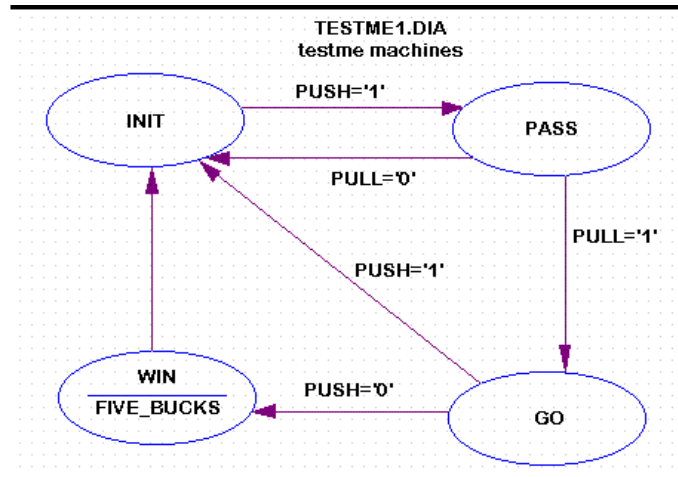


TESTING STATE MACHINES PHYSICAL TESTING

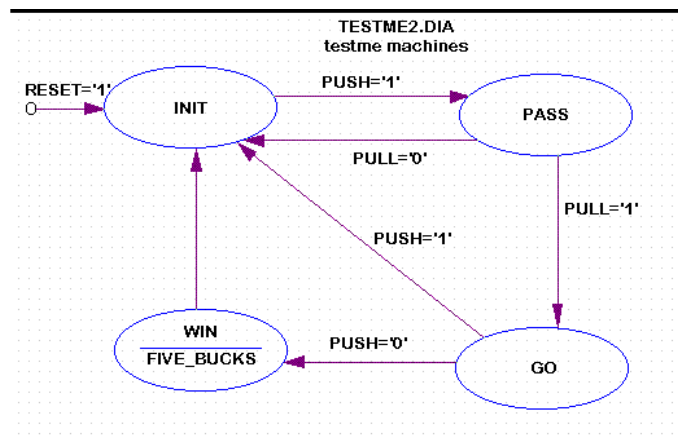
When a state machine is designed, it may be simulated and tested at various places in the design process. These notes deal with the testing of the actual machine. We shall divide this subject into three parts, depending on the nature of the machine.

The first version of this machine has two inputs and one output. It has no RESET and the state variables are not accessible (they are buried nodes). This is the hardest of the three types of machine to test. The testing usually involves some analysis of the machine to determine (1) how to get it into a known state, (2) how to determine which if the four states the machine may be in, and, finally, (3) how to check all of the transitions.



The second version of this machine is shown next.

This is identical to the first machine except it has an asynchronous reset. This eliminates the first problem with the first machine. We can get it into a known state (assuming the reset works). It also makes the second task a bit easier.



Since the testing of either of these two machines is rather complicated, we shall go to the third type which is the easiest to test.

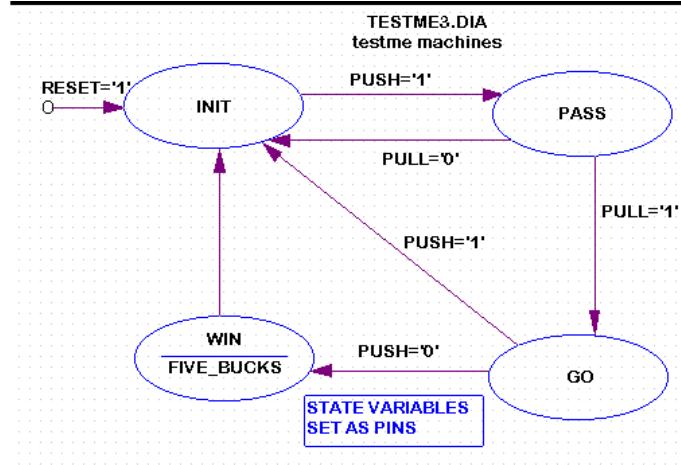
This machine is shown on the next page. To implement in StateCAD, one only need pull up the VARIABLE menu and change the state variables as shown below.

CLK	CLOCK	HIGH	PIN
FIVE_BUCKS	OUT:COM	HIGH	PIN
PULL	INPUT	HIGH	PIN
PUSH	INPUT	HIGH	PIN
RESET	INPUT	HIGH	PIN
SV0	STATE	HIGH	PIN
SV1	STATE	HIGH	PIN

The have her been changed from NODE to PIN.

With the ability to reset to the machine to a known state and the ability to monitor its states, the only task is to test each transition (and output). To make a thorough test, all transitions must be covered (at least) once. The goal is to do this with the least number of tests.

The general procedure for doing this is to prioritize transitions, taking them in the following order: (1) transitions which keep the machine in the same state, (2) transitions which lead the machine to a state from which a second transition may be made back to the first state, (3 and 3+) transitions which lead away from the current state. The philosophy of the above is simple: test all transitions in a given state (if possible) before getting too far away from it.



Let us apply this concept. (1) RESET the machine to INIT. Then (2) set PUSH= 0 and clock the machine. It should stay at INIT. Next (3) set PUSH= '1' and clock the machine. It should go to PASS. State INIT has all transitions tested.

At state PASS, the shortest path back to PASS is via INIT. (4) set PULL to 0 and clock the machine. It should end up in state INIT. (5) with PUSH still 1, clock the machine again to return it to PASS. Step (5) does not do any testing!. Next (6) with PULL= '1', clock the machine to move it to GO. The PASS state has now been tested.

(7) With PUSH= 1, clock the machine. It should go to state INIT. Steps (8) and (9) are then needed to get back to GO. These are the same as steps (5) and (6) and perform no new tests.

We test the other transition at GO by setting (10) PUSH= 0 and clocking the machine. This brings us to WIN with FIVE_BUCKS asserted. One more clock (11) should return the machinf to INIT, completing the test. This may be described with a set of TEST VECTORS

Note that ten test steps are required to test seven transitions. It is unlikely that another test strategy would take less; some would take more.

step	reset	state	push	pull	five_bucks
1	1	?	-	-	0
2	0	INIT	0	-	0
3	0	INIT	1	-	0
4	0	PASS	-	0	0
5	0	INIT	1	-	0
6	0	PASS	-	1	0
7	0	GO	1	-	0
8	0	INIT	-	1	0
9	0	PASS	0	-	0
10	0	WIN	-	-	1
DONE	0	INIT	-	-	-

Now, how to test the second machine where we can not monitor the states? How can we determine what state the machine goes to? In this case, we can tell when its in the WIN state by its distinct output. We can set it, initially to the INIT state. What about the other two states? If we assume it to be in state GO, setting PULL= 1 and clocking the machine should put in in the WIN state. If it does, we were in the GO state. If we believe it to be in the PASS state, setting PUSH to 0 and PULL to 1, and clocking the machine twice should put us in the WIN state. A three step sequence is needed to verify the INIT state.

The test for machine 3 above has to be modified so that we can verify each state. This adds considerably to the number of tests required. We will not here go into the process of determining what they should be.

The first machine has no reset so we have to get the machine into a known state before we begin the actual testing. In this case we would have to get it to the WIN state to begin testing.

ALTERNATE NOTES

TESTING WITH STATE CAD