Example of using ABC for synthesis and mapping

Make sure to have *.lib or *.genlib library and abc.rc (aliases) files in the same directory. You will also need abc.rc file with the abbreviations.

```
trollinger [tests]> abc
UC Berkeley, ABC 1.01 (compiled May 15 2012 10:32:23)
abc 01> gen -a -N 6 adder6.blif  // generate 6-bit RC adder
Warning: The network contains hierarchy.
Hierarchy reader flattened 6 instances of logic boxes and left 0 black boxes.
abc 02> show
abc 02> strash // structural hashing
abc 03> show  // shows AIG netlist
abc 03> dc2   // standard synthesis script
abc 04> show
abc 04> logic // goes back to SOP form, flattened
abc 05> show
abc 05> read_library stdcell.lib // or mcnc.genlib
The number of gates read = 51.
Read 51 gates from file "stdcell.lib".
Selected 47 functionally unique gates. Time =    0.01 sec
Created 17 rules and 57 matches. Time =    0.00 sec
abc 05> map  // map on st.cell library (load the library first !)
Warning: The network was strashed and balanced before mapping.
A simple supergate library is derived from gate library "stdcell.lib".
Loaded 146 unique 5-input supergates from "stdcell.super".  Time =    0.00 sec
abc 06> show     // shows mapped netlist with SOP nodes
abc 06> show -g  // shows mapped netlist with gates (very useful !)
abc 06> print_gates
AND2X1     Fanin =  2   Instance =        2   Area =       8.00     5.37 %
AND3X1     Fanin =  3   Instance =        1   Area =       5.00     3.36 %
AOI211X1   Fanin =  4   Instance =        1   Area =       5.00     3.36 %
.....................
.....................
OR2X1      Fanin =  2   Instance =        1   Area =       4.00     2.68 %
XOR2X1     Fanin =  2   Instance =        4   Area =      28.00    18.79 %
XOR3X1     Fanin =  3   Instance =        1   Area =      17.00    11.41 %
TOTAL                   Instance =       31   Area =     149.00   100.00 %
abc 06> strash
abc 07> show     // back to AIG
abc 07> if -K 4  // FPGA mapping onto LUT-4 (replaces command fpga)
abc 08> show     // shows netlist of LUT nodes (in SOP format)
abc 08> show -?
show: unknown option ?
usage: show [-srgfh]
       visualizes the network structure using DOT and GSVIEW
        -s    : toggles visualization of sequential networks [default = no].
        -r    : toggles ordering nodes in reverse order [default = yes].
        -g    : toggles printing gate names for mapped network [default = no].
        -f    : toggles visualizing flop dependency graph [default = no].
        -h    : print the command usage
abc 08> write adder6-lut4.blif
abc 08> write adder6-lut4.v
abc 08> q
```