number of independent cycles).

**Self-Loop Elimination:** To eliminate a self-adjacent loop, we add an edge, whose weight is $+w_{self}$ between each pairs of input and output variables of a module.

The cost of merging two variables $(v_i, v_j)$ is calculated as $w_{ij} = -\alpha_1 w_{co} - \alpha_2 w_{ob} + \alpha_3 (P_i - P_0) + \alpha_4 2^{-C_{ij}} + \alpha_5 n_{ij} + \alpha_6 w_{self}$ where $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5$, and $\alpha_6$ are user defined coefficients, defining optimization goals.

Our register allocation algorithm is shown in Figure 1. The algorithm starts with greedy LEA to find the minimum number of registers (step 1). For all variable pairs the cost of merging these variables is calculated (steps 2-6). In the final step, the weighted graph coloring algorithm is applied to the ECG, to perform a testable register allocation with minimum number of registers (step 7).

![Figure 1. Testable register allocation algorithm](image)

2. Experimental results

Six circuits, with 8-bit bus width, were chosen to evaluate our register allocation method [2]. The experimental results are summarized in Table 1.

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Greedy LEA</th>
<th>Weighted Graph Coloring</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Fault Coverage</td>
<td>ATPG Time (s)</td>
</tr>
<tr>
<td>DiffEq</td>
<td>83%</td>
<td>9.26</td>
</tr>
<tr>
<td>Paulin</td>
<td>93%</td>
<td>8.23</td>
</tr>
<tr>
<td>Oven-Ctrl</td>
<td>67.9%</td>
<td>136.1</td>
</tr>
<tr>
<td>Real</td>
<td>84.2%</td>
<td>49.65</td>
</tr>
<tr>
<td>TsengA</td>
<td>84.3%</td>
<td>13.3</td>
</tr>
<tr>
<td>EWF</td>
<td>81%</td>
<td>349</td>
</tr>
</tbody>
</table>

3. Conclusion

Experimental results evaluating our method on six benchmarks are given and indicate our register allocation improves ATPG time and fault coverage.

4. References
