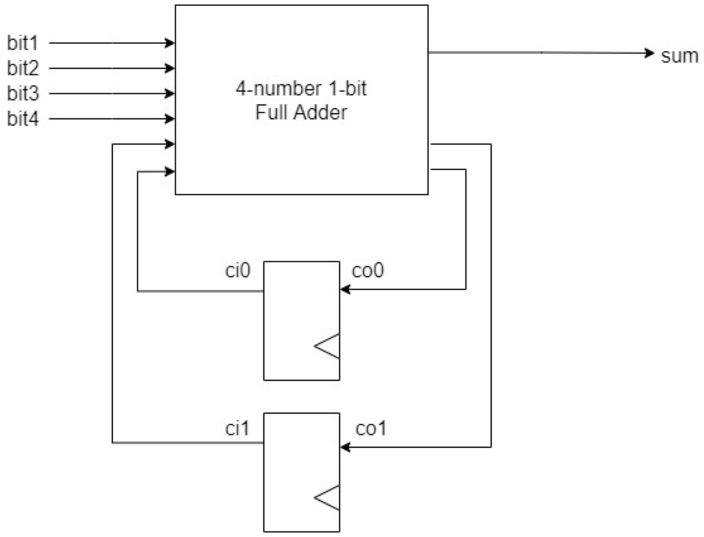
LSV-PA1 Part 1&2 Report

r09943186 Chaoho Hsieh 謝兆和

1. (a) Create a BLIF file representing a four-number serial adder.



Above is the Block diagram of my circuit. I model the 4-bit serial adder as a 4-bit combinational full adder with 6 inputs (4 one-bit PIs and 2-bit carry-in) and 3 outputs (1-bit PO and 2-bit carry-out).

The maximum sum of a 4-bit full adder is 7. Therefore, we need a 2-bit flip-flop to store the carry-out of each clock cycle. The following are the Boolean function of sum, CO_0, and CO_1.

$$\mathrm{sum} = a \oplus b \oplus c \oplus d \oplus \mathrm{CI}_0$$

$$\mathrm{CO}_0 = \mathrm{CI}_1'(a'bc'd + ab'cd' + abc'\mathrm{CI}_0' + bc'd'\mathrm{CI}_0 + a'b'd\mathrm{CI}_0 + a'cd\mathrm{CI}_0' + a'cd'\mathrm{CI}_0 + bcd'\mathrm{CI}_0' + ab'c'\mathrm{CI}_0 + ab'd\mathrm{CI}_0')$$
$$+\mathrm{CI}_1(a'b'c'd' + a'b'c'\mathrm{CI}_0' + a'b'd'\mathrm{CI}_0' + a'c'd'\mathrm{CI}_0' + b'c'd'\mathrm{CI}_0' + abcd + abc\mathrm{CI}_0 + abd\mathrm{CI}_0 + acd\mathrm{CI}_0 + bcd\mathrm{CI}_0)$$

$$\mathrm{CO}_1 = abcd + abc\mathrm{CI}_0 + abd\mathrm{CI}_0 + acd\mathrm{CI}_0 + bcd\mathrm{CI}_0 + \mathrm{CI}_1(ab + ac + ad + a\mathrm{CI}_0 + bc + bd + b\mathrm{CI}_0 + cd + c\mathrm{CI}_0 + d\mathrm{CI}_0)$$

Where a, b, c, d in the equations correspond to bit1, bit2, bit3, bit4 in the block diagram.

The following is the BLIF code, which includes the truth table of the 4-bit combinational full adder.

```
.model 4NumSerialAdder
.inputs bit1 bit2 bit3 bit4
.outputs sum
.subckt 4NumFullAdder a=bit1 b=bit2 c=bit3 d=bit4 ci1=carryIn1 ci0=carryIn0 s=sum co1=carryOut1
co0=carryOut0
.latch carryOut1 carryIn1 0
.latch carryOut0 carryIn0 0
.end

.model 4NumFullAdder
.inputs a b c d ci1 ci0
```

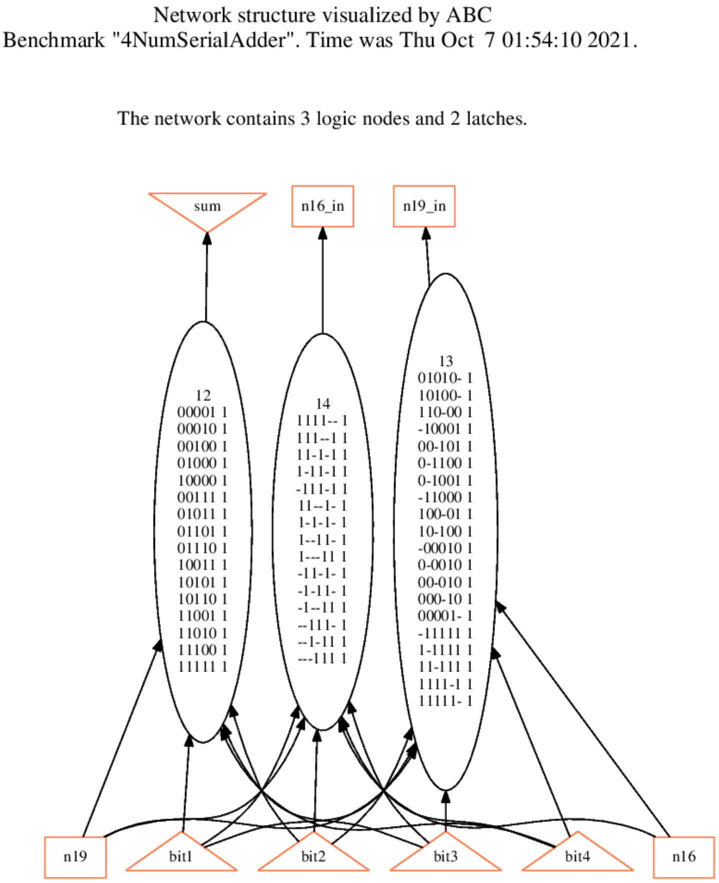
```
.names a b c d ci0 s
00001 1
00010 1
00100 1
01000 1
10000 1
00111 1
01011 1
01101 1
01110 1
10011 1
10101 1
10110 1
11001 1
11010 1
11100 1
11111 1
```

```
.names a b c d ci1 ci0 co1
1111-- 1
111--1 1
11-1-1 1
1-11-1 1
-111-1 1
11--1- 1
1-1-1- 1
1--11- 1
1---11 1
-11-1- 1
-1-11- 1
-1--11 1
--111- 1
--1-11 1
---111 1
.end
```

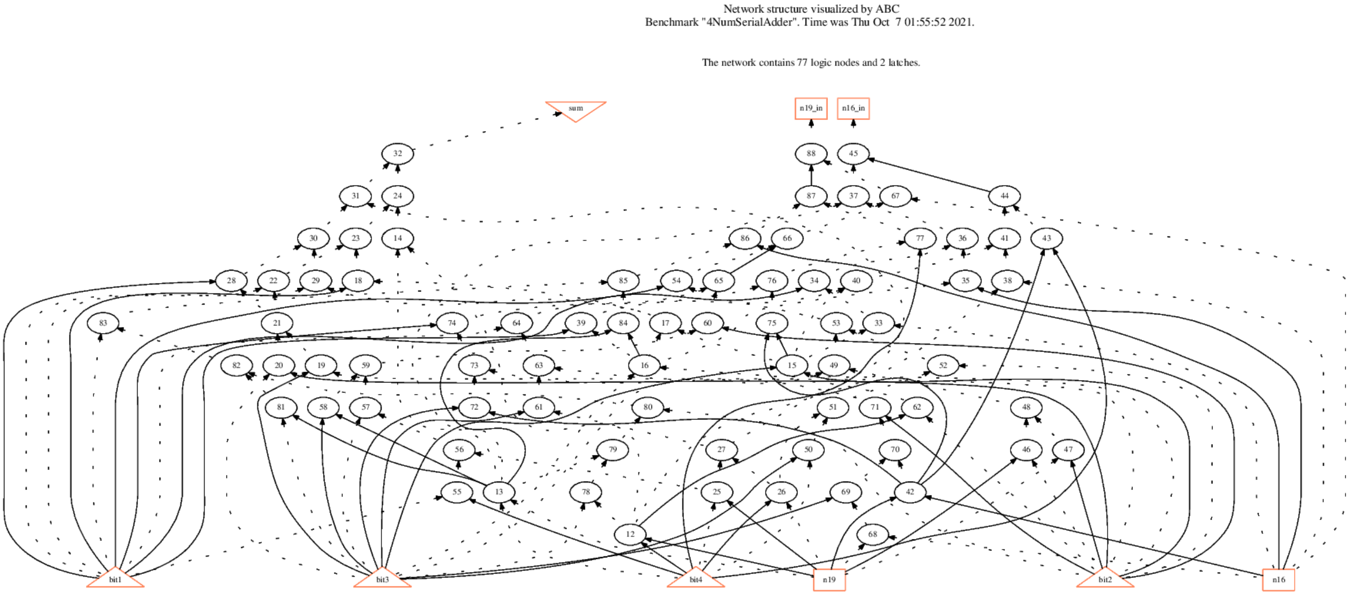
```
.names a b c d ci1 ci0 co0
01010- 1
10100- 1
110-00 1
-10001 1
00-101 1
0-1100 1
0-1001 1
-11000 1
100-01 1
10-100 1
-00010 1
0-0010 1
00-010 1
000-10 1
00001- 1
-11111 1
1-1111 1
11-111 1
1111-1 1
11111- 1
```

```
.outputs s co1 co0
```

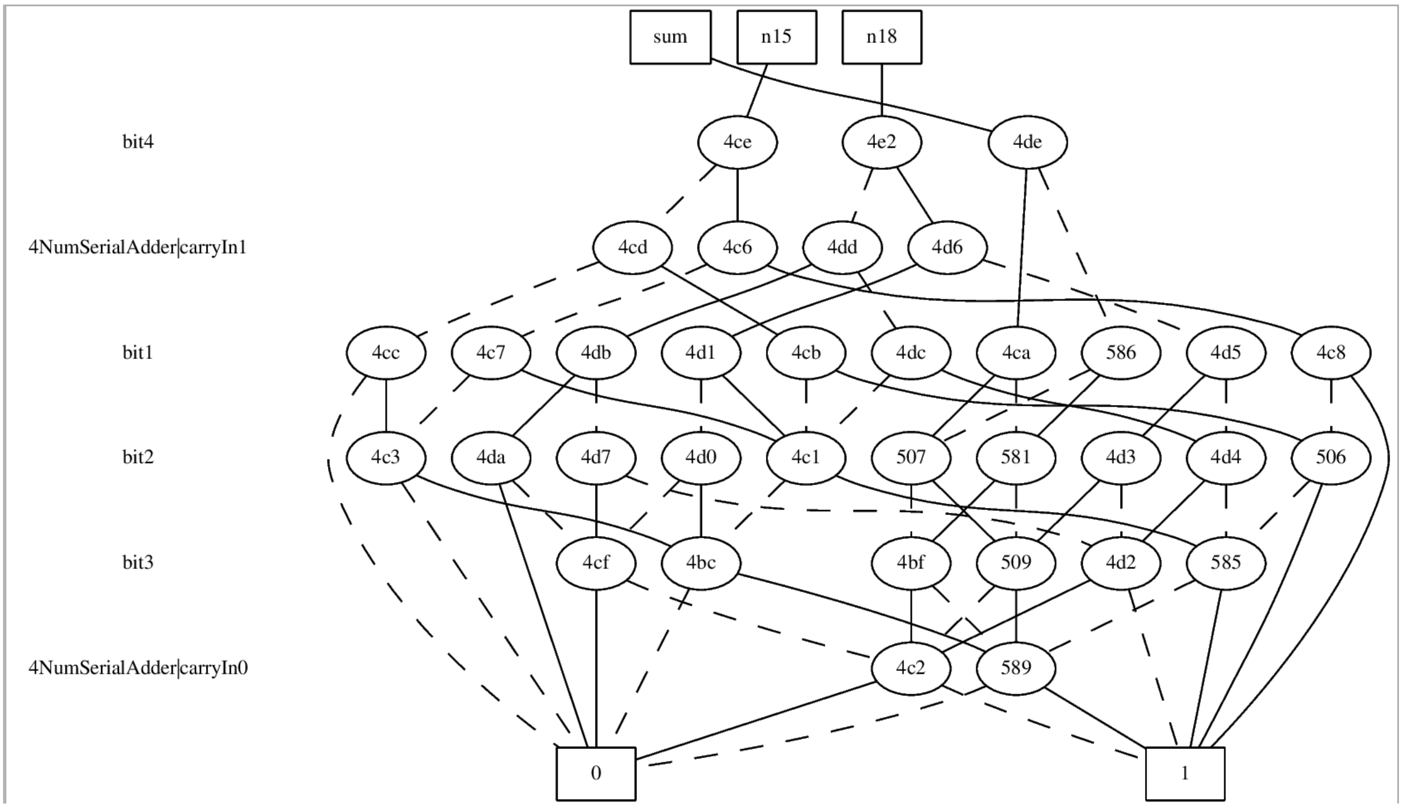
1. (b) Original Network (after step 3)



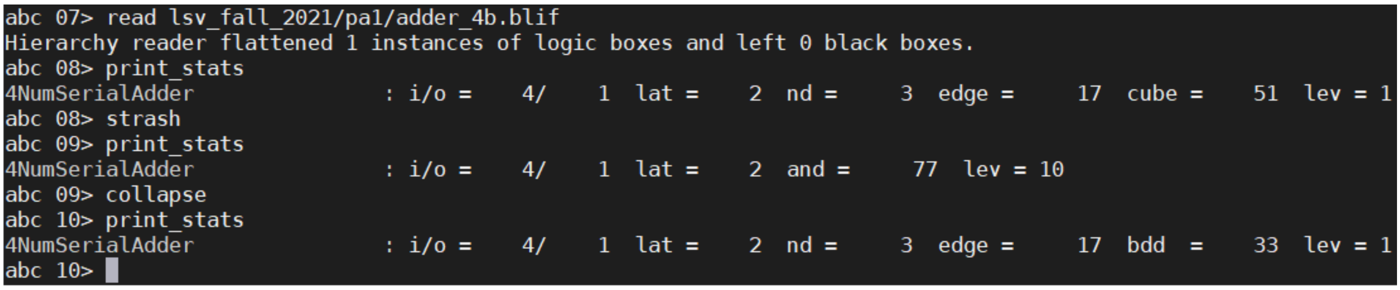
Strashed (after step 5)


Network structure visualized by ABC
Benchmark "4NumSerialAdder". Time was Thu Oct 7 01:55:52 2021.
The network contains 77 logic nodes and 2 latches.
sum
n19_in
n16_in
bit1
bit3
bit4
n19
bit2
n16

Collapsed (after step 7)



Stats of these 3 networks

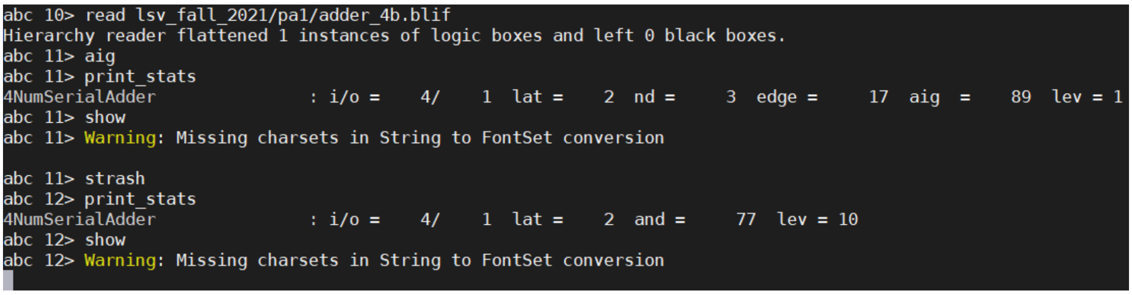


```
abc 07> read lsv_fall_2021/pa1/adder_4b.blif
Hierarchy reader flattened 1 instances of logic boxes and left 0 black boxes.
abc 08> print_stats
4NumSerialAdder                 : i/o =    4/    1  lat =    2  nd =     3  edge =     17  cube =    51  lev = 1
abc 08> strash
abc 09> print_stats
4NumSerialAdder                 : i/o =    4/    1  lat =    2  and =     77  lev = 10
abc 09> collapse
abc 10> print_stats
4NumSerialAdder                 : i/o =    4/    1  lat =    2  nd =     3  edge =     17  bdd  =    33  lev = 1
abc 10> 
```

2. (a)-1

The difference between command `aig` and `strash` is that `aig` converts the individual nodes to AIGs, while keeping the larger structure of the original network. On the contrary, `collapse` converts the entire network into a single completely-reduced AIG.

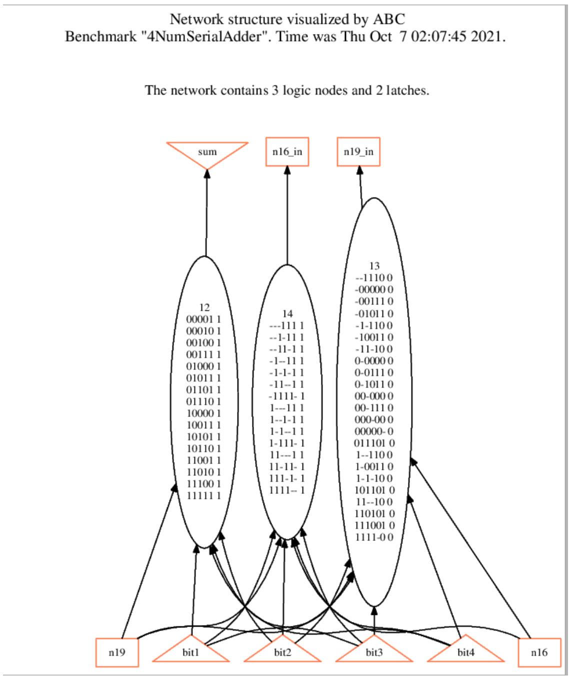
Terminal Screenshot:



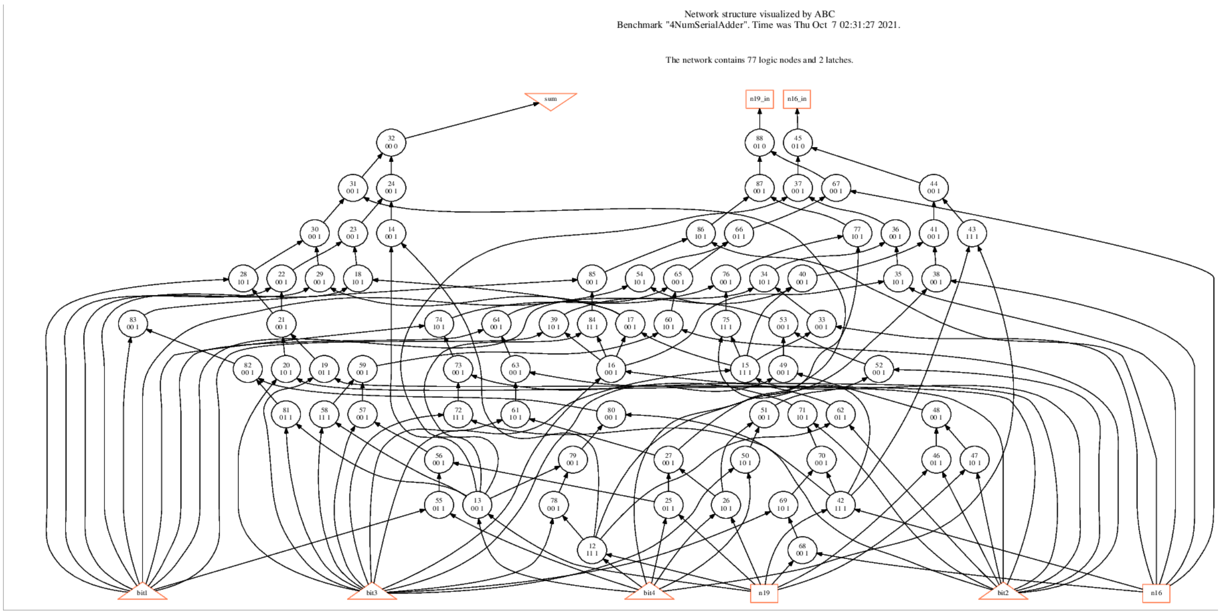
```
abc 10> read lsv_fall_2021/pa1/adder_4b.blif
Hierarchy reader flattened 1 instances of logic boxes and left 0 black boxes.
abc 11> aig
abc 11> print_stats
4NumSerialAdder               : i/o =    4/    1  lat =    2  nd =     3  edge =     17  aig  =    89  lev = 1
abc 11> show
abc 11> Warning: Missing charsets in String to FontSet conversion

abc 11> strash
abc 12> print_stats
4NumSerialAdder               : i/o =    4/    1  lat =    2  and =     77  lev = 10
abc 12> show
abc 12> Warning: Missing charsets in String to FontSet conversion
```

After “aig” command



After “strash” command

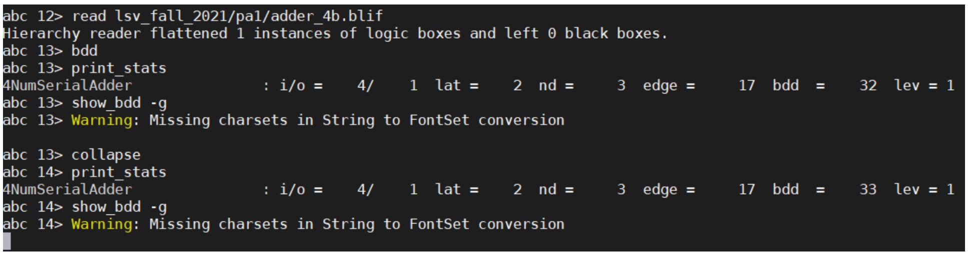

Network structure visualized by ABC
Benchmark "4NumSerialAdder". Time was Thu Oct 7 02:31:27 2021.
The network contains 77 logic nodes and 2 latches.
sum
n19_in
n16_in
bit1
bit3
bit4
n19
bit2
n16

2. (a)-2

The difference between command `bdd` and `collapse` is that `bdd` converts the individual nodes to BDDs, while keeping the larger structure of the original network. On the contrary, `collapse` converts the entire network into a single completely-reduced BDD.

In this example, however, the two commands appear to yield the identical result by using `show-bdd` to visualize the network.

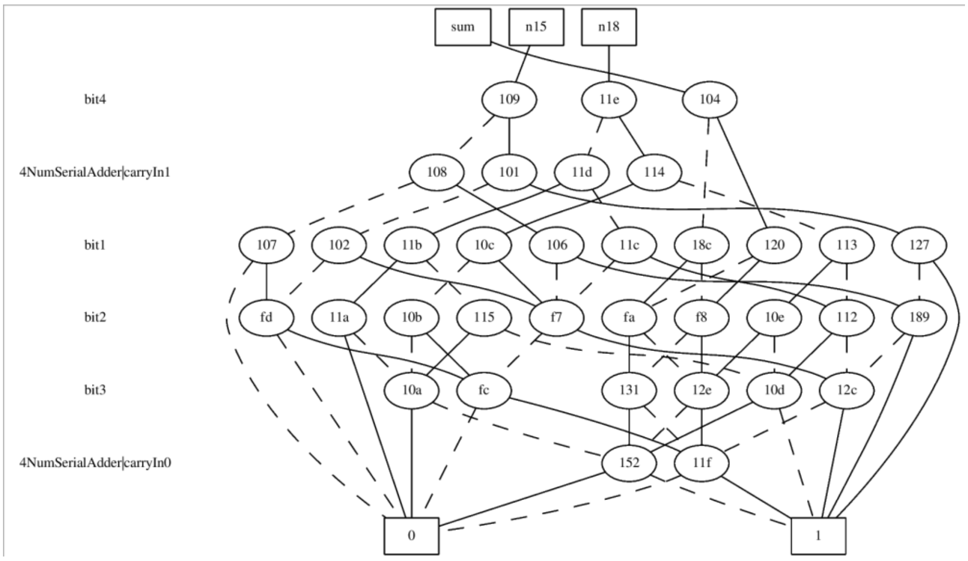
Terminal Screenshot:



```
abc 12> read lsv_fall_2021/pa1/adder_4b.blif
Hierarchy reader flattened 1 instances of logic boxes and left 0 black boxes.
abc 13> bdd
abc 13> print_stats
4NumSerialAdder              : i/o =    4/    1  lat =    2  nd =     3  edge =     17  bdd  =     32  lev = 1
abc 13> show_bdd -g
abc 13> Warning: Missing charsets in String to FontSet conversion

abc 13> collapse
abc 14> print_stats
4NumSerialAdder              : i/o =    4/    1  lat =    2  nd =     3  edge =     17  bdd  =     33  lev = 1
abc 14> show_bdd -g
abc 14> Warning: Missing charsets in String to FontSet conversion
```

After either `bdd` or `collapse` command



2.(b)

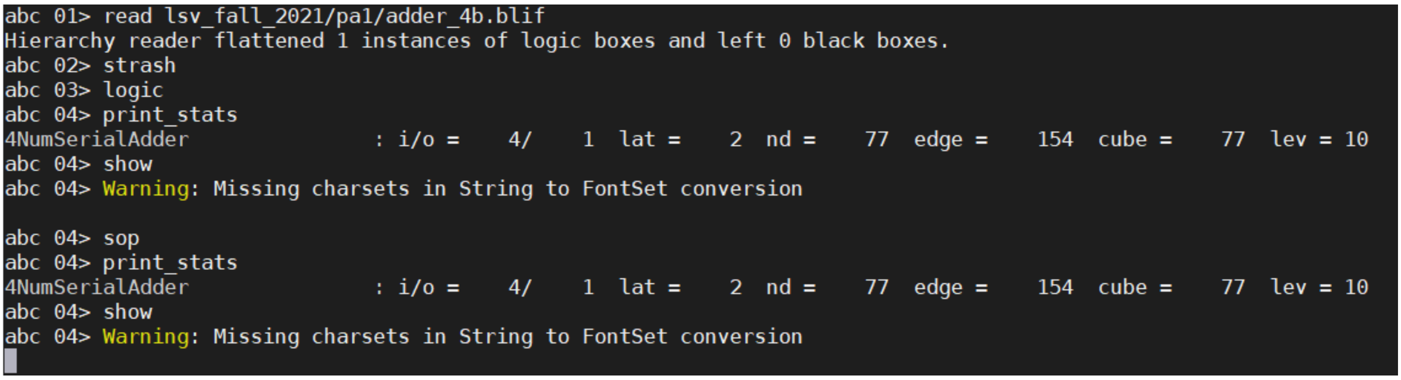
Command sequence:

```
logic
sop
```

Command `sop` converts local functions of the nodes to SOPs, while command `logic` transforms the AIG into a logic network with the SOP representation of the two-input AND-gates. Note that `sop` can only be used after `logic`.

In this example, however, the two commands appears to yield the identical result as `strash`.

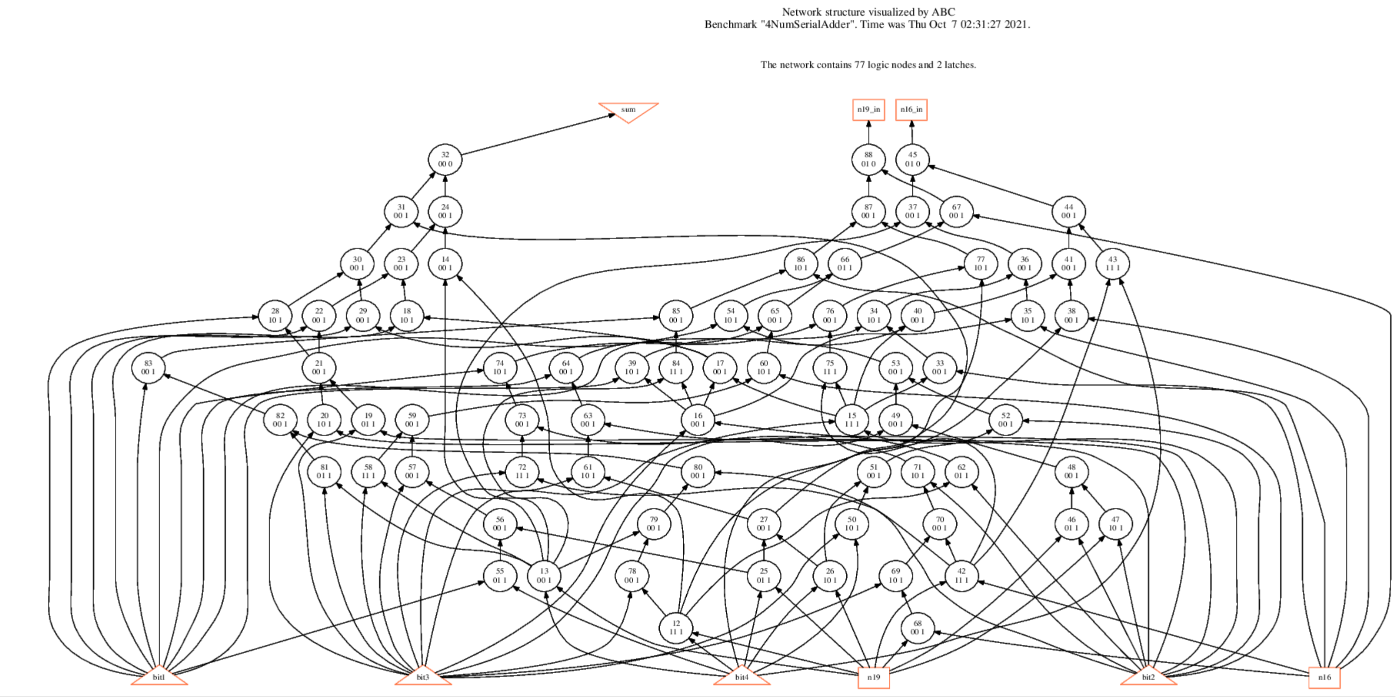
Terminal Screenshot:



```
abc 01> read lsv_fall_2021/pa1/adder_4b.blif
Hierarchy reader flattened 1 instances of logic boxes and left 0 black boxes.
abc 02> strash
abc 03> logic
abc 04> print_stats
4NumSerialAdder               : i/o =    4/    1  lat =    2  nd =    77  edge =    154  cube =    77  lev = 10
abc 04> show
abc 04> Warning: Missing charsets in String to FontSet conversion

abc 04> sop
abc 04> print_stats
4NumSerialAdder               : i/o =    4/    1  lat =    2  nd =    77  edge =    154  cube =    77  lev = 10
abc 04> show
abc 04> Warning: Missing charsets in String to FontSet conversion
```

After either `strash`, `logic`, `sop` command


Network structure visualized by ABC
Benchmark "4NumSerialAdder". Time was Thu Oct 7 02:31:27 2021.
The network contains 77 logic nodes and 2 latches.
sum
n19_in
n16_in
bit1
bit3
bit4
n19
bit2
n16