Logic Synthesis & Verification, Fall 2012

National Taiwan University

Programming Assignment 1

Due on 2012/10/17 before lecture

1 [Using ABC]

(a) Use BLIF manual (`http://www.eecs.berkeley.edu/~alanmi/publications/other/blif.pdf`) to create a BLIF file representing a two-bit multiplier.

(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 by email to instructor:

1. the BLIF file
2. a screenshot of your ABC execution steps
3. the 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 [Programming ABC]

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. 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 "`lsv1`" 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 "lsv1"

Call the new procedure (say, `Abc_NtkPrintObjs`) from `Abc_CommandLSV1()` in file `\src\base\abci\abc.c`

```
int Abc_CommandLSV1( Abc_Frame_t * pAbc, int argc, char ** argv)
{
  ⋮
  Abc_NtkPrintTypesAndObjs( pNtk );
  ⋮
}
```

Items to turn in by email to instructor:

1. your codes of `Abc_CommandLSV1` and other new function calls if there is any
2. screenshots of ABC running your new command "`lsv1`" on the multiplier example to verify correctness