Final Project Report

Anthony Broderick

Mikel Corsetti

CIS 4930: VLSI Testing

Table of Contents

Abstract

The goal of this project was to create a circuit testing application in python that reads from given ckt or txt files and allows the user to do multiple things to see how to test this circuit; these processes include fault collapsing, listing the fault classes, simulating the circuit, and generating test cases for faults using the PODEM algorithm. The next few sections will go into more detail on each of these functions and how we designed them, as well as what issues we had deriving them. After those sections we will show some screenshots of our program operating, and the outputs associated with those tests. To begin the program, run main.py.

Input Net-List

The first function we created for this program was the net-list parsing algorithm. This takes the files provided by the user in a specific predetermined format and creates a list of data structures to be used by the rest of the program. The main structures formed in the calling of this function are the lists of primary input and outputs, gate list, and full fault list for all inputs. it starts this by reading in all the primary input wires and placing them in their list as well as the primary outputs, then it starts reading in the gates provided in the file and created a new structure for each one; filling in information like the input and output lines, if it has a primary input or not, and adding the faults associated with that gates input to the fault universe. After this is all done, we will have a list of gates, primary inputs/outputs, and a fault universe to call upon for the subsequent function calls. This also creates fan-outs following the same design as the gate structures, just with the outputs being the same value as the inputs. Also, every wire made for these structures is initialized to X value to allow for later functions to operate smoothly.

We had issues with this function primarily due to the somewhat less strict formatting standards we were expected to read from. This led to us having to account for extra spaces, naming schemes, and different delimiting values; however, we did eventually get it to parse any test case we were given.

Fault Collapsing

For the fault collapse function was one of the more complicated functions for us to understand and implement to our program. This function works by taking the created fault universe in the input call and applying a few evaluations to each. This function starts by reading each gate that is associated with a primary input and proceeds to apply fault equivalence and then fault dominance to its listed faults to decide which to leave and remove from the fault universe. we use the basic Boolean equations of:

$$\boldsymbol{Fault\ Equivalence}$$

$$input\ s-a-c = output\ s-a-(c \oplus i)$$

$$we\ keep\ one\ input\ and\ remove\ the\ rest$$

$$\boldsymbol{Fault\ Dominance}$$

$$output\ s-a-(c \odot i)\ dominates\ input\ s-a-(c')'s$$

$$remove\ output\ and\ keep\ inputs$$

We had some issues understanding how to get to what needed to be included and /or removed from the fault universe, and what directive we needed to undertake to decide whether to use outputs or inputs in our fault universe. This function also runs for our fanouts as we created them as gates to allow for modular use of these formulae.

Simulation

The simulate function was one of the easiest ones to come up with and get working. We started on the PODEM function before this one and created an internal function called evaluate_gate(). This was then leveraged to be used to just be recursively called for every gate in the net-list and applies the correct values to every wire on the gate structure before moving to the next. This function does take a user input of a value for each primary input and then a list of faults on any wires in the circuit. Then, it runs the simulator for every individual fault showing each wire in the circuit's value as well as the outputs on the circuit.

Generating Tests (PODEM)

The PODEM function was one of our most difficult functions to implement. We began by piecing together all the bits of pseudocode from the class lectures on the functions PODEM, Objective, and Backtrace. Our initial issues came from the program only propagating the fault through to the output with further issues in that process. The gates would propagate the D through even if the other input(s) were a don't care, it only set a value (1/0/X) for the PI feeding into the same gate as the fault line while leaving every other PI as a don't care 'X', and the propagation of D wasn't always correct.

We then moved on to a code base we found online and tried adopting its functions to the way we did our gates, fanouts, and wire/value assignments. Again, this attempt proved to have even more issues. A similar issue where the PIs not inputting into the same gate as the fault line were all assigned '0' no matter what. Furthermore, it didn't get the one PI it did assign correct in the cases with AND gates, assigning the non-fault value to controlling values, preventing the fault value from propagating. The code did not use PODEM correctly either, relying on a test vector generator function to incorrectly assign the PIs, flipping their values from '0' to '1' to '0' for no reason. After trying to make sense of and correct this code, we ultimately decided it would be easier to start from scratch.

Finally, using the example we had for PODEM in lecture slides 6 for SSFs, we went through each step and wrote down what was happening. This helped us gain a better understanding of what each functions goal was, whether it was assigning any values within the function, and what it was supposed to take as inputs and return. With this done, we wrote each function in plain English so we can decide on code for what it needs to do. We then adapted the English description and steps line by line into the appropriate code for our project. After doing so, our next step was implementing print statements to debug the program; We found areas where we left portions with their English/pseudocode instead of Python code, improper evaluation of gates based on don't care ('X') lines, fanout handling, what backtrace was returning, and a few other minor errors.

Our biggest change from the example we followed in the lectures was how we chose which gate in the D-frontier to use. We initially had it choose the first gate in the D-frontier each time but ran into issues regarding conflicts and reversing decisions. With choosing the first gate of the D-frontier, if the program had the final gate as the chosen gate and there was a conflict in the gate prior to it, it would reverse the decision of the final gates input to one PI instead of trying multiple. This was most noticeable in cases where the inputs to a gate were D and D'. We then tried always selecting the last gate, but finally made it so that it would iterate through the gates in D-frontier whenever Backtrace would get to a gate where none of the inputs are 'X'. If it checked all gates in the D-frontier with no success, it could safely be determined that no test vector exists.

Our final code had PODEM run recursively with the following function goals. Objective() to set the initial fault line on first run, then find necessary input to propagate fault on every subsequent run. Backtrace(k, vk) (with k and vk being the wire and value to set the wire to from Objective() respectfully) to backtrace to a PI without changing any wire values. Imply(j, vj) (with j and vj being the PI line and complemented value needed for propagation from Backtrace(k, vk) respectfully) to keep evaluating gates until it is no longer changing outputs. Check to see if PODEM() returns 'SUCCESS', followed by inverting

the value used for imply if PODEM returns any conflict. If PODEM() does not return success again, it will reverse the chosen PI to ‘X’ and return ‘FAILURE’, ending the function and determining that the fault is undetectable. Instead, if the PODEM has the error propagated to a PO, it returns ‘SUCCESS’ and moves on to the next fault value if any.

We also originally had the PODEM only run for a user defined PI fault line and fault value, but then extended this to fanouts of PIs, and then any wire of the circuit after minor coding changes. With the code working for any wire in the circuit, we made the PODEM function run for every s-a-fault for every wire in the circuit and made the program print out the test vector for each one.

Conclusions

After all our functions were operating as planned we came to a few conclusions. One, implementing the theory we have been studying in class to an application is difficult. Two, these types of programs are not the most complex theory wise at their core, but to make them useful for real world applications there would need to be so many more variables and functions thrown in to allow for more rigorous testing of silicon designs. Third, building on the thought of number two, these programs would take forever to do these types of tests on actual production circuit designs due to the large size of the designs; there would need to be massive optimization of these applications to allow for as much to be tested in as short a time as possible. This paper will end with some images of the program being run and the inputs/outputs we use/generate during use.

Netlist: benchmarks/t4_3.ckt

[0] Read the input net-list

→ benchmarks/t4_3.ckt (or wherever the file is located on your computer)

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 0
Enter the path to the net-list file: benchmarks/t4_3.ckt
Primary Inputs: ['1gat', '2gat', '3gat', '4gat']
Primary Outputs: ['9gat']
Fanouts: {'1gat': 2}
Wire Values: {'5gat': 'X', '2gat': 'X', '3gat': 'X', '6gat': 'X', '1gat': 'X', '7gat': 'X', '8gat': 'X', '4gat':
'X', '9gat': 'X', '1gat_fan0': 'X', '1gat_fan1': 'X'}
Gates:
  G0: ['5gat'] = or(2gat, 3gat)   c=1, inv=0
  G1: ['6gat'] = not(1gat_fan0)   c=0, inv=1
  G2: ['7gat'] = and(1gat_fan1, 5gat)   c=0, inv=0
  G3: ['8gat'] = and(6gat, 4gat)   c=0, inv=0
  G4: ['9gat'] = or(7gat, 8gat)   c=1, inv=0
  Fanout_1gat: ['1gat_fan0', '1gat_fan1']

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t4_3.ckt

[2] List fault classes

Output before running [1] Perform fault collapsing

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 2
Fault List:
1gat: s-a-0
1gat: s-a-1
1gat_fan0: s-a-0
1gat_fan0: s-a-1
1gat_fan1: s-a-0
1gat_fan1: s-a-1
2gat: s-a-0
2gat: s-a-1
3gat: s-a-0
3gat: s-a-1
4gat: s-a-0
4gat: s-a-1
5gat: s-a-0
5gat: s-a-1
6gat: s-a-0
6gat: s-a-1
7gat: s-a-0
7gat: s-a-1
8gat: s-a-0
8gat: s-a-1
9gat: s-a-0
9gat: s-a-1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t4_3.ckt

[1] Perform fault collapsing

Followed by [2] List fault classes

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 1
Faults collapsed.

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

```
Select an option: 2
Fault List:
1gat: s-a-0
1gat: s-a-1
1gat_fan1: s-a-0
1gat_fan1: s-a-1
2gat: s-a-0
2gat: s-a-1
3gat: s-a-0
4gat: s-a-0
4gat: s-a-1
['6gat']: s-a-0
['6gat']: s-a-1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t4_3.ckt

[3] Simulate

Test vector: 10X1

Faults injected:

(None, just hit enter)

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 3
Primary inputs:
  Enter value for 1gat (0/1/X): 1
  Enter value for 2gat (0/1/X): 0
  Enter value for 3gat (0/1/X): X
  Enter value for 4gat (0/1/X): 1

Available wires in circuit (can inject faults on any):
  1gat, 1gat_fan0, 1gat_fan1, 2gat, 3gat, 4gat, 5gat, 6gat, 7gat, 8gat, 9gat

Enter faults to inject (comma-separated, e.g. 'a: s-a-0, b: s-a-0'), or leave blank:

9gat: X

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t4_3.ckt

[3] Simulate

Test vector: 1010

Faults injected:

1gat: s-a-1, 1gat_fan0: s-a-0, 2gat: s-a-1, 2gat: s-a-0, 4gat: s-a-0, 4gat: s-a-1, 5gat: s-a-1, 6gat: s-a-0

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 3
Primary inputs:
  Enter value for 1gat (0/1/X): 1
  Enter value for 2gat (0/1/X): 0
  Enter value for 3gat (0/1/X): 1
  Enter value for 4gat (0/1/X): 0

Available wires in circuit (can inject faults on any):
  1gat, 1gat_fan0, 1gat_fan1, 2gat, 3gat, 4gat, 5gat, 6gat, 7gat, 8gat, 9gat

Enter faults to inject (comma-separated, e.g. 'a: s-a-0, b: s-a-0'), or leave blank: 1gat: s-a-1, 1gat_fan0: s-a-0, 2gat: s-a-1, 2ga
t: s-a-0, 4gat: s-a-0, 4gat: s-a-1, 5gat: s-a-1, 6gat: s-a-0

fault line  fault value  9gat
------------------------------
1gat_fan0   s-a-0        1
2gat        s-a-0        1
4gat        s-a-0        1
6gat        s-a-0        1
1gat        s-a-1        1
2gat        s-a-1        1
4gat        s-a-1        1
5gat        s-a-1        D'

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit
```

Netlist: benchmarks/t4_3.ckt

[4] Generate tests (PODEM)

No input needed

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 4
Test Vector Format: 1gat 2gat 3gat 4gat
1gat s-a-0: 1001
1gat s-a-1: 0001
1gat_fan0 s-a-0: 1001
1gat_fan0 s-a-1: 0XX1
1gat_fan1 s-a-0: 11XX
1gat_fan1 s-a-1: 01X0
2gat s-a-0: 110X
2gat s-a-1: 100X
3gat s-a-0: 101X
3gat s-a-1: 100X
4gat s-a-0: 0XX1
4gat s-a-1: 0XX0
5gat s-a-0: 11XX
5gat s-a-1: 10XX
6gat s-a-0: 0XX1
6gat s-a-1: 1001
7gat s-a-0: X1X0
7gat s-a-1: X0X0
8gat s-a-0: X001
8gat s-a-1: X000
9gat s-a-0: XXX1
9gat s-a-1: XXX0

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t4_21.ckt

[0] Read the input net-list

→ benchmarks/t4_21.ckt (or wherever the file is located on your computer)

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 0
Enter the path to the net-list file: benchmarks/t4_21.ckt
Primary Inputs: ['1gat', '2gat', '3gat', '4gat', '5gat']
Primary Outputs: ['9gat']
Fanouts: {'3gat': 2}
Wire Values: {'10gat': 'X', '1gat': 'X', '2gat': 'X', '6gat': 'X', '3gat': 'X', '7gat': 'X', '4gat': 'X', '8gat': 'X', '5gat': 'X',
'9gat': 'X', '3gat_fan0': 'X', '3gat_fan1': 'X'}
Gates:
  G0: ['10gat'] = and(1gat, 2gat)   c=0, inv=0
  G1: ['6gat'] = nor(3gat_fan0, 10gat)   c=1, inv=1
  G2: ['7gat'] = and(3gat_fan1, 4gat)   c=0, inv=0
  G3: ['8gat'] = or(5gat, 7gat)   c=1, inv=0
  G4: ['9gat'] = or(6gat, 8gat)   c=1, inv=0
  Fanout_3gat: ['3gat_fan0', '3gat_fan1']

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t4_21.ckt

[2] List fault classes

Output before running [1] Perform fault collapsing

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 2
Fault List:
10gat: s-a-0
10gat: s-a-1
1gat: s-a-0
1gat: s-a-1
2gat: s-a-0
2gat: s-a-1
3gat: s-a-0
3gat: s-a-1
3gat_fan0: s-a-0
3gat_fan0: s-a-1
3gat_fan1: s-a-0
3gat_fan1: s-a-1
4gat: s-a-0
4gat: s-a-1
5gat: s-a-0
5gat: s-a-1
6gat: s-a-0
6gat: s-a-1
7gat: s-a-0
7gat: s-a-1
8gat: s-a-0
8gat: s-a-1
9gat: s-a-0
9gat: s-a-1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: ∏
```

Netlist: benchmarks/t4_21.ckt

[1] Perform fault collapsing

Followed by [2] List fault classes

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 1
Faults collapsed.

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 2
Fault List:
1gat: s-a-0
1gat: s-a-1
2gat: s-a-1
3gat: s-a-0
3gat: s-a-1
3gat_fan0: s-a-0
3gat_fan0: s-a-1
3gat_fan1: s-a-0
3gat_fan1: s-a-1
4gat: s-a-1
5gat: s-a-0
5gat: s-a-1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t4_21.ckt

[3] Simulate

Test vector: 10101

Faults injected:

(None, just hit enter)

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 3
Primary inputs:
  Enter value for 1gat (0/1/X): 1
  Enter value for 2gat (0/1/X): 0
  Enter value for 3gat (0/1/X): 1
  Enter value for 4gat (0/1/X): 0
  Enter value for 5gat (0/1/X): 1

Available wires in circuit (can inject faults on any):
  10gat, 1gat, 2gat, 3gat, 3gat_fan0, 3gat_fan1, 4gat, 5gat, 6gat, 7gat, 8gat, 9gat

Enter faults to inject (comma-separated, e.g. 'a: s-a-0, b: s-a-0'), or leave blank:

9gat: 1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t4_21.ckt

[3] Simulate

Test vector: 10100

Faults injected:

1gat: s-a-1, 3gat_fan0: s-a-0, 2gat: s-a-1, 2gat: s-a-0, 4gat: s-a-0, 4gat: s-a-1, 5gat: s-a-1, 6gat: s-a-0

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 3
Primary inputs:
  Enter value for 1gat (0/1/X): 1
  Enter value for 2gat (0/1/X): 0
  Enter value for 3gat (0/1/X): 1
  Enter value for 4gat (0/1/X): 0
  Enter value for 5gat (0/1/X): 0

Available wires in circuit (can inject faults on any):
  10gat, 1gat, 2gat, 3gat, 3gat_fan0, 3gat_fan1, 4gat, 5gat, 6gat, 7gat, 8gat, 9gat

Enter faults to inject (comma-separated, e.g. 'a: s-a-0, b: s-a-0'), or leave blank: 1gat: s-a-1, 3gat_fan0: s-a-0, 2gat: s-a-1, 2ga
t: s-a-0, 4gat: s-a-0, 4gat: s-a-1, 5gat: s-a-1, 6gat: s-a-0

fault line  fault value  9gat
------------------------------
3gat_fan0   s-a-0        D'
2gat        s-a-0        0
4gat        s-a-0        0
6gat        s-a-0        D
1gat        s-a-1        0
2gat        s-a-1        0
4gat        s-a-1        D'
5gat        s-a-1        D'

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t4_21.ckt

[4] Generate tests (PODEM)

No input needed

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 4
Test Vector Format: 1gat 2gat 3gat 4gat 5gat
10gat s-a-0: 1X0X0
10gat s-a-1: 0X0X0
1gat s-a-0: 110X0
1gat s-a-1: 010X0
2gat s-a-0: 110X0
2gat s-a-1: 100X0
3gat s-a-0: 11110
3gat s-a-1: 11010
3gat_fan0 s-a-0: 0X100
3gat_fan0 s-a-1: 0X0X0
3gat_fan1 s-a-0: XX110
3gat_fan1 s-a-1: 11010
4gat s-a-0: XX110
4gat s-a-1: XX100
5gat s-a-0: 0X101
5gat s-a-1: 0X100
6gat s-a-0: 1XX00
6gat s-a-1: 0XX00
7gat s-a-0: 0X110
7gat s-a-1: 0X100
8gat s-a-0: 0X1X1
8gat s-a-1: 0X1X0
9gat s-a-0: XXXX1
9gat s-a-1: XXXX0

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t5_10.ckt

[0] Read the input net-list

→ benchmarks/t5_10.ckt (or wherever the file is located on your computer)

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 0
Enter the path to the net-list file: benchmarks/t5_10.ckt
Primary Inputs: ['agat', 'bgat', 'cgat', 'dgat', 'egat']
Primary Outputs: ['mgat']
Fanouts: {'cgat': 2}
Wire Values: {'fgat': 'X', 'agat': 'X', 'bgat': 'X', 'jgat': 'X', 'cgat': 'X', 'igat': 'X', 'dgat': 'X', 'kgat':
'X', 'egat': 'X', 'mgat': 'X', 'cgat_fan0': 'X', 'cgat_fan1': 'X'}
Gates:
  G0: ['fgat'] = and(agat, bgat)   c=0, inv=0
  G1: ['jgat'] = nor(fgat, cgat_fan0)   c=1, inv=1
  G2: ['igat'] = and(cgat_fan1, dgat)   c=0, inv=0
  G3: ['kgat'] = or(igat, egat)   c=1, inv=0
  G4: ['mgat'] = or(jgat, kgat)   c=1, inv=0
  Fanout_cgat: ['cgat_fan0', 'cgat_fan1']

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t5_10.ckt

[2] List fault classes

Output before running [1] Perform fault collapsing

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 2
Fault List:
agat: s-a-0
agat: s-a-1
bgat: s-a-0
bgat: s-a-1
cgat: s-a-0
cgat: s-a-1
cgat_fan0: s-a-0
cgat_fan0: s-a-1
cgat_fan1: s-a-0
cgat_fan1: s-a-1
dgat: s-a-0
dgat: s-a-1
egat: s-a-0
egat: s-a-1
fgat: s-a-0
fgat: s-a-1
igat: s-a-0
igat: s-a-1
jgat: s-a-0
jgat: s-a-1
kgat: s-a-0
kgat: s-a-1
mgat: s-a-0
mgat: s-a-1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t5_10.ckt

[1] Perform fault collapsing

Followed by [2] List fault classes

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 1
Faults collapsed.

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 2
Fault List:
agat: s-a-0
agat: s-a-1
bgat: s-a-1
cgat: s-a-0
cgat: s-a-1
cgat_fan0: s-a-0
cgat_fan0: s-a-1
cgat_fan1: s-a-0
cgat_fan1: s-a-1
dgat: s-a-1
egat: s-a-0
egat: s-a-1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t5_10.ckt

[3] Simulate

Test vector: 110X0

Faults injected:

agat: s-a-0, agat: s-a-1, bgat: s-a-0, bgat:s-a-1, cgat_fan0: s-a-0, cgat_fan0: s-a-1, cgat_fan1: s-a-0

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 3
Primary inputs:
  Enter value for agat (0/1/X): 1
  Enter value for bgat (0/1/X): 1
  Enter value for cgat (0/1/X): 0
  Enter value for dgat (0/1/X): X
  Enter value for egat (0/1/X): 0

Available wires in circuit (can inject faults on any):
  agat, bgat, cgat, cgat_fan0, cgat_fan1, dgat, egat, fgat, igat, jgat, kgat, mgat

Enter faults to inject (comma-separated, e.g. 'a: s-a-0, b: s-a-0'), or leave blank: agat: s-a-0, agat: s-a-1, bgat: s-a-0, bgat:s-a
-1, cgat_fan0: s-a-0, cgat_fan0: s-a-1, cgat_fan1: s-a-0

fault line  fault value  mgat
--------------------------------
agat        s-a-0        D'
bgat        s-a-0        D'
cgat_fan0   s-a-0        0
cgat_fan1   s-a-0        D
agat        s-a-1        0
bgat        s-a-1        0
cgat_fan0   s-a-1        0

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t5_10.ckt

[3] Simulate

Test vector: 110X0

Faults injected:

None

```
Select an option: 3
Primary inputs:
  Enter value for agat (0/1/X): 1
  Enter value for bgat (0/1/X): 1
  Enter value for cgat (0/1/X): 0
  Enter value for dgat (0/1/X): X
  Enter value for egat (0/1/X): 0

Available wires in circuit (can inject faults on any):
  agat, bgat, cgat, cgat_fan0, cgat_fan1, dgat, egat, fgat, igat, jgat, kgat, mgat

Enter faults to inject (comma-separated, e.g. 'a: s-a-0, b: s-a-0'), or leave blank:

mgat: 0

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t5_10.ckt

[4] Generate tests (PODEM)

No input needed

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 4
Test Vector Format: agat bgat cgat dgat egat
agat s-a-0: 110X0
agat s-a-1: 010X0
bgat s-a-0: 110X0
bgat s-a-1: 100X0
cgat s-a-0: 11110
cgat s-a-1: 11010
cgat_fan0 s-a-0: 0X100
cgat_fan0 s-a-1: 0X0X0
cgat_fan1 s-a-0: XX110
cgat_fan1 s-a-1: 11010
dgat s-a-0: XX110
dgat s-a-1: XX100
egat s-a-0: 11001
egat s-a-1: 11000
fgat s-a-0: 1X0X0
fgat s-a-1: 0X0X0
igat s-a-0: 11010
igat s-a-1: 11000
jgat s-a-0: XX100
jgat s-a-1: XX0X0
kgat s-a-0: 110X1
kgat s-a-1: 110X0
mgat s-a-0: XXXX1
mgat s-a-1: XXXX0

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit
```

Netlist: benchmarks/t5_26a.ckt

[0] Read the input net-list

→ benchmarks/t5_26a.ckt (or wherever the file is located on your computer)

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 0
Enter the path to the net-list file: benchmarks/t5_26a.ckt
Primary Inputs: ['X', 'Y', 'CI']
Primary Outputs: ['S', 'CO']
Fanouts: {'X': 2, 'L': 3, 'Y': 2, 'CI': 2, 'T': 3, 'N': 2}
Wire Values: {'L': 'X', 'X': 'X', 'Y': 'X', 'Q': 'X', 'R': 'X', 'N': 'X', 'T': 'X', 'CI': 'X', 'U': 'X', 'V': 'X', 'S': 'X', 'CO': '
X', 'X_fan0': 'X', 'X_fan1': 'X', 'L_fan0': 'X', 'L_fan1': 'X', 'L_fan2': 'X', 'Y_fan0': 'X', 'Y_fan1': 'X', 'CI_fan0': 'X', 'CI_fan
1': 'X', 'T_fan0': 'X', 'T_fan1': 'X', 'T_fan2': 'X', 'N_fan0': 'X', 'N_fan1': 'X'}
Gates:
  G0: ['L'] = nand(X_fan0, Y_fan0)   c=0, inv=1
  G1: ['Q'] = nand(X_fan1, L_fan0)   c=0, inv=1
  G2: ['R'] = nand(L_fan1, Y_fan1)   c=0, inv=1
  G3: ['N'] = nand(Q, R)   c=0, inv=1
  G4: ['T'] = nand(CI_fan0, N_fan0)   c=0, inv=1
  G5: ['U'] = nand(CI_fan1, T_fan0)   c=0, inv=1
  G6: ['V'] = nand(T_fan1, N_fan1)   c=0, inv=1
  G7: ['S'] = nand(U, V)   c=0, inv=1
  G8: ['CO'] = nand(L_fan2, T_fan2)   c=0, inv=1
  Fanout_X: ['X_fan0', 'X_fan1']
  Fanout_L: ['L_fan0', 'L_fan1', 'L_fan2']
  Fanout_Y: ['Y_fan0', 'Y_fan1']
  Fanout_CI: ['CI_fan0', 'CI_fan1']
  Fanout_T: ['T_fan0', 'T_fan1', 'T_fan2']
  Fanout_N: ['N_fan0', 'N_fan1']

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t5_26a.ckt

[2] List fault classes

Output before running [1] Perform fault collapsing

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 2
Fault List:
CI: s-a-0
CI: s-a-1
CI_fan0: s-a-0
CI_fan0: s-a-1
CI_fan1: s-a-0
CI_fan1: s-a-1
CO: s-a-0
CO: s-a-1
L: s-a-0
L: s-a-1
L_fan0: s-a-0
L_fan0: s-a-1
L_fan1: s-a-0
L_fan1: s-a-1
L_fan2: s-a-0
L_fan2: s-a-1
N: s-a-0
N: s-a-1
N_fan0: s-a-0
N_fan0: s-a-1
N_fan1: s-a-0
N_fan1: s-a-1
Q: s-a-0
Q: s-a-1
R: s-a-0
R: s-a-1
S: s-a-0
S: s-a-1
T: s-a-0
T: s-a-1
T_fan0: s-a-0
T_fan0: s-a-1
T_fan1: s-a-0
T_fan1: s-a-1
T_fan2: s-a-0
T_fan2: s-a-1
U: s-a-0

U: s-a-1
V: s-a-0
V: s-a-1
X: s-a-0
X: s-a-1
X_fan0: s-a-0
X_fan0: s-a-1
X_fan1: s-a-0
X_fan1: s-a-1
Y: s-a-0
Y: s-a-1
Y_fan0: s-a-0
Y_fan0: s-a-1
Y_fan1: s-a-0
Y_fan1: s-a-1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t5_26a.ckt

[1] Perform fault collapsing

Followed by [2] List fault classes

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 1
Faults collapsed.

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 2
Fault List:
CI: s-a-0
CI: s-a-1
CI_fan0: s-a-0
CI_fan0: s-a-1
CI_fan1: s-a-0
CI_fan1: s-a-1
X: s-a-0
X: s-a-1
X_fan0: s-a-0
X_fan0: s-a-1
X_fan1: s-a-0
X_fan1: s-a-1
Y: s-a-0
Y: s-a-1
Y_fan0: s-a-1
Y_fan1: s-a-0
Y_fan1: s-a-1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t5_26a.ckt

[3] Simulate

Test vector: 110

Faults injected:

(None, just hit enter)

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 3
Primary inputs:
  Enter value for X (0/1/X): 1
  Enter value for Y (0/1/X): 1
  Enter value for CI (0/1/X): 0

Available wires in circuit (can inject faults on any):
  CI, CI_fan0, CI_fan1, CO, L, L_fan0, L_fan1, L_fan2, N, N_fan0, N_fan1, Q, R, S, T, T_fan0, T_fan1, T_fan2, U, V, X, X_fan0, X_fan
1, Y, Y_fan0, Y_fan1

Enter faults to inject (comma-separated, e.g. 'a: s-a-0, b: s-a-0'), or leave blank:

S: 0
CO: 1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t5_26a.ckt

[3] Simulate

Test vector: 110

Faults injected:

X: s-a-1, X: s-a-0, Y: s-a-1, Y: s-a-0, CI: s-a-0, CI_fan0: s-a-1, CI_fan1: s-a-1, U: s-a-0, T_fan1: s-a-0

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 3
Primary inputs:
  Enter value for X (0/1/X): 1
  Enter value for Y (0/1/X): 1
  Enter value for CI (0/1/X): 0

Available wires in circuit (can inject faults on any):
  CI, CI_fan0, CI_fan1, CO, L, L_fan0, L_fan1, L_fan2, N, N_fan0, N_fan1, Q, R, S, T, T_fan0, T_fan1, T_fan2, U, V, X, X_fan0, X_fan
1, Y, Y_fan0, Y_fan1

Enter faults to inject (comma-separated, e.g. 'a: s-a-0, b: s-a-0'), or leave blank: X: s-a-1, X: s-a-0, Y: s-a-1, Y: s-a-0, CI: s-a
-0, CI_fan0: s-a-1, CI_fan1: s-a-1, U: s-a-0, T_fan1: s-a-0

fault line  fault value  S   CO
---------------------------------
X           s-a-0        D'  D
Y           s-a-0        D'  D
CI          s-a-0        0   1
U           s-a-0        D'  1
T_fan1      s-a-0        0   1
X           s-a-1        0   1
Y           s-a-1        0   1
CI_fan0     s-a-1        0   1
CI_fan1     s-a-1        D'  1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t5_26a.ckt

[4] Generate tests (PODEM)

No input needed

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 4
Test Vector Format: X Y CI
CI s-a-0: 001
CI s-a-1: 000
CI_fan0 s-a-0: 011
CI_fan0 s-a-1: 010
CI_fan1 s-a-0: 001
CI_fan1 s-a-1: 000
CO s-a-0: X1X
CO s-a-1: X0X
L s-a-0: 110
L s-a-1: 100
L_fan0 s-a-0: 110
L_fan0 s-a-1: 100
L_fan1 s-a-0: 010
L_fan1 s-a-1: 010
L_fan2 s-a-0: 11X
L_fan2 s-a-1: 100
N s-a-0: X10
N s-a-1: X00
N_fan0 s-a-0: 011
N_fan0 s-a-1: X01
N_fan1 s-a-0: 110
N_fan1 s-a-1: 100
Q s-a-0: 110
Q s-a-1: X00
R s-a-0: 010
R s-a-1: 000
S s-a-0: X1X
S s-a-1: X0X
T s-a-0: 01X
T s-a-1: X0X
T_fan0 s-a-0: 111
T_fan0 s-a-1: 101
T_fan1 s-a-0: 010
T_fan1 s-a-1: 100
T_fan2 s-a-0: 01X
T_fan2 s-a-1: X0X
U s-a-0: 11X


U s-a-1: 101
V s-a-0: 110
V s-a-1: 100
X s-a-0: 100
X s-a-1: 000
X_fan0 s-a-0: 110
X_fan0 s-a-1: 010
X_fan1 s-a-0: 100
X_fan1 s-a-1: 000
Y s-a-0: 010
Y s-a-1: 000
Y_fan0 s-a-0: 110
Y_fan0 s-a-1: 100
Y_fan1 s-a-0: 010
Y_fan1 s-a-1: 000

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t6_24_v1.ckt.txt

[0] Read the input net-list

→ benchmarks/t6_24_v1.ckt.txt (or wherever the file is located on your computer)

```
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 0
Enter the path to the net-list file: benchmarks/t6_24_v1.ckt.txt
Primary Inputs: ['1gat', '2gat', '3gat', '4gat', '5gat', '6gat']
Primary Outputs: ['16gat']
Fanouts: {'5gat': 3, '17gat': 3, '4gat': 3, '2gat': 2, '1gat': 2, '6gat': 3, '3gat': 2}
Wire Values: {'13gat': 'X', '5gat': 'X', '17gat': 'X', '8gat': 'X', '12gat': 'X', '2gat': 'X', '11gat': 'X', '4gat': 'X', '7gat': 'X
', '10gat': 'X', '1gat': 'X', '18gat': 'X', '3gat': 'X', '9gat': 'X', '6gat': 'X', '14gat': 'X', '15gat': 'X', '19gat': 'X', '20gat'
: 'X', '21gat': 'X', '22gat': 'X', '16gat': 'X', '5gat_fan0': 'X', '5gat_fan1': 'X', '5gat_fan2': 'X', '17gat_fan0': 'X', '17gat_fan
1': 'X', '17gat_fan2': 'X', '4gat_fan0': 'X', '4gat_fan1': 'X', '4gat_fan2': 'X', '2gat_fan0': 'X', '2gat_fan1': 'X', '1gat_fan0': '
X', '1gat_fan1': 'X', '6gat_fan0': 'X', '6gat_fan1': 'X', '6gat_fan2': 'X', '3gat_fan0': 'X', '3gat_fan1': 'X'}
Gates:
  G0: ['13gat'] = nand(5gat_fan0, 17gat_fan0)   c=0, inv=1
  G1: ['8gat'] = nand(5gat_fan1, 5gat)   c=0, inv=1
  G2: ['12gat'] = nand(2gat_fan0, 8gat)   c=0, inv=1
  G3: ['11gat'] = nand(4gat_fan0, 17gat_fan1)   c=0, inv=1
  G4: ['7gat'] = nand(4gat_fan1, 4gat)   c=0, inv=1
  G5: ['10gat'] = nand(1gat_fan0, 7gat)   c=0, inv=1
  G6: ['18gat'] = and(2gat_fan1, 3gat_fan0)   c=0, inv=0
  G7: ['17gat'] = nand(1gat_fan1, 18gat)   c=0, inv=1
  G8: ['9gat'] = nand(6gat_fan0, 6gat)   c=0, inv=1
  G9: ['14gat'] = nand(3gat_fan1, 9gat)   c=0, inv=1
  G10: ['15gat'] = nand(6gat_fan1, 17gat_fan2)   c=0, inv=1
  G11: ['19gat'] = and(10gat, 11gat)   c=0, inv=0
  G12: ['20gat'] = and(12gat, 13gat)   c=0, inv=0
  G13: ['21gat'] = and(14gat, 15gat)   c=0, inv=0
  G14: ['22gat'] = and(19gat, 20gat)   c=0, inv=0
  G15: ['16gat'] = nand(21gat, 22gat)   c=0, inv=1
  Fanout_5gat: ['5gat_fan0', '5gat_fan1', '5gat_fan2']
  Fanout_17gat: ['17gat_fan0', '17gat_fan1', '17gat_fan2']
  Fanout_4gat: ['4gat_fan0', '4gat_fan1', '4gat_fan2']
  Fanout_2gat: ['2gat_fan0', '2gat_fan1']
  Fanout_1gat: ['1gat_fan0', '1gat_fan1']
  Fanout_6gat: ['6gat_fan0', '6gat_fan1', '6gat_fan2']
  Fanout_3gat: ['3gat_fan0', '3gat_fan1']

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t6_24_v1.ckt.txt

[2] List fault classes

Output before running [1] Perform fault collapsing

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 2
Fault List:
10gat: s-a-0
10gat: s-a-1
11gat: s-a-0
11gat: s-a-1
12gat: s-a-0
12gat: s-a-1
13gat: s-a-0
13gat: s-a-1
14gat: s-a-0
14gat: s-a-1
15gat: s-a-0
15gat: s-a-1
16gat: s-a-0
16gat: s-a-1
17gat: s-a-0
17gat: s-a-1
17gat_fan0: s-a-0
17gat_fan0: s-a-1
17gat_fan1: s-a-0
17gat_fan1: s-a-1
17gat_fan2: s-a-0
17gat_fan2: s-a-1
18gat: s-a-0
18gat: s-a-1
19gat: s-a-0
19gat: s-a-1
1gat: s-a-0
1gat: s-a-1
1gat_fan0: s-a-0
1gat_fan0: s-a-1
1gat_fan1: s-a-0
1gat_fan1: s-a-1
20gat: s-a-0
20gat: s-a-1
21gat: s-a-0
21gat: s-a-1
22gat: s-a-0
```

```
22gat: s-a-1
2gat: s-a-0
2gat: s-a-1
2gat_fan0: s-a-0
2gat_fan0: s-a-1
2gat_fan1: s-a-0
2gat_fan1: s-a-1
3gat: s-a-0
3gat: s-a-1
3gat_fan0: s-a-0
3gat_fan0: s-a-1
3gat_fan1: s-a-0
3gat_fan1: s-a-1
4gat: s-a-0
4gat: s-a-1
4gat_fan0: s-a-0
4gat_fan0: s-a-1
4gat_fan1: s-a-0
4gat_fan1: s-a-1
4gat_fan2: s-a-0
4gat_fan2: s-a-1
5gat: s-a-0
5gat: s-a-1
5gat_fan0: s-a-0
5gat_fan0: s-a-1
5gat_fan1: s-a-0
5gat_fan1: s-a-1
5gat_fan2: s-a-0
5gat_fan2: s-a-1
6gat: s-a-0
6gat: s-a-1
6gat_fan0: s-a-0
6gat_fan0: s-a-1
6gat_fan1: s-a-0
6gat_fan1: s-a-1
6gat_fan2: s-a-0
6gat_fan2: s-a-1
7gat: s-a-0
7gat: s-a-1
8gat: s-a-0
8gat: s-a-1
9gat: s-a-0
9gat: s-a-1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
```

Netlist: benchmarks/t6_24_v1.ckt.txt

[1] Perform fault collapsing

Followed by [2] List fault classes

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 1
Faults collapsed.

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 2
Fault List:
1gat: s-a-0
1gat: s-a-1
1gat_fan0: s-a-0
1gat_fan0: s-a-1
1gat_fan1: s-a-0
1gat_fan1: s-a-1
2gat: s-a-0
2gat: s-a-1
2gat_fan0: s-a-0
2gat_fan0: s-a-1
2gat_fan1: s-a-0
2gat_fan1: s-a-1
3gat: s-a-0
3gat: s-a-1
3gat_fan0: s-a-1
3gat_fan1: s-a-0
3gat_fan1: s-a-1
4gat: s-a-0
4gat: s-a-1
4gat_fan0: s-a-0
4gat_fan0: s-a-1
4gat_fan1: s-a-0
4gat_fan1: s-a-1
5gat: s-a-0
5gat: s-a-1
5gat_fan0: s-a-0
5gat_fan0: s-a-1

5gat_fan1: s-a-0
5gat_fan1: s-a-1
6gat: s-a-0
6gat: s-a-1
6gat_fan0: s-a-0
6gat_fan0: s-a-1
6gat_fan1: s-a-0
6gat_fan1: s-a-1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t6_24_v1.ckt.txt

[3] Simulate

Test vector: 101010

Faults injected:

(None, just hit enter)

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 3
Primary inputs:
  Enter value for 1gat (0/1/X): 1
  Enter value for 2gat (0/1/X): 0
  Enter value for 3gat (0/1/X): 1
  Enter value for 4gat (0/1/X): 0
  Enter value for 5gat (0/1/X): 1
  Enter value for 6gat (0/1/X): 0

Available wires in circuit (can inject faults on any):
  10gat, 11gat, 12gat, 13gat, 14gat, 15gat, 16gat, 17gat, 17gat_fan0, 17gat_fan1, 17gat_fan2, 18gat, 19gat, 1gat, 1gat_fan0, 1gat_fa
n1, 20gat, 21gat, 22gat, 2gat, 2gat_fan0, 2gat_fan1, 3gat, 3gat_fan0, 3gat_fan1, 4gat, 4gat_fan0, 4gat_fan1, 4gat_fan2, 5gat, 5gat_f
an0, 5gat_fan1, 5gat_fan2, 6gat, 6gat_fan0, 6gat_fan1, 6gat_fan2, 7gat, 8gat, 9gat

Enter faults to inject (comma-separated, e.g. 'a: s-a-0, b: s-a-0'), or leave blank:

16gat: 1

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t6_24_v1.ckt.txt

[3] Simulate

Test vector: XXX100

Faults injected:

1gat: s-a-1, 17gat_fan0: s-a-0, 2gat: s-a-1, 2gat: s-a-0, 4gat: s-a-0, 13gat: s-a-1, 5gat: s-a-1, 20gat: s-a-0

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 3
Primary inputs:
  Enter value for 1gat (0/1/X): X
  Enter value for 2gat (0/1/X): X
  Enter value for 3gat (0/1/X): X
  Enter value for 4gat (0/1/X): 1
  Enter value for 5gat (0/1/X): 0
  Enter value for 6gat (0/1/X): 0

Available wires in circuit (can inject faults on any):
  10gat, 11gat, 12gat, 13gat, 14gat, 15gat, 16gat, 17gat, 17gat_fan0, 17gat_fan1, 17gat_fan2, 18gat, 19gat, 1gat, 1gat_fan0, 1gat_fa
n1, 20gat, 21gat, 22gat, 2gat, 2gat_fan0, 2gat_fan1, 3gat, 3gat_fan0, 3gat_fan1, 4gat, 4gat_fan0, 4gat_fan1, 4gat_fan2, 5gat, 5gat_f
an0, 5gat_fan1, 5gat_fan2, 6gat, 6gat_fan0, 6gat_fan1, 6gat_fan2, 7gat, 8gat, 9gat

Enter faults to inject (comma-separated, e.g. 'a: s-a-0, b: s-a-0'), or leave blank: 1gat: s-a-1, 17gat_fan0: s-a-0, 2gat: s-a-1, 2g
at: s-a-0, 4gat: s-a-0, 13gat: s-a-1, 5gat: s-a-1, 20gat: s-a-0

fault line  fault value  16gat
---------------------------------
17gat_fan0  s-a-0        X
2gat        s-a-0        X
4gat        s-a-0        X
20gat       s-a-0        D'
1gat        s-a-1        D
2gat        s-a-1        X
13gat       s-a-1        D
5gat        s-a-1        X

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option:
```

Netlist: benchmarks/t6_24_v1.ckt.txt

[4] Generate tests (PODEM)

No input needed

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 4
Test Vector Format: 1gat 2gat 3gat 4gat 5gat 6gat
10gat s-a-0: 111111
10gat s-a-1: 111011
11gat s-a-0: 111111
11gat s-a-1: X00100
12gat s-a-0: 111111
12gat s-a-1: 111101
13gat s-a-0: 111111
13gat s-a-1: 0X0010
14gat s-a-0: 111111
14gat s-a-1: 111110
15gat s-a-0: 111111
15gat s-a-1: 00000X
16gat s-a-0: XX1XXX
16gat s-a-1: XX0XXX
17gat s-a-0: 001001
17gat s-a-1: 000001
17gat_fan0 s-a-0: 111111
17gat_fan0 s-a-1: 0X0010
17gat_fan1 s-a-0: 111111
17gat_fan1 s-a-1: X00100
17gat_fan2 s-a-0: 111111
17gat_fan2 s-a-1: 000001
18gat s-a-0: 101101
18gat s-a-1: 100101
19gat s-a-0: 111X11
19gat s-a-1: X00X00
1gat s-a-0: 111111
1gat s-a-1: 011111
1gat_fan0 s-a-0: 100000
1gat_fan0 s-a-1: 000000
1gat_fan1 s-a-0: 111111
1gat_fan1 s-a-1: 011011
20gat s-a-0: 1111X1
20gat s-a-1: 0X00X0
21gat s-a-0: 11111X
21gat s-a-1: 00000X
22gat s-a-0: 111XX1
22gat s-a-1: XX0XX0
```

```
2gat s-a-0: Undetectable fault
2gat s-a-1: Undetectable fault
2gat_fan0 s-a-0: 111101
2gat_fan0 s-a-1: 000000
2gat_fan1 s-a-0: 111111
2gat_fan1 s-a-1: 101101
3gat s-a-0: 001000
3gat s-a-1: 000000
3gat_fan0 s-a-0: 111111
3gat_fan0 s-a-1: 110111
3gat_fan1 s-a-0: 111110
3gat_fan1 s-a-1: 000000
4gat s-a-0: 111111
4gat s-a-1: 111011
4gat_fan0 s-a-0: X00100
4gat_fan0 s-a-1: 000000
4gat_fan1 s-a-0: 111111
4gat_fan1 s-a-1: 111111
4gat_fan2 s-a-0: Undetectable fault
4gat_fan2 s-a-1: Undetectable fault
5gat s-a-0: 111111
5gat s-a-1: 111101
5gat_fan0 s-a-0: 0X0010
5gat_fan0 s-a-1: 000000
5gat_fan1 s-a-0: 111111
5gat_fan1 s-a-1: 111111
5gat_fan2 s-a-0: Undetectable fault
5gat_fan2 s-a-1: Undetectable fault
6gat s-a-0: 111111
6gat s-a-1: 111110
6gat_fan0 s-a-0: 111111
6gat_fan0 s-a-1: 111111
6gat_fan1 s-a-0: 001001
6gat_fan1 s-a-1: 000000
6gat_fan2 s-a-0: Undetectable fault
6gat_fan2 s-a-1: Undetectable fault
7gat s-a-0: 111111
7gat s-a-1: 100000
8gat s-a-0: 111111
8gat s-a-1: 111101
9gat s-a-0: 111111
9gat s-a-1: 111110

    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
```

[5] Exit

Ends program gracefully

```
    [0] Read the input net-list
    [1] Perform fault collapsing
    [2] List fault classes
    [3] Simulate
    [4] Generate tests (PODEM)
    [5] Exit

Select an option: 5
Exiting program.
```