# Logic Synthesis and Verification PA1 Report

R10522526 羅崇榮

2. [Using ABC]

(a) Verification

input:

pat.txt

```
0 0 0 0
0 0 0 1
0 0 1 0
0 0 1 1
0 1 0 0
0 1 0 1
0 1 1 0
0 1 1 1
1 0 0 0
1 0 0 1
1 0 1 0
1 0 1 1
1 1 0 0
1 1 0 1
1 1 1 0
1 1 1 1
```

output:

```
abc 08> sim -v -A pat.txt
0000 0000
0001 0000
0010 0000
0011 0000
0100 0000
0101 0001
0110 0010
0111 0011
1000 0000
1001 0010
1010 0100
1011 0110
1100 0000
1101 0011
1110 0110
1111 1001
Simulation of 1 frame with 1 word asserted output 0 in frame 0. Time =     0.00
sec
```

(b)

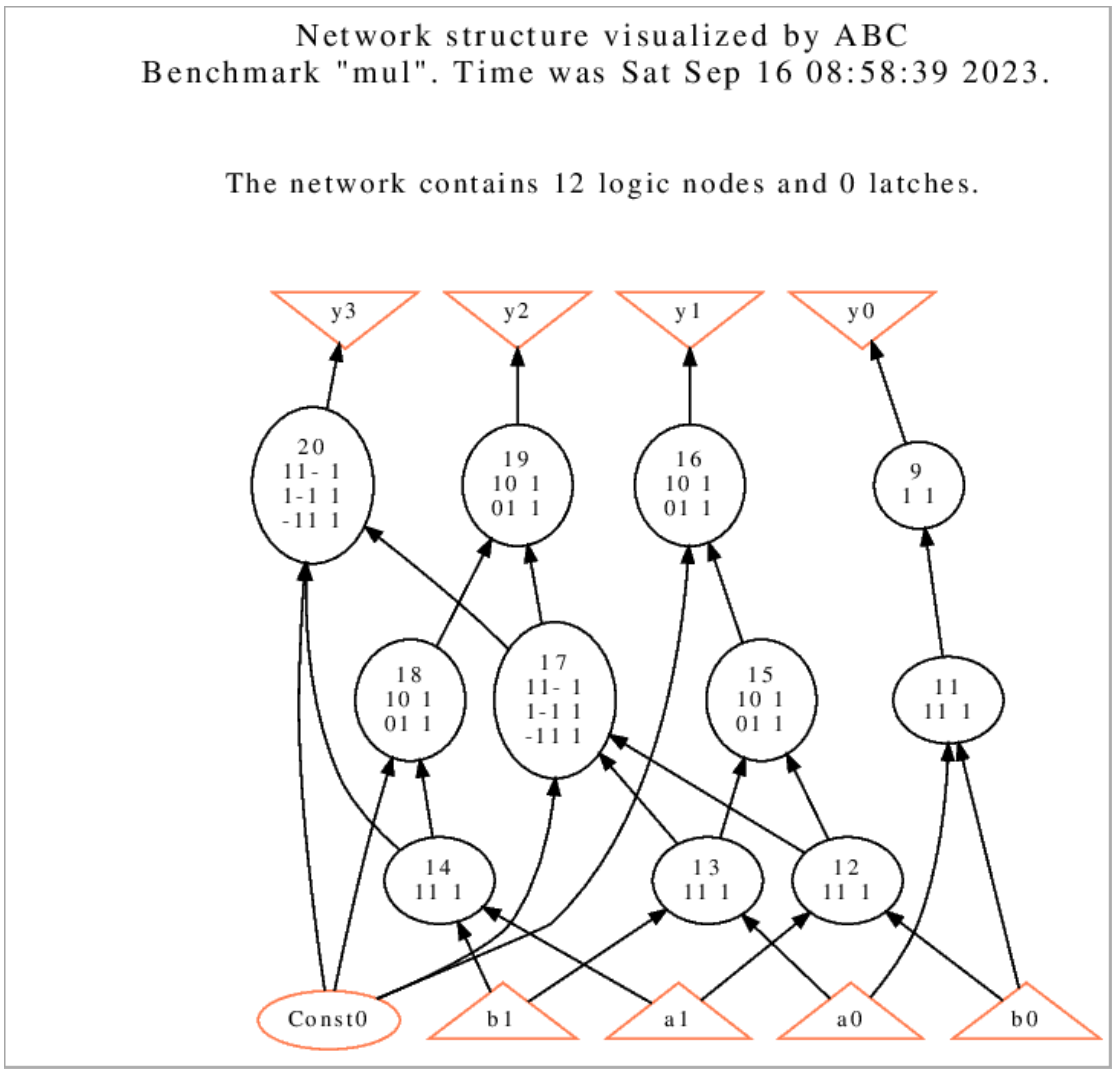
(1) read

```
abc 03> read mul.blif
Warning: Constant-0 drivers added to 1 non-driven nets in network "mul":
0
Hierarchy reader flattened 4 instances of logic boxes and left 0 black boxes.
```

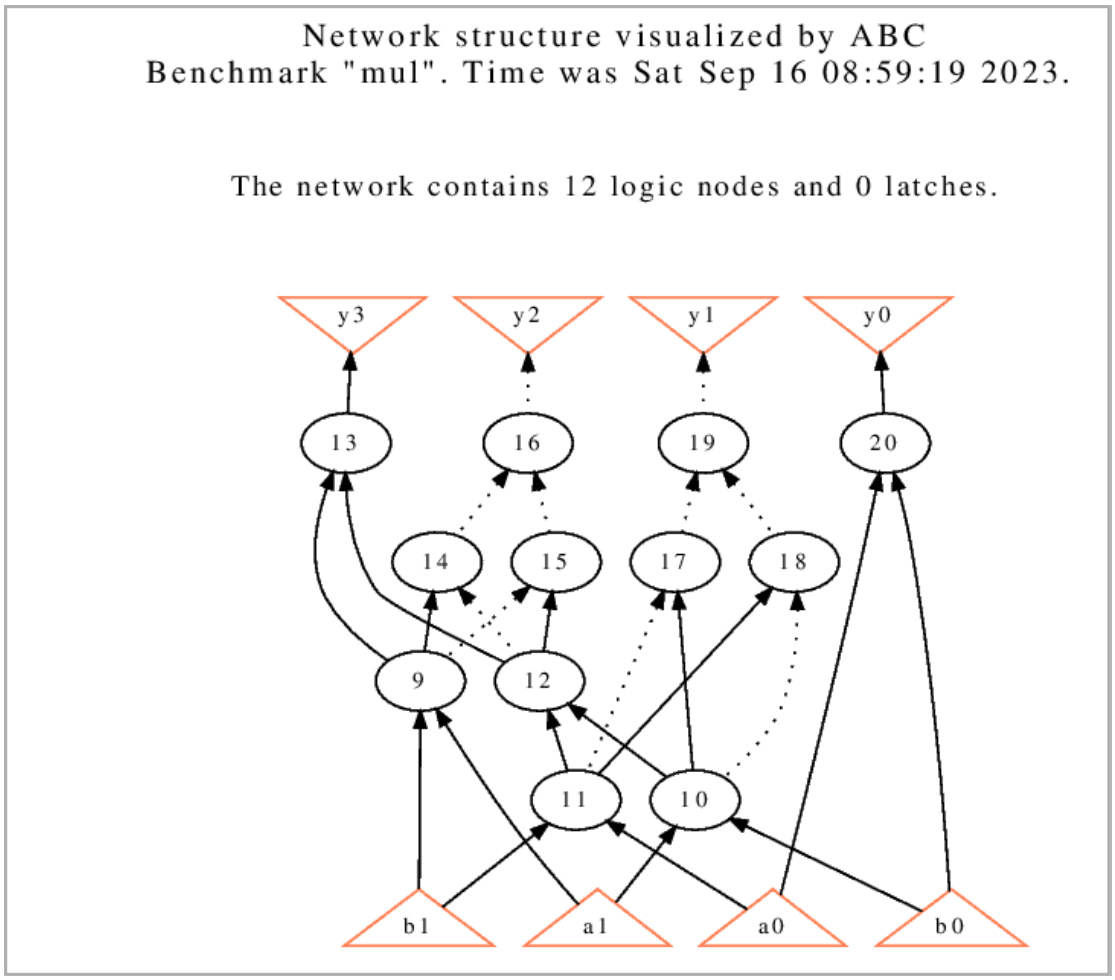
(2) print_stats

```
abc 04> print_stats
mul                       : i/o =    4/    4  lat =    0  nd =    12  edge =
    23  cube =    19  lev = 3
```

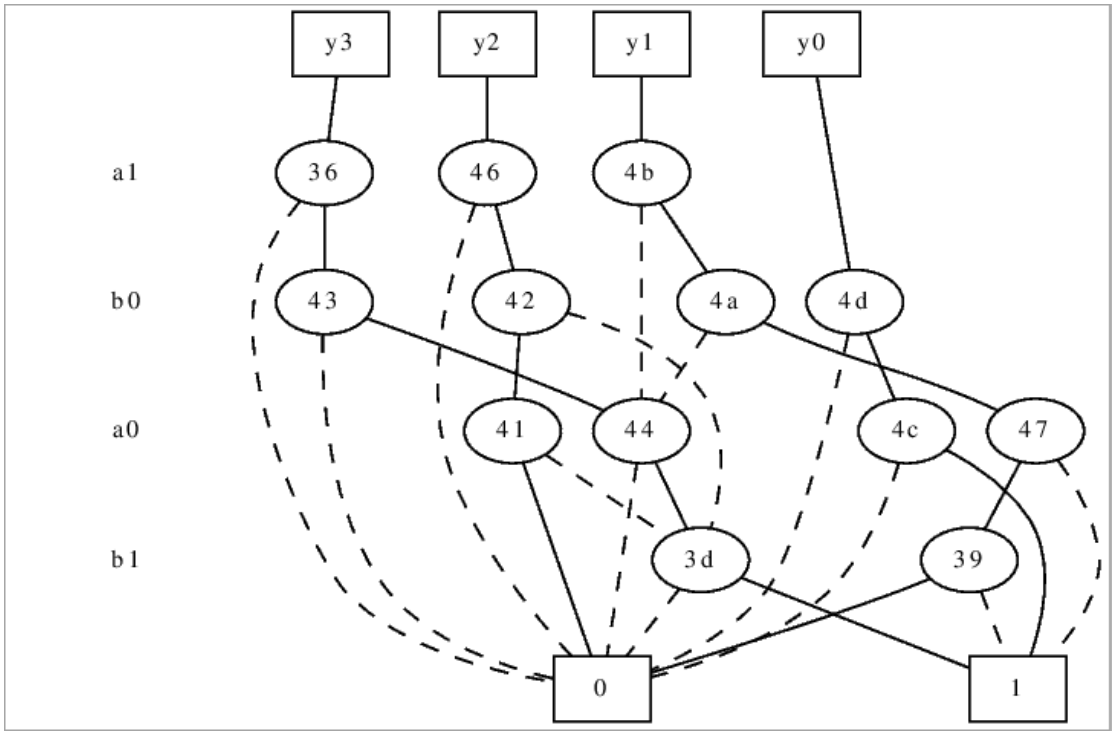
(3) show



(4)(5) strash + show



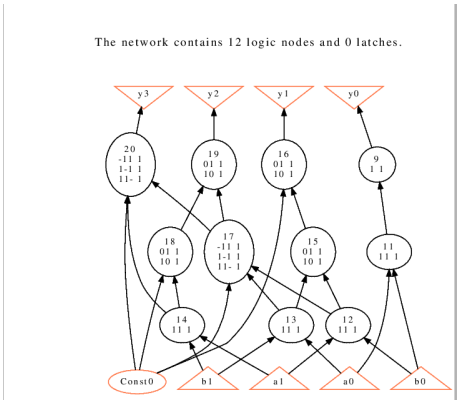

(6)(7) collapse + show_bdd -g



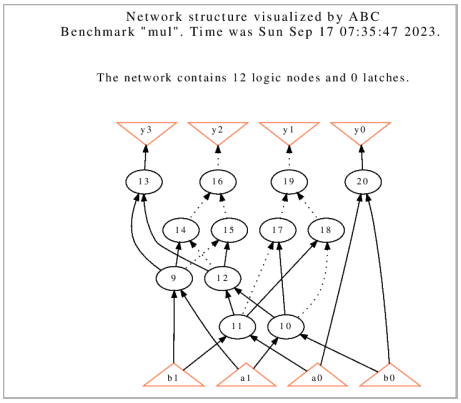
3. [ABC Boolean Function Representations]

(a)

1. ‘aig’ simply convert the logic network to AIG, while ‘strash’ transforms the logic network into AIG by one-level structural hashing.

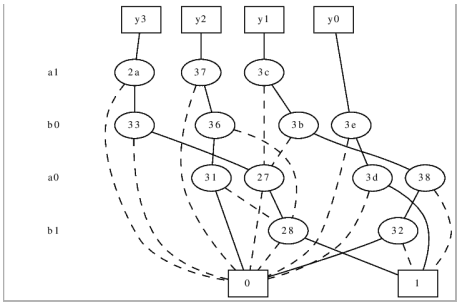



(a) aig

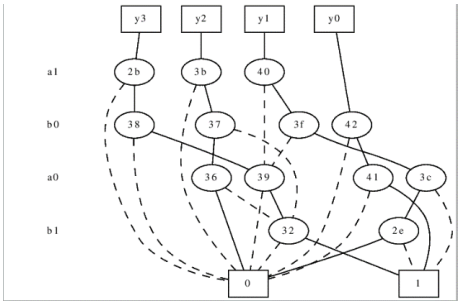



(b) strash

2. ‘bdd’ simply convert the logic network to BDD, while ‘collapse’ transforms the logic network into BDD by building global functions.

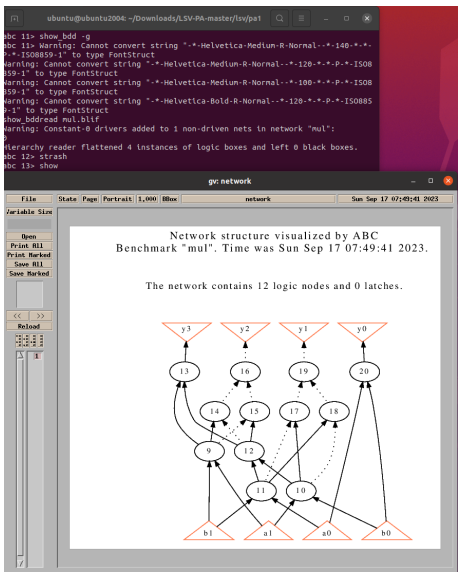



(a) bdd

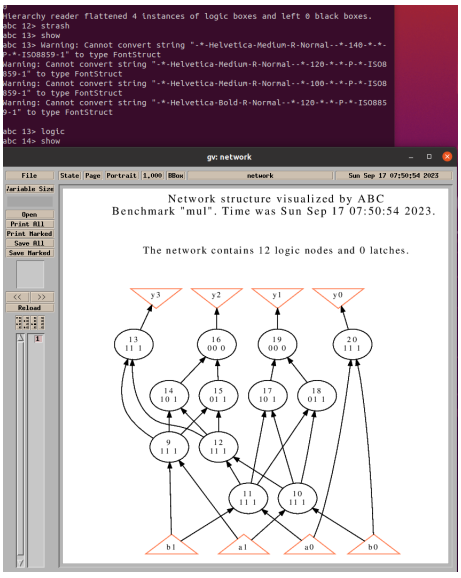



(b) collapse

(b) logic




(a) strash




(b) logic