**Working of the design code:**

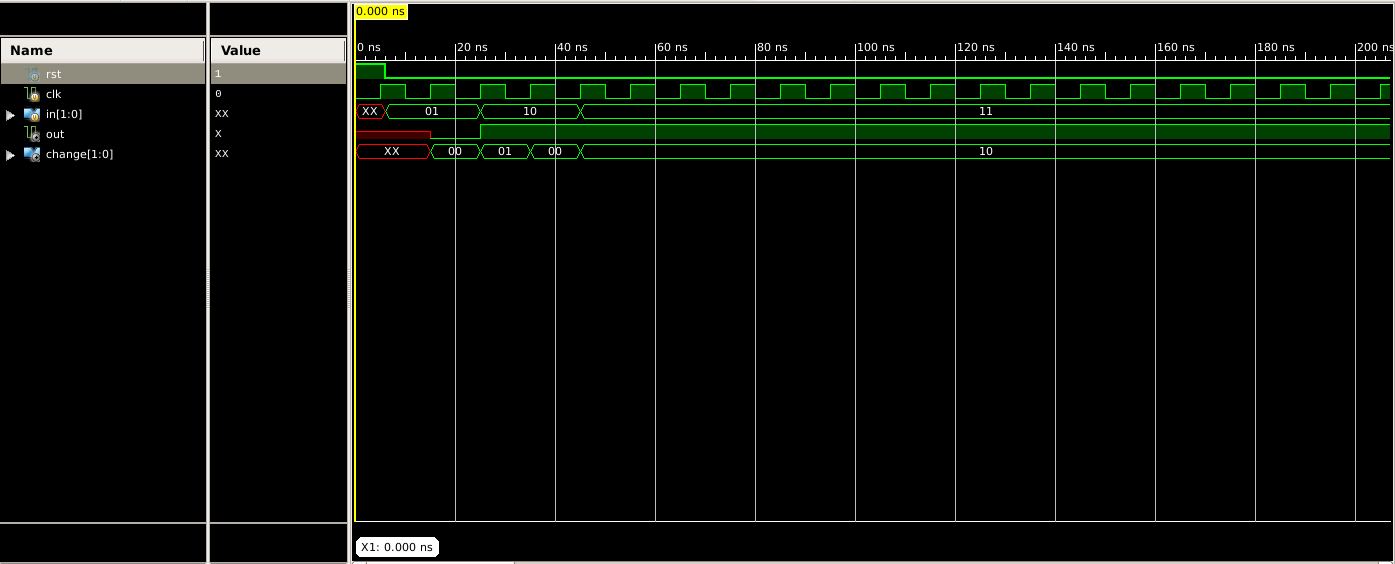
First, we have fixed 3 input formats, if the input is 2’b01=Rs5 or 2’b10=Rs10 or 2’b11=Rs20. Also 2’b11 equals Rs20 only if it is given as input otherwise 2’b11 represents Rs15 State. Now a module has been created and the ports have been mentioned within the brackets. Within the module block we first mention the parameters s0,s1,s2,s3 these will signify our different states of the FSM model we developed In the previous sections. Rs0 state/Reset state, Rs5 state, Rs10 state, Rs15 state are represented by s0,s1,s2,s3 respectively. We also have created a parameter n which will act as our product selector i.e whether it is a Rs10,Rs15,Rs20 product. Now inside the always block, according to the value of n there will be a different section of the code that will run and give us the output. For example if n=10:- This program runs based on case statements. The first case is for s0: within these cases there are multiple if loops that will determine the next state of the vending machine and also the output. If the input is of 2’b00 which is essentially Rs0 the state of the machine will not change and also won’t give any output, now when the input is 2’b01 the next becomes s1 as the input given was Rs5 but it is not sufficient to give the desired product which is of worth Rs10 and so the machine waits for more currency to be added until it has enough to dispense the product with the change if any. Similarly the program will run the same way our FSM model works.

**Working of the test bench code:**

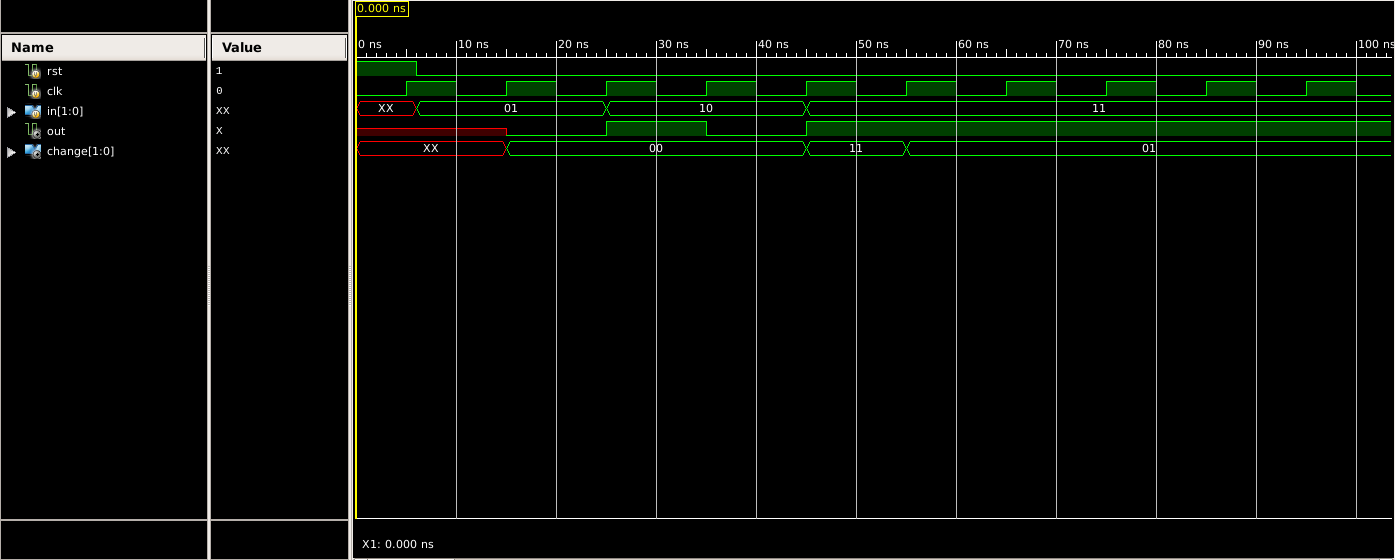
Test bench coding is necessary as it will verify the design of our program and it also ensures the timing and overall functionality of the machine designed. Code starts with module creation, inside the module we define the input and output ports as reg and wire respectively. Then we instantiate the design module that we had created in the section prior to this. Now we initialize rst=1 and clk=0, when rst=0 the machine will start working with every positive edge of the clk waveform. After a #6 delay we set rst to zero and give our first denomination input which is 2’b01 (Rs5), after another #19 delay we give our second input of 2’b10 (Rs10) this delay is given so that we can study and understand the working accurately. Further input of 2’b11 (Rs20) is given after another #20 for the same reason and the block is ended. We set the clock waveform to have high and low values every 5ns.

**Waveforms:**

N=10



N=15



N=20

