Logic Synthesis & Verification, Fall 2013

National Taiwan University

Programming Assignment 1

Due on 2013/10/16 before lecture

1 [Using ABC]

(10%)

(a) Use BLIF manual (`http://www.eecs.berkeley.edu/~alanmi/publications/other/blif.pdf`) to create a BLIF file representing a four-bit comparator, which outputs 1 if the input (a_3, a_2, a_1, a_0) is greater than (b_3, b_2, b_1, b_0) assuming a_3 and b_3 are the most significant bits.

(b) Perform the following steps to practice using ABC (`http://www.eecs.berkeley.edu/~alanmi/abc/`):

1. read the BLIF file into ABC (command "`read`")
2. check statistics (command "`print_stats`")
3. visualize the network structure (command "`show`")
4. convert to AIG (command "`strash`")
5. visualize the AIG (command "`show`")
6. convert to BDD (command "`collapse`")
7. visualize the BDD (command "`show_bdd`"; note that `show_bdd` only shows the first PO; command "`cone`" can be applied in combination to show other POs)

Items to turn in:

1. The BLIF file.
2. Screenshot of your ABC execution steps.
3. Results of "`show`" and "`show_bdd`".

Comment 1: For commands "`show`" and "`show_bdd`" to work, please download the binary of software "dot" from GraphViz webpage (`http://www.graphviz.org`) and put it in the same directory as the ABC binary or anywhere else in the path.

Comment 2: Make sure GSview and Ghostscript are installed on your computer. (`http://pages.cs.wisc.edu/~ghost/gsview/`) A proper path of `gsview32.exe` needs to be specified in "`\src\base\abc\abcShow.c`." For Windows 7, the path is likely to be `C:\Program Files (x86)\Ghostgum\gsview\gsview32.exe`.

2 [ABC Boolean Function Representations]

(10%) In ABC there are different ways to represent Boolean functions.

(a) Please compare the following differences.
   1. logic network in AIG vs. structurally hashed AIG (by command "`srtash`")
   2. logic network in BDD vs. collapsed BDD (by command "`collapse`")

(b) Given a structurally hashed AIG, please find a sequence of ABC commands to covert it to a logic network in SOP.

3 [Programming ABC]

(80%) Write a procedure in ABC environment to iterate over the objects of the network. First print the network types (netlist, logic, strash, etc.) and network functionality (SOP, BDD, AIG, etc.). Then visit each object and list its **ID number**, **type**, and **fanin ID numbers** for each object on a separate line (10% bonus for reversed topological order traversal). If a network is of "strash" functionality (i.e., AIG), further indicate whether each fanin of a node is inverted. (Please refer to `\src\base\abc\abc.h`.) Integrate this procedure into ABC, so that running command "`print_ntkobj`" would invoke your code, and print the result.

Programming help:

Example of code to iterate over the objects

```
void Abc_NtkCleanCopy( Abc_Ntk_t * pNtk )
{
  Abc_Obj_t * pObj;
  int i;
  Abc_NtkForEachObj( pNtk, pObj, i )
    pObj->pCopy = NULL;
}
```

Example of code to create new command "`print_ntkobj`"

Call the new procedure (say, `Abc_NtkPrintObjs`) from `Abc_CommandPrintNtkObj()` in file `\src\base\abci\abc.c`

```
int Abc_CommandPrintNtkObj( Abc_Frame_t * pAbc, int argc, char ** argv)
{
  ⋮
  Abc_NtkPrintObjs( pNtk );
  ⋮
}
```

Items to turn in:

1. Your codes of `Abc_CommandPrintNtkObj` and other new function calls if there is any. (Please indicate if the reversal is in a reversed topological order.)
2. Screenshots of ABC running your new command "`print_ntkobj`" on the comparator example (under representations: strash, collapse, and logic network in SOP, AIG, BDD) to verify correctness.