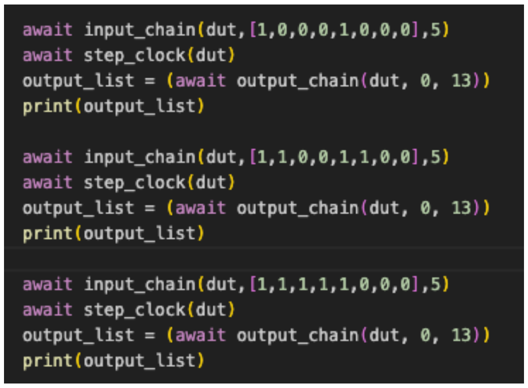
Task #2: Testing the Adder

Test code



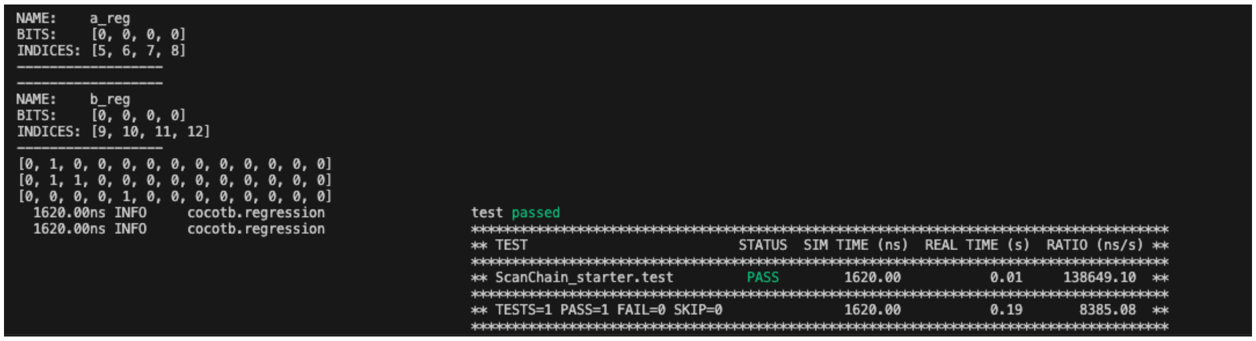
```python
await input_chain(dut,[1,0,0,0,1,0,0,0],5)
await step_clock(dut)
output_list = (await output_chain(dut, 0, 13))
print(output_list)

await input_chain(dut,[1,1,0,0,1,1,0,0],5)
await step_clock(dut)
output_list = (await output_chain(dut, 0, 13))
print(output_list)

await input_chain(dut,[1,1,1,1,1,0,0,0],5)
await step_clock(dut)
output_list = (await output_chain(dut, 0, 13))
print(output_list)
```

Case 1: 4’b0001 + 4’b0001 = 5’b00010

Case 2: 4’b0011 + 4’b0011 = 5’b00110

Case 3: 4’b1111 + 4’b0001 = 5’b10000

Output



```
NAME:    a_reg
BITS:    [0, 0, 0, 0]
INDICES: [5, 6, 7, 8]
------------------
------------------
NAME:    b_reg
BITS:    [0, 0, 0, 0]
INDICES: [9, 10, 11, 12]
------------------
[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
  1620.00ns INFO     cocotb.regression                  test passed
  1620.00ns INFO     cocotb.regression                  ***************************************************************************************
                                                        ** TEST                          STATUS  SIM TIME (ns)  REAL TIME (s)  RATIO (ns/s) **
                                                        ***************************************************************************************
                                                        ** ScanChain_starter.test         PASS        1620.00           0.01     138649.10  **
                                                        ***************************************************************************************
                                                        ** TESTS=1 PASS=1 FAIL=0 SKIP=0               1620.00           0.19       8385.08  **
                                                        ***************************************************************************************
```

From the left to the right: x_out[0], x_out[1], x_out[2], x_out[3], x_out[4]

Case1: 01000

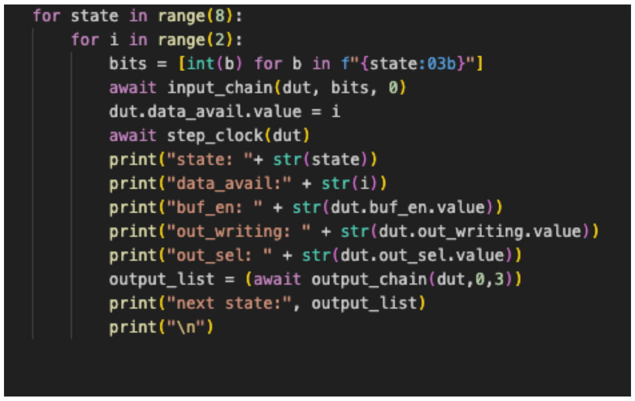
Case2:01100

Case3:00001

The result is correct

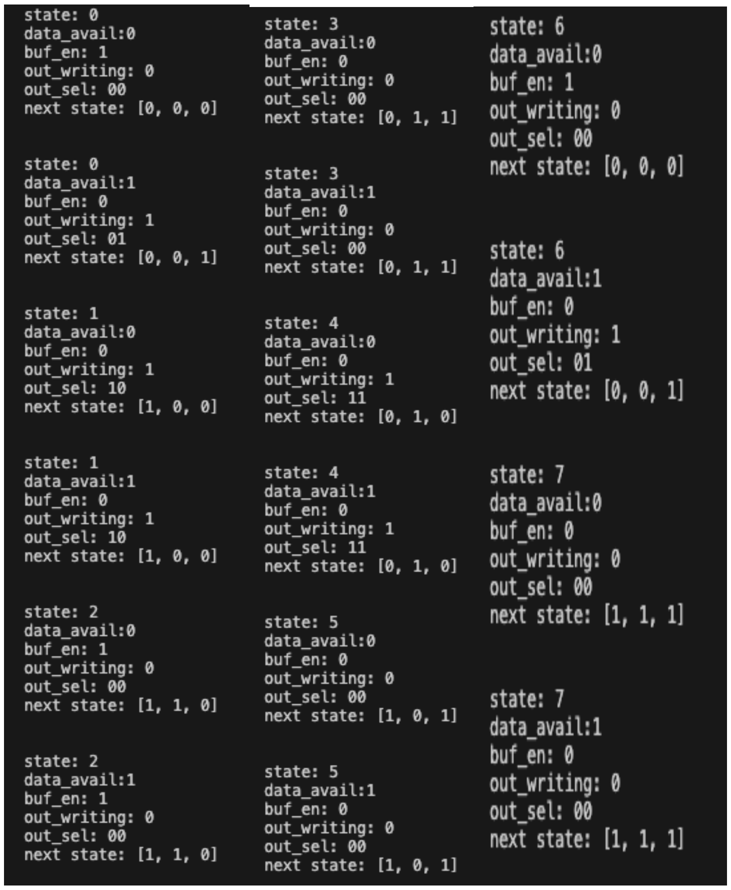
Task #3: FSM Testing

Code:



```
for state in range(8):
    for i in range(2):
        bits = [int(b) for b in f"{state:03b}"]
        await input_chain(dut, bits, 0)
        dut.data_avail.value = i
        await step_clock(dut)
        print("state: "+ str(state))
        print("data_avail:" + str(i))
        print("buf_en: " + str(dut.buf_en.value))
        print("out_writing: " + str(dut.out_writing.value))
        print("out_sel: " + str(dut.out_sel.value))
        output_list = (await output_chain(dut,0,3))
        print("next state:", output_list)
        print("\n")
```

Output:

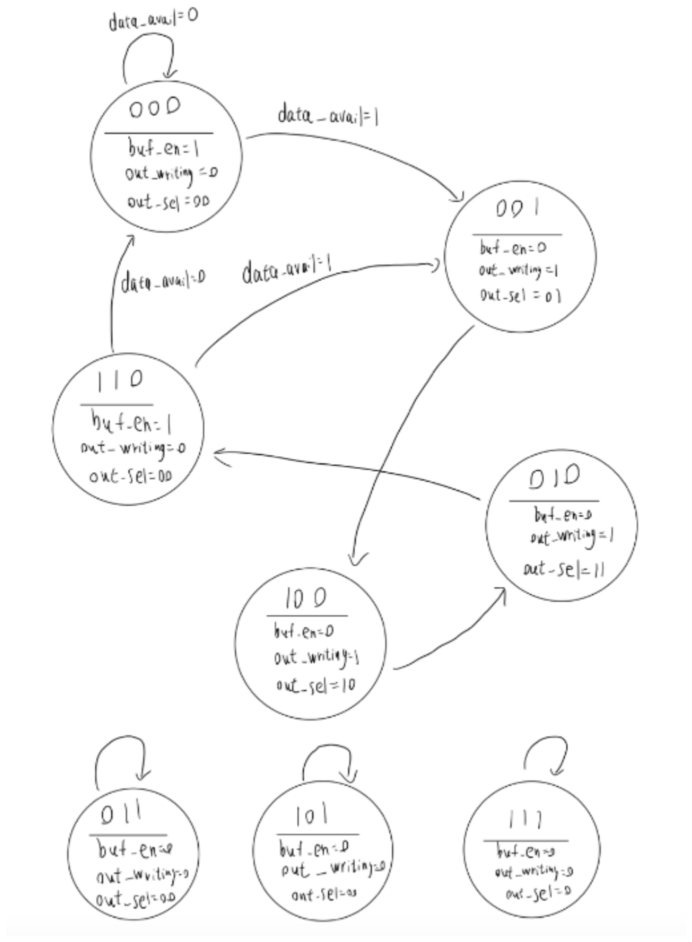


```
state: 0
data_avail:0
buf_en: 1
out_writing: 0
out_sel: 00
next state: [0, 0, 0]

state: 0
data_avail:1
buf_en: 0
out_writing: 1
out_sel: 01
next state: [0, 0, 1]

state: 1
data_avail:0
buf_en: 0
out_writing: 1
out_sel: 10
next state: [1, 0, 0]

state: 1
data_avail:1
buf_en: 0
out_writing: 1
out_sel: 10
next state: [1, 0, 0]

state: 2
data_avail:0
buf_en: 1
out_writing: 0
out_sel: 00
next state: [1, 1, 0]

state: 2
data_avail:1
buf_en: 1
out_writing: 0
out_sel: 00
next state: [1, 1, 0]
```

```
state: 3
data_avail:0
buf_en: 0
out_writing: 0
out_sel: 00
next state: [0, 1, 1]

state: 3
data_avail:1
buf_en: 0
out_writing: 0
out_sel: 00
next state: [0, 1, 1]

state: 4
data_avail:0
buf_en: 0
out_writing: 1
out_sel: 11
next state: [0, 1, 0]

state: 4
data_avail:1
buf_en: 0
out_writing: 1
out_sel: 11
next state: [0, 1, 0]

state: 5
data_avail:0
buf_en: 0
out_writing: 0
out_sel: 00
next state: [1, 0, 1]

state: 5
data_avail:1
buf_en: 0
out_writing: 0
out_sel: 00
next state: [1, 0, 1]
```

```
state: 6
data_avail:0
buf_en: 1
out_writing: 0
out_sel: 00
next state: [0, 0, 0]

state: 6
data_avail:1
buf_en: 0
out_writing: 1
out_sel: 01
next state: [0, 0, 1]

state: 7
data_avail:0
buf_en: 0
out_writing: 0
out_sel: 00
next state: [1, 1, 1]

state: 7
data_avail:1
buf_en: 0
out_writing: 0
out_sel: 00
next state: [1, 1, 1]
```

State diagram:



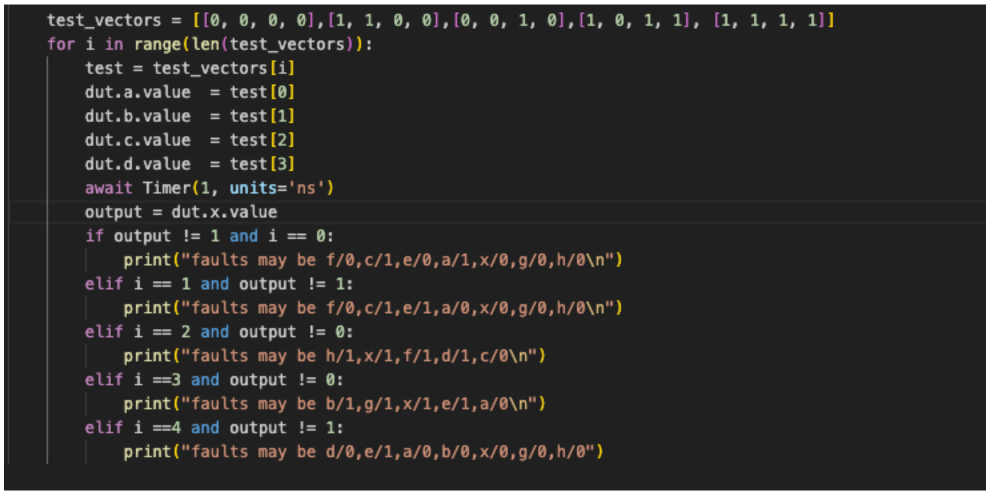
Invalid state: 011,101,111

Task #4: Fault Models

Minimum test vectors:

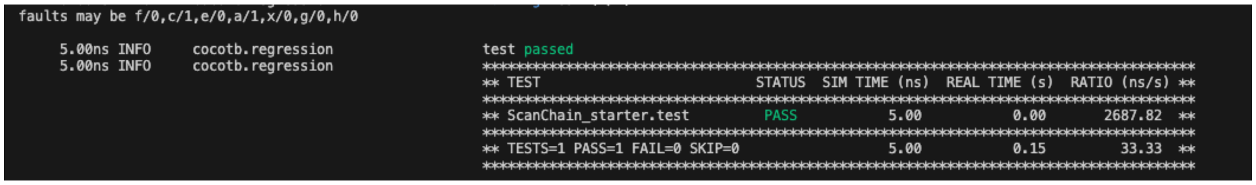
Test vectors	Fault free output	Detected faults
0000	1	f/0,c/1,e/0,a/1,x/0,g/0,h/0
1100	1	f/0,c/1,e/1,a/0,x/0,g/0,h/0
0010	0	h/1,x/1,f/1,d/1,c/0
1011	0	b/1,g/1,x/1,e/1,a/0
1111	1	d/0,e/1,a/0,b/0,x/0,g/0,h/0

Test code:



```python
test_vectors = [[0, 0, 0, 0],[1, 1, 0, 0],[0, 0, 1, 0],[1, 0, 1, 1], [1, 1, 1, 1]]
for i in range(len(test_vectors)):
    test = test_vectors[i]
    dut.a.value  = test[0]
    dut.b.value  = test[1]
    dut.c.value  = test[2]
    dut.d.value  = test[3]
    await Timer(1, units='ns')
    output = dut.x.value
    if output != 1 and i == 0:
        print("faults may be f/0,c/1,e/0,a/1,x/0,g/0,h/0\n")
    elif i == 1 and output != 1:
        print("faults may be f/0,c/1,e/1,a/0,x/0,g/0,h/0\n")
    elif i == 2 and output != 0:
        print("faults may be h/1,x/1,f/1,d/1,c/0\n")
    elif i ==3 and output != 0:
        print("faults may be b/1,g/1,x/1,e/1,a/0\n")
    elif i ==4 and output != 1:
        print("faults may be d/0,e/1,a/0,b/0,x/0,g/0,h/0")
```

Fault1:

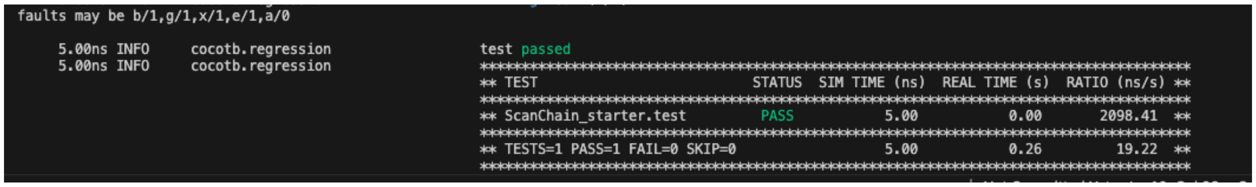


```
faults may be f/0,c/1,e/0,a/1,x/0,g/0,h/0

     5.00ns INFO     cocotb.regression                  test passed
     5.00ns INFO     cocotb.regression                  ***************************************************************************************
                                                        ** TEST                          STATUS  SIM TIME (ns)  REAL TIME (s)  RATIO (ns/s) **
                                                        ***************************************************************************************
                                                        ** ScanChain_starter.test         PASS           5.00           0.00       2687.82  **
                                                        ***************************************************************************************
                                                        ** TESTS=1 PASS=1 FAIL=0 SKIP=0                  5.00           0.15         33.33  **
                                                        ***************************************************************************************
```

The fault could be e/0, a/1

Because only test {00000}fails and other tests are passed

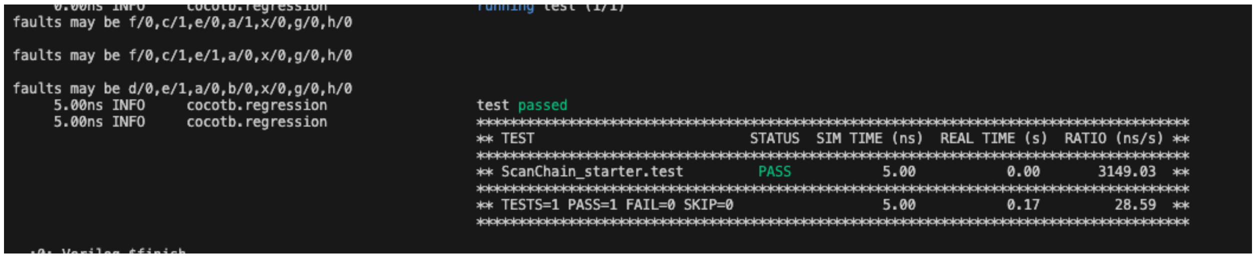
Fault2:



```
faults may be b/1,g/1,x/1,e/1,a/0

     5.00ns INFO     cocotb.regression                   test passed
     5.00ns INFO     cocotb.regression                   ******************************************************************************
                                                         ** TEST                          STATUS  SIM TIME (ns)  REAL TIME (s)  RATIO (ns/s) **
                                                         ******************************************************************************
                                                         ** ScanChain_starter.test         PASS           5.00           0.00       2098.41  **
                                                         ******************************************************************************
                                                         ** TESTS=1 PASS=1 FAIL=0 SKIP=0                  5.00           0.26         19.22  **
                                                         ******************************************************************************
```

The fault could be b/1, g/1 because only test {1011} fails and other tests are passed

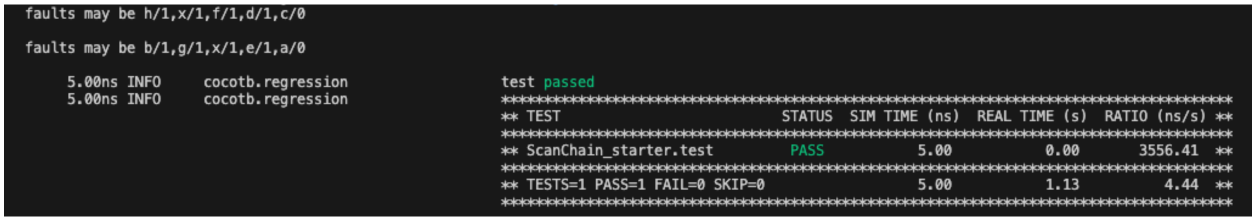
Fault3:



```
faults may be f/0,c/1,e/0,a/1,x/0,g/0,h/0

faults may be f/0,c/1,e/1,a/0,x/0,g/0,h/0

faults may be d/0,e/1,a/0,b/0,x/0,g/0,h/0
     5.00ns INFO     cocotb.regression                   test passed
     5.00ns INFO     cocotb.regression                   ******************************************************************************
                                                         ** TEST                          STATUS  SIM TIME (ns)  REAL TIME (s)  RATIO (ns/s) **
                                                         ******************************************************************************
                                                         ** ScanChain_starter.test         PASS           5.00           0.00       3149.03  **
                                                         ******************************************************************************
                                                         ** TESTS=1 PASS=1 FAIL=0 SKIP=0                  5.00           0.17         28.59  **
                                                         ******************************************************************************
```

The fault could be x/0, g/0, h/0. Just use the intersection of faults detected by test {0000}, {1100}, {0010}

Fault4:



```
faults may be h/1,x/1,f/1,d/1,c/0

faults may be b/1,g/1,x/1,e/1,a/0

     5.00ns INFO     cocotb.regression                   test passed
     5.00ns INFO     cocotb.regression                   ******************************************************************************
                                                         ** TEST                          STATUS  SIM TIME (ns)  REAL TIME (s)  RATIO (ns/s) **
                                                         ******************************************************************************
                                                         ** ScanChain_starter.test         PASS           5.00           0.00       3556.41  **
                                                         ******************************************************************************
                                                         ** TESTS=1 PASS=1 FAIL=0 SKIP=0                  5.00           1.13          4.44  **
                                                         ******************************************************************************
```

The fault could be x/1, Just use the intersection of faults detected by test{0010}, {1011}

Fault 5:

No fault detected

Task #5: Reflection

1. I can find some tests that can mask some faults, and activate and propagate the other fault. In addition, multiple stuck at faults model can be used to address this case
2. Benefits are increasing the fault coverage, easy to test faults. Downsides are increasing the chip area and impacting the timing path of the functional mode
3. I may design BIST circuit in the design. The circuit doesn't have scan chain and external testing points, but it can test itself to make sure the circuit has no faults
4. After optimizing, the existing scan chain may be broken. In addition, the timing issues may be triggered.

Task #6: Project work

I finished basic RTL design and built a testbench to verify all functions. There are some errors and I will try my best to solve them.