BDDEIRO: BDD VISUALIZATION TOOL

Mateus V. N. Gomes, Carlos E. Klock, Tiago M. G. Cardoso, Renato P. Ribas, André I. Reis

Nangate Research Lab – Instituto de Informática, UFRGS
Av. Bento Gonçalves, 9500 – CEP 91501-970, Porto Alegre, Brazil
{mvngomes,ceklock,tmgcardoso,rpribas,andreis}@inf.ufrgs.br

ABSTRACT

This paper introduces a new tool developed as a complement for existing BDD software packages. Its main objective is laying out BDDs in a visual way, making it easier for the user to check the correctness of the structures generated through any implemented method. Through a nice graphical interface, one can interact with a BDD using the available features and preview the logic function in terms of transistor networks. BDDeiro can be easily adapted to different packages due to its XML input, making it possible for developers to use the tool for different applications.

1. INTRODUCTION

BDDs have been applied successfully to a wide variety of tasks, particularly in very large scale integration (VLSI) computer-aided design (CAD) [1][2]. BDDeiro was developed to be part of BDD manipulation packages, due to the small number of existing visualization tools.

Besides regular BDD visualization, BDDeiro introduces the feature of displaying functions as transistor networks, as well as their respective pull-up and pull-down networks. This characteristic can be used not only by developers, but also as a didactic exercise for logic synthesis students. The displayed structures can be captured and saved as a *.PNG* image file, allowing generated visualizations to be easily portable.

2. BINARY DECISION DIAGRAMS

Binary Decision Diagrams (BDDs) are graph representations of Boolean expressions. In these structures, variables correspond to nodes, and their values are the edges connecting them. The graph should be examined from top to bottom and each node originates two edges: one representing the direction to follow when the variable assumes the logic value 0, and another for when it assumes the logic value 1. Therefore, the combination of values from the variables defines a path in the graph. At the end of a descending path there is a terminal node containing a binary value (0 or 1), which represents the logic function's value for a given input vector [3].

Manipulation of functions using BDDs, as well as their optimization, is very efficient. These structures have the advantage of carrying logic circuit topological information, which can make the generation of a transistor network easier; each node is associated to a multiplexer, using the CMOS pass transistor method, where the multiplexers are built with a pair of transmission gates.

3. TOOL INTERFACE

BDDeiro features different modes of visualizing a Boolean function, allowing the user to select the most convenient one at the time. Fig. 1 shows the generated view of the 2-input AND function seen as a regular BDD.

Logic functions can also be seen as a transistor network. Currently, a structure can be visualized as a netlist of NMOS (see Fig. 1), PMOS transistors, or transmission gates (parallel-connected PMOS and NMOS transistors).

Besides the visualization modes, BDDeiro also features the possibility of obtaining the pull-up and pull-down networks of a circuit. Fig. 2 shows the generated pull-down and pull-up networks for the XOR2 function.

Users can manipulate the generated display through freely moving nodes to make the visualization clearer and zooming in and out. An edge counter is available as additional information. These features allow better comprehension of specific parts of the structures. The user interface for the tool is shown in Fig. 3.

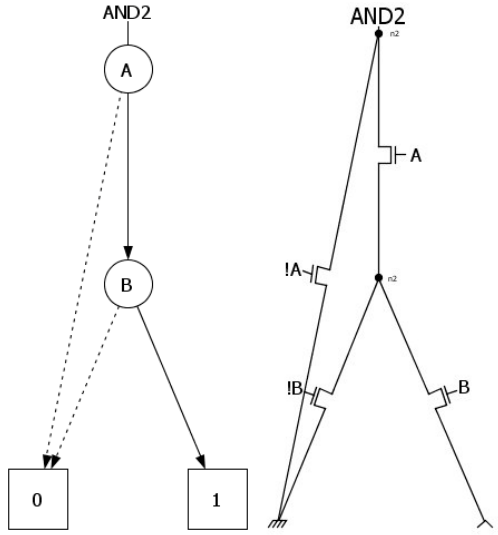

Fig. 1: AND2 – Generated view of BDD and NMOS network.

4. INPUT METHOD

BDDeiro receives as input a structure described as a GraphML file, based on XML. It consists of a language core to describe structural properties of a graph and a

flexible extension mechanism to add application-specific data [4]. In this case, information such as node levels and edge types are needed, as shown in Fig. 4. After that, the structure is described through the declaration of its nodes and edges, as seen in Fig. 5.

The main advantage of this input method is that, given the format description, developers can easily adapt any BDD package to output results in the same format.

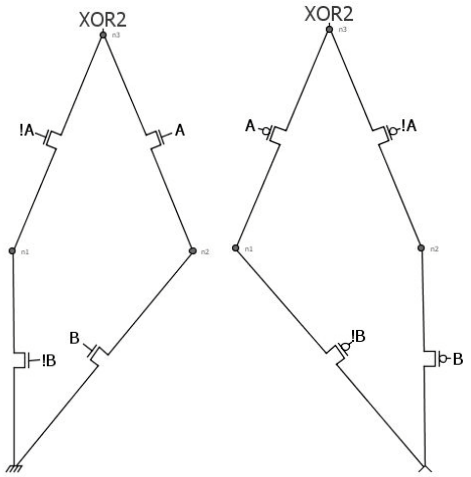

Fig. 2: XOR2 – Pull-down and pull-up networks.

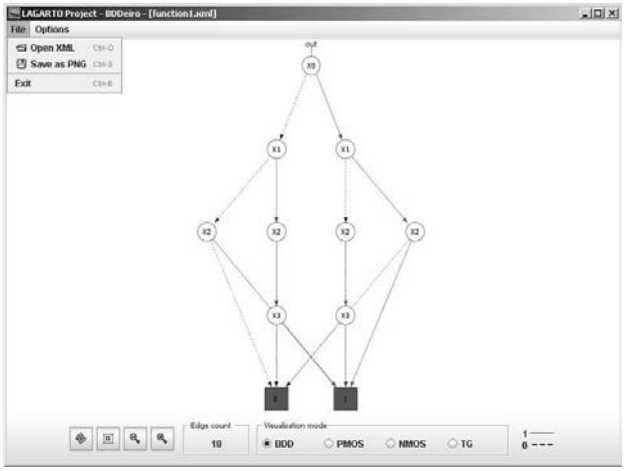

Fig. 3: BDDeiro user interface.

5. CONCLUSION AND FUTURE WORK

BDDs are key structures in VLSI CAD systems and other aspects of computer science. Currently, many software packages manipulate these structures but there is a lack of visualization tools. BDDeiro has come to fill this gap and make the process of checking obtained results easier and faster.

The rendering of different kinds of transistors and the XML format of the input opens the possibility of adapting the tool to generate visualizations for other types of structures, such as decomposed BDDs [5], being a flexible software that may be useful for different logic synthesis packages.

6. REFERENCES

[1] Minato S.I., *Binary Decision Diagrams and Applications for VLSI CAD*, Kluwer, Norwell, 1996.

[2] Drechsler R., B. Becker, *Binary Decision Diagrams: Theory and Implementation*, Kluwer, Norwell, 1998.

[3] Wagner, F. R., A. I. Reis, and R. P. Ribas, *Fundamentos de Circuitos Digitais,* Sagra Luzzatto, Porto Alegre, 2006.

[4] *The GraphML File Format*, http://graphdrawing.org/, accessed on 05/30/2006.

[5] Buch P. et al., *Logic Synthesis for Large Pass Transistor Networks*. ICCAD, p. 663, 1997.

```
<?xml version="1.0" encoding="UTF-8"?>
<graphml>

 <struct name="structure">BDD</struct>

 <!-- node attributes -->
 <key id="id" for="node" attr.name="id" attr.type="string"/>

 <key id="lb" for="node" attr.name="label" attr.type="string"/>

 <key id="tp" for="node" attr.name="type" attr.type="boolean">
    <default>false</default>
 </key>

 <key id="le" for="node" attr.name="level" attr.type="int"/>

 <key id="fn" for="node" attr.name="function" attr.type="string"/>

 <!-- edge attributes -->
 <key id="et" for="edge" attr.name="etype" attr.type="string"/>
```

Fig. 4: GraphML description of the structure's attributes.

```
<!-- graph description -->
 <graph id="G" edgedefault="directed">
   <node id="t0">
      <data key="id">t0</data>
      <data key="lb">0</data>
      <data key="tp">true</data>
      <data key="le">0</data>
   </node>
   <node id="t1">
      <data key="id">t1</data>
      <data key="lb">1</data>
      <data key="tp">true</data>
      <data key="le">0</data>
   </node>
   <node id="n1">
      <data key="id">n1</data>
      <data key="lb">X</data>
      <data key="le">2</data>
   </node>
   <edge id="e1" source="n1" target="t0">
      <data key="et">0-edge</data>
   </edge>
   <edge id="e2" source="n1" target="t1">
      <data key="et">1-edge</data>
   </edge>
 </graph>
</graphml>
```

Fig. 5: GraphML description of a BDD structure.