

# Logic Synthesis & Verification PA1 report

R11943172 黃冠瑋

2.

(b).

1.

```
abc 01> read lsv/pa1/mul.blif
abc 02> █
```

2.

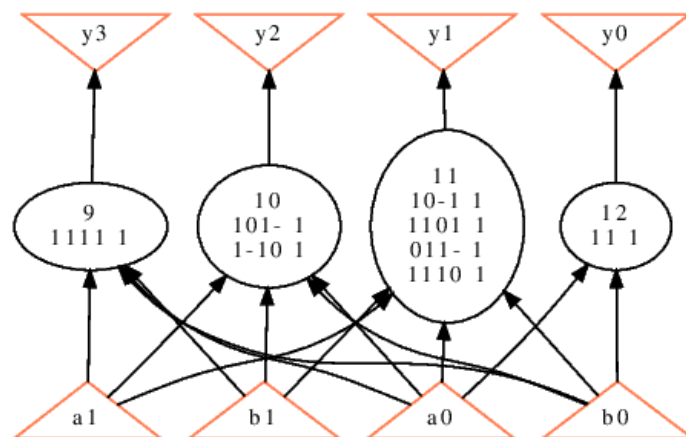
```
abc 02> print_stats
mul                                     : i/o =   4/   4  lat =   0  nd =   4  edge =   14  cube =   8  lev = 1
abc 02> █
```

3.

---

Network structure visualized by ABC  
Benchmark "mul". Time was Mon Sep 11 20:17:29 2023.

The network contains 4 logic nodes and 0 latches.



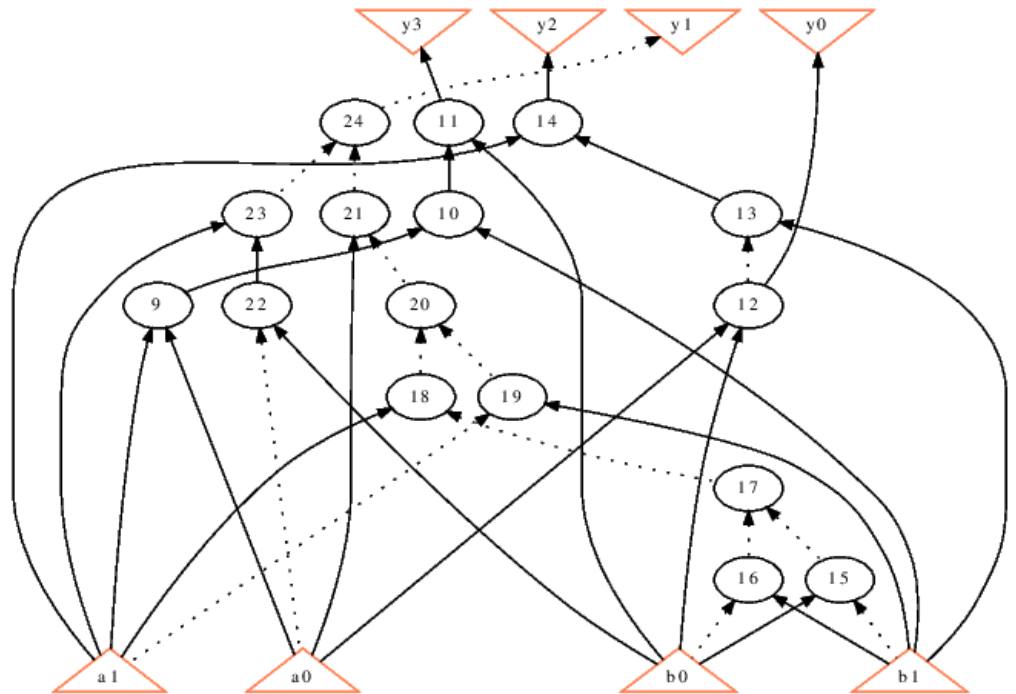
4.

```
abc 03> strash
abc 04> print_stats
mul                                     : i/o =   4/   4  lat =   0  and =   16  lev = 6
abc 04> █
```

5.

Network structure visualized by ABC  
Benchmark "mul". Time was Mon Sep 11 20:20:58 2023.

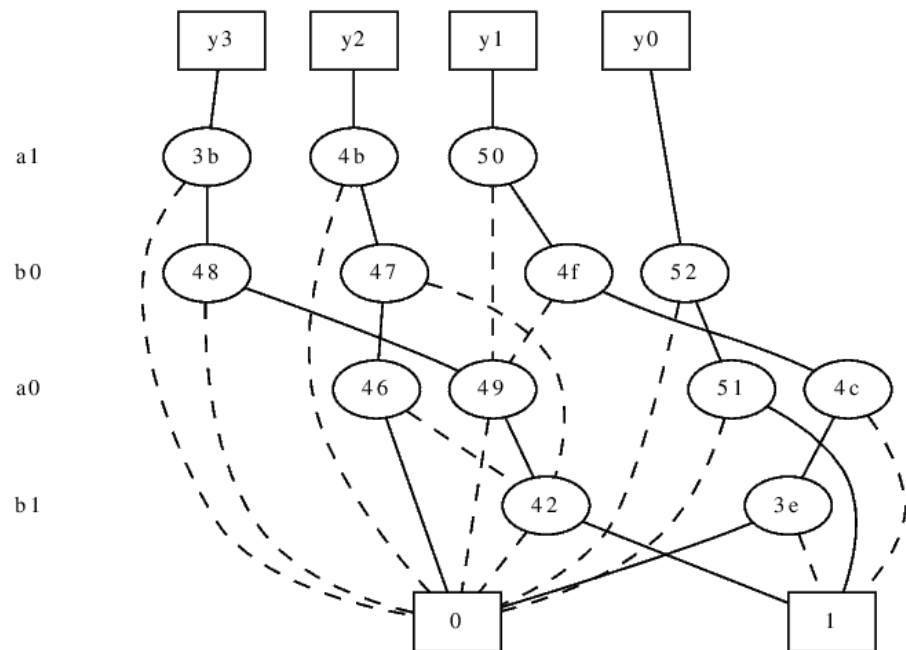
The network contains 16 logic nodes and 0 latches.



6.

```
abc 04> collapse
abc 05> print_stats
mul                               : i/o = 4/ 4 lat = 0 nd = 4 edge = 14 bdd = 14 lev = 1
abc 05> █
```

7.

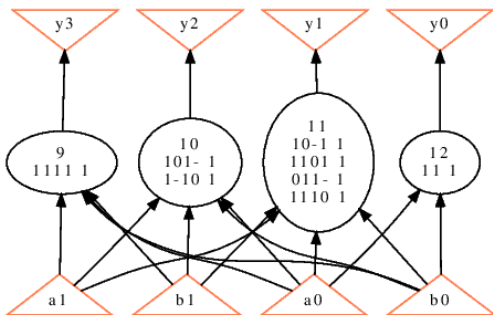
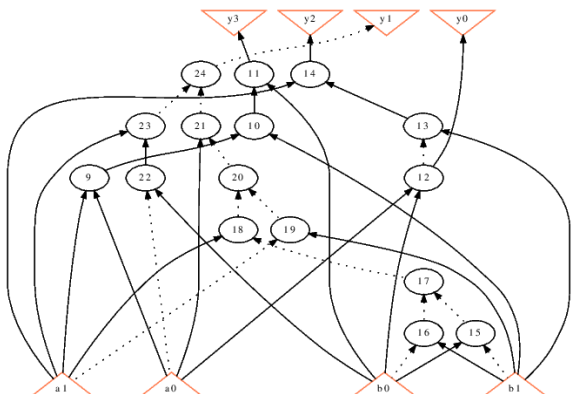


3.

(a)

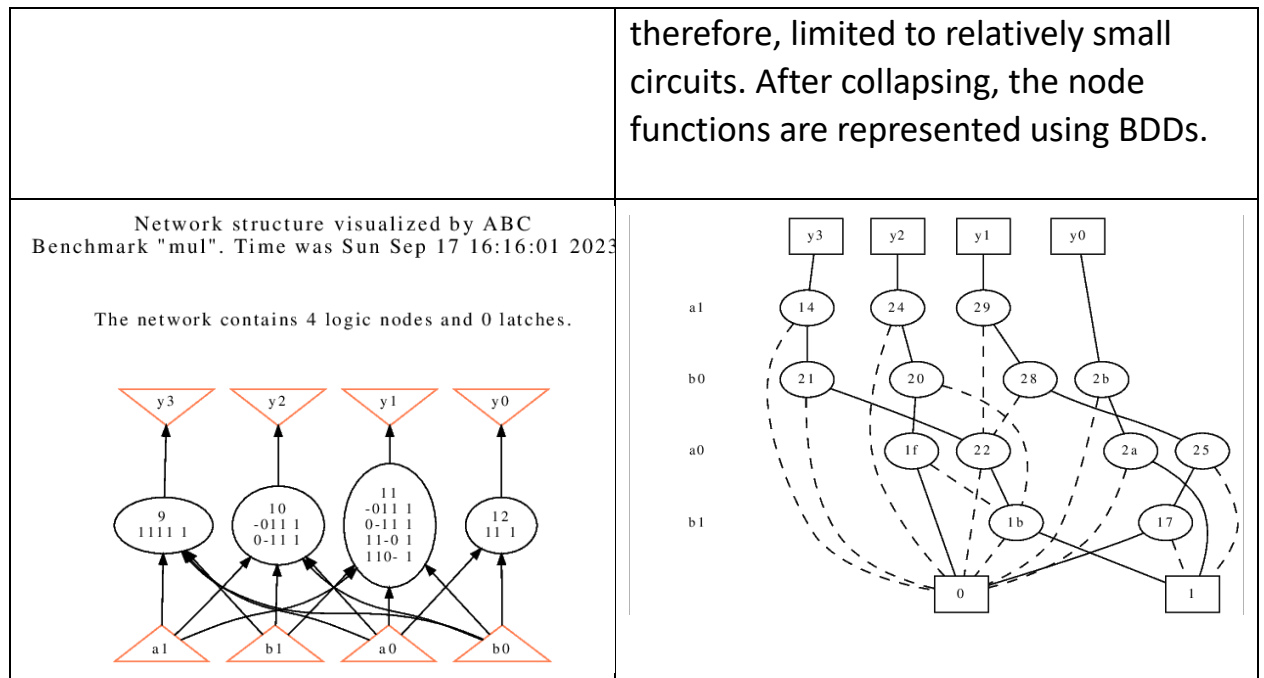
reference: <https://people.eecs.berkeley.edu/~alanmi/abc/>

(1)

Command "aig"	Command "strash"
Converts local functions of the nodes to AIGs.	Transforms the current network into an AIG by one-level structural hashing. The resulting AIG is a logic network composed of two-input AND gates and inverters represented as complemented attributes on the edges.
<p>Network structure visualized by ABC hmark "mul". Time was Mon Sep 11 20:17:29 202</p> <p>The network contains 4 logic nodes and 0 latches.</p> 	<p>Network structure visualized by ABC Benchmark "mul". Time was Mon Sep 11 20:20:58 2023.</p> <p>The network contains 16 logic nodes and 0 latches.</p> 

(2)

Command "bdd"	Command "collapse"
Converts local functions of the nodes to BDDs	Recursively composes the fanin nodes into the fanout nodes resulting in a network, in which each CO is produced by a node, whose fanins are CIs. Collapsing is performed by building global functions using BDDs and is,



(b)

Use command "logic" then command "lsv\_print\_nodes" to find the SOP expressed function. The result is below:

```
abc 02> strash
abc 03> logic
abc 04> lsv_print_nodes
Object Id = 9, name = n9
    Fanin-0: Id = 1, name = a1
    Fanin-1: Id = 2, name = a0
The SOP of this node:
11 1
Object Id = 10, name = n10
    Fanin-0: Id = 3, name = b1
    Fanin-1: Id = 9, name = n9
The SOP of this node:
11 1
Object Id = 11, name = n11
    Fanin-0: Id = 4, name = b0
    Fanin-1: Id = 10, name = n10
The SOP of this node:
11 1
```

```
Object Id = 12, name = n12
  Fanin-0: Id = 2, name = a0
  Fanin-1: Id = 4, name = b0
The SOP of this node:
11 1
Object Id = 13, name = n13
  Fanin-0: Id = 1, name = a1
  Fanin-1: Id = 12, name = n12
The SOP of this node:
10 1
Object Id = 14, name = n14
  Fanin-0: Id = 3, name = b1
  Fanin-1: Id = 13, name = n13
The SOP of this node:
11 1
Object Id = 15, name = n15
  Fanin-0: Id = 1, name = a1
  Fanin-1: Id = 4, name = b0
The SOP of this node:
11 1
Object Id = 16, name = n16
  Fanin-0: Id = 3, name = b1
  Fanin-1: Id = 15, name = n15
The SOP of this node:
10 1
```

```
Object Id = 17, name = n17
  Fanin-0: Id = 2, name = a0
  Fanin-1: Id = 16, name = n16
The SOP of this node:
11 1
Object Id = 18, name = n18
  Fanin-0: Id = 2, name = a0
  Fanin-1: Id = 3, name = b1
The SOP of this node:
11 1
Object Id = 19, name = n19
  Fanin-0: Id = 4, name = b0
  Fanin-1: Id = 18, name = n18
The SOP of this node:
10 1
Object Id = 20, name = n20
  Fanin-0: Id = 1, name = a1
  Fanin-1: Id = 19, name = n19
The SOP of this node:
11 1
Object Id = 21, name = n21
  Fanin-0: Id = 17, name = n17
  Fanin-1: Id = 20, name = n20
The SOP of this node:
00 0
```