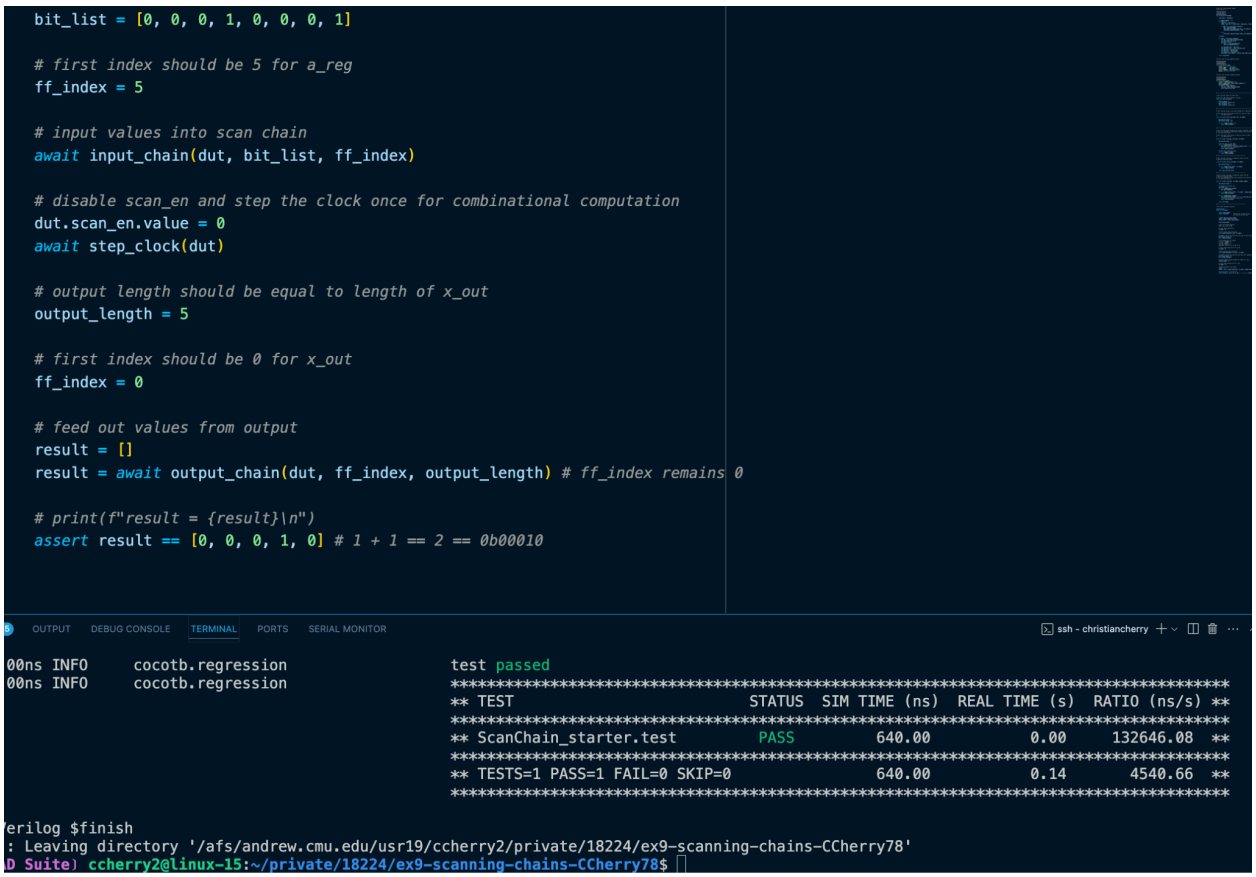
Christian Cherry

18224 EX9

Task #2 (Testing the Adder)

My first step was to find what CHAIN_LENGTH was for the adder, which was found with chain.chain_length. I then printed out the scan chain with print_chain and found that x_out, a_reg, and b_reg had corresponding indices in the scan chain. A_reg and b_reg take up 8 indices for the scan chain sequentially, with x_out taking up the first four. I then set up a bit list for my input values that represent a_reg and b_reg, and the 0s for x_out, and set my ff_index to 0 to match the x_out index, calling the multi-bit input chain function. After disabling the scan chain and stepping the clock once, I got out my x_out values by calling the output_chain function on the 0th index and the length of x_out. I then had to update the makefile to use adder_out.sv with the scan chains. I then had to rework how I was indexing into the scan chain for output values, then finally passed the addition test.

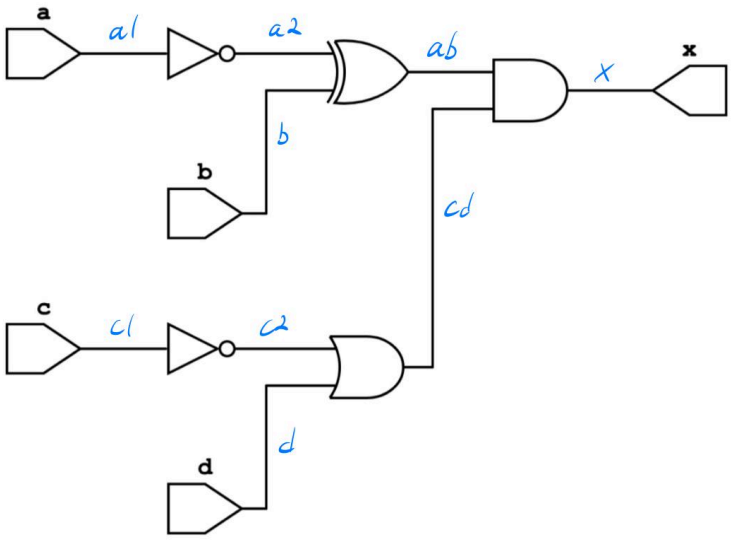
1 + 1 output:



```
    bit_list = [0, 0, 0, 1, 0, 0, 0, 1]

    # first index should be 5 for a_reg
    ff_index = 5

    # input values into scan chain
    await input_chain(dut, bit_list, ff_index)

    # disable scan_en and step the clock once for combinational computation
    dut.scan_en.value = 0
    await step_clock(dut)

    # output length should be equal to length of x_out
    output_length = 5

    # first index should be 0 for x_out
    ff_index = 0

    # feed out values from output
    result = []
    result = await output_chain(dut, ff_index, output_length) # ff_index remains 0

    # print(f"result = {result}\n")
    assert result == [0, 0, 0, 1, 0] # 1 + 1 == 2 == 0b00010
```

```
OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SERIAL MONITOR                ssh - christiancherry

00ns INFO     cocotb.regression                 test passed
00ns INFO     cocotb.regression                 ****************************************************************************************
                                                ** TEST                          STATUS  SIM TIME (ns)  REAL TIME (s)  RATIO (ns/s) **
                                                ****************************************************************************************
                                                ** ScanChain_starter.test         PASS         640.00           0.00     132646.08  **
                                                ****************************************************************************************
                                                ** TESTS=1 PASS=1 FAIL=0 SKIP=0                640.00           0.14       4540.66  **
                                                ****************************************************************************************

erilog $finish
: Leaving directory '/afs/andrew.cmu.edu/usr19/ccherry2/private/18224/ex9-scanning-chains-CCherry78'
D Suite) ccherry2@linux-15:~/private/18224/ex9-scanning-chains-CCherry78$
```

Task #4 (Fault Models)



MSB = a

	SA0	SA1
a1	{10000}	{00000}
a2	{00000}	{10000}
b	{11001}	{10000}
ab	{11001}	{10001}
c1	{11110}	{11001}
c2	{11001}	{11110}
d	{11111}	{11110}
cd	{11111}	{11110}
x	{11111}	{11110}

Vectors that catch multiple faults

{00000}
{10000}
{11001}
{11110}
{11111}

Task #5 (Reflection)

1. One other fault that could happen during manufacturing could be a signal that is slow to rise or fall. You could detect it by activating a signal transition, and seeing if it transitions in enough time, similar to the stuck-at fault model, but with respect to time. Another fault could be bridges, where two signals are stuck together because of some physical short. You could detect this by seeing if two wires always change at the same time when you only apply a signal to one of them.
2. Having a scan chain allows you to have controllability and observability for your design, but they take up more physical space and can decrease the speed and increase the power consumption of your design.
3. Another method you can use if scan chains is not an option is possibly a built-in self test, where your design self-checks if it is functioning correctly.
4. Having scan chains inserted before optimization can cause possible timing issues or space issues. These issues could be dealt with by only inserting scan chains after optimization.

Task #6 (Project Work)

I am currently in the process of making my I2C master receiver based on the datasheet for the LIS33DE accelerometer. I have the FSM-D mostly done, but I need to work out some kinks. I plan to move on to the design of the neuron modules soon.