

StateCAD

State Machine
Design

Overview

This tutorial discusses how to use StateCAD inside of XILINX. StateCAD is a visual representation of a **State Diagram**. State Diagrams, discussed in the textbook (Katz has an example of a combination lock represented using a state diagram on Pages 20-21), visually describe the behavior of a sequential machine.

StateCAD lets you layout a sequential machine, defining outputs, inputs, and transitions between states. When the diagram is complete the software translates the visual representation into a Hardware Description Language (HDL). It is not necessary to understand the language. You may want to note, however, that you have little control over how the software translates your diagram into useable logic. By default, StateCAD uses **One Hot Encoding** (discussed in the class lectures). Because of this, the logic may not be fully minimized because One Hot Encoding uses more flip flops. The tradeoff, however, is that the feedback logic is *usually* simpler. For using StateCAD, you will not need to understand how to translate from a state diagram into useable sequential logic. StateCAD does everything for you.

After the HDL has been generated you can add it to your project as a source and turn it into a schematic symbol. For this tutorial, we will create a state diagram for a **vending machine**, generate the HDL, and create the schematic symbol. This tutorial is broken up into the three following sections.

- Opening StateCAD and Creating the Diagram
- Setting Up Variables and Using Statebench
- Creating the Schematic Block

First, however, we must know what states we would like to make and the inputs.

Vending Machine Specifications

For simplicity, let us assume that you can place nickels, dimes, and quarters into the machine. Any combination of coins adding up to fifty cents will yield a “ready to vend” command telling the machine to vend a drink when a choice has been selected (This logic does not deal with what drink the user wants, only that he or she has entered enough coins). We will also need to deal with change being distributed if the amount is over fifty cents. Because of existing equipment, the “change” output will need to be the actual number in eight bit binary. For instance, if the change is supposed to be fifteen cents, we will need to output the number fifteen in binary or “00001111”.

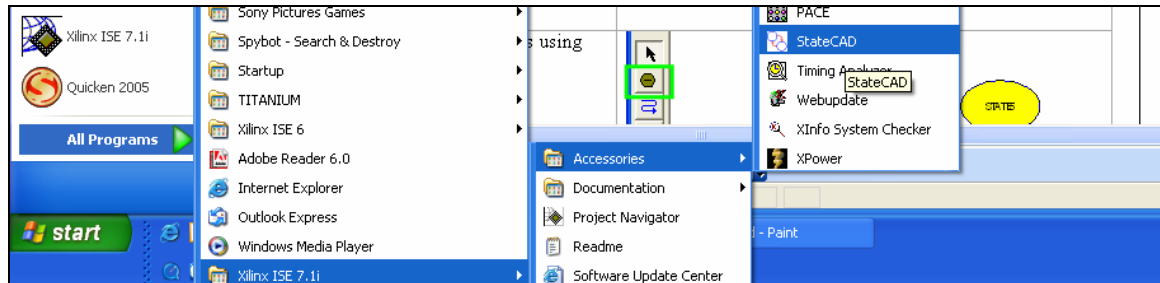
For the inputs, let us assume that we have two bits.

	<u>COIN</u>	<u>IN1</u>	<u>IN2</u>
1.	A nickel is	‘0’	‘0’
2.	A dime is	‘0’	‘1’
3.	A quarter is	‘1’	‘1’
4.	No input is	‘1’	‘0’

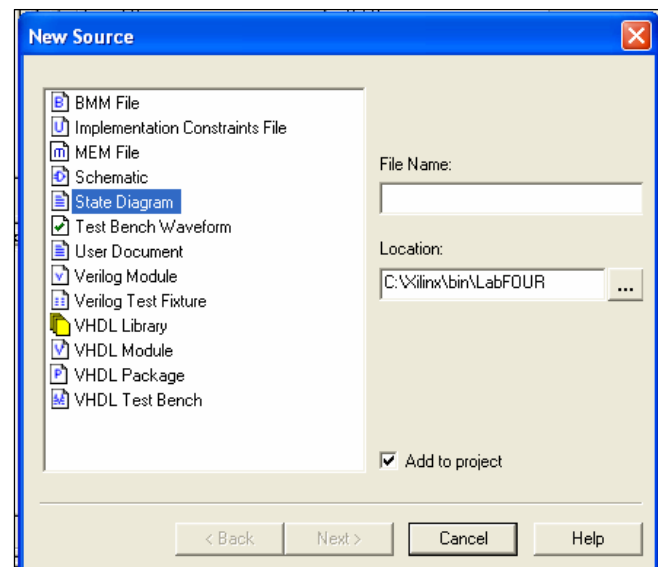
With this we should be able to completely define our state machine.

Opening StateCAD and Creating the Diagram

1. **Open StateCAD by navigating through the start menu**
Start>All Programs>XILINX ISE 7.1i>Accessories>StateCAD.

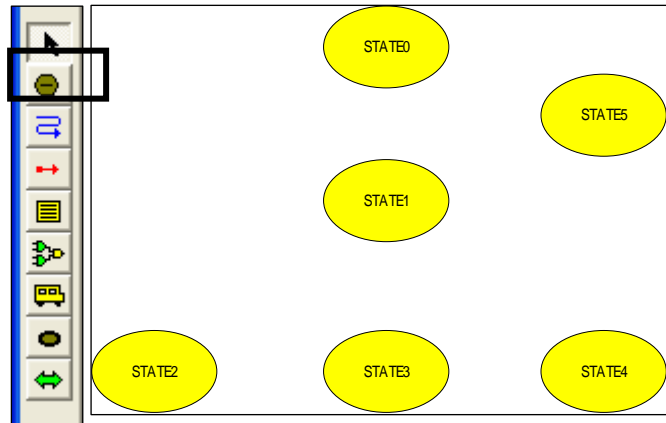


2. You can also access StateCAD by **adding a new source** to your XILINX project and choosing **State Diagram** from the source list.

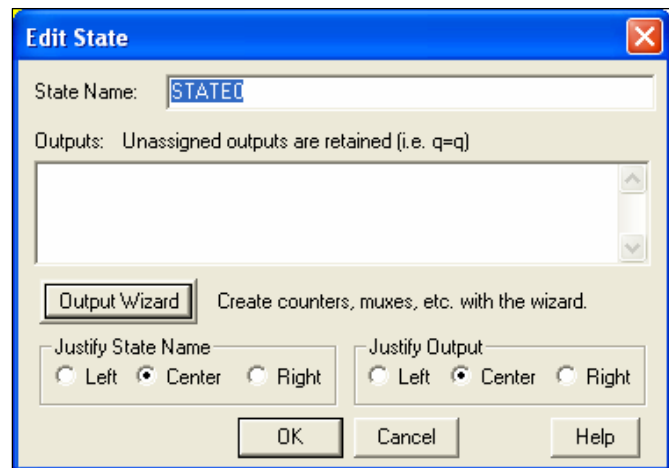


3. To make the Machine, we will need several states:
 - An initial state
 - A state that waits for input
 - A state for each coin that is input: Nickel, Dime, and Quarter
 - A state that outputs “Vend” and the change amount.
 We will use an internal variable to keep track of how much money is input.
 The machine will stay in the “vend” state until it is reset.

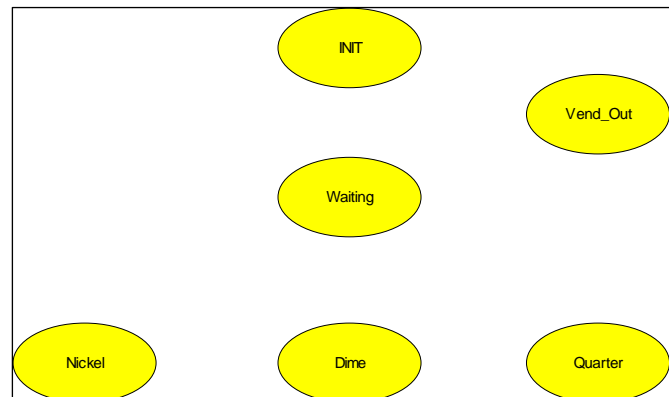
4. **Place States** on the **canvas** using the highlighted tool.



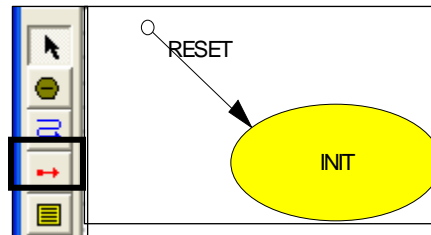
5. **Double click** on the **top state** to bring up another menu. With this menu you can **rename** and **define the outputs** of a state. For now, only change the name to **INIT**.



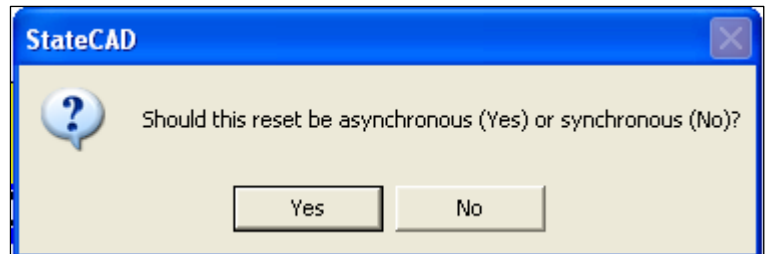
6. **Rename** the rest of the states something meaningful. For this tutorial the names **INIT**, **Waiting**, **Nickel**, **Dime**, **Quarter**, and **Vend_Out** are used.



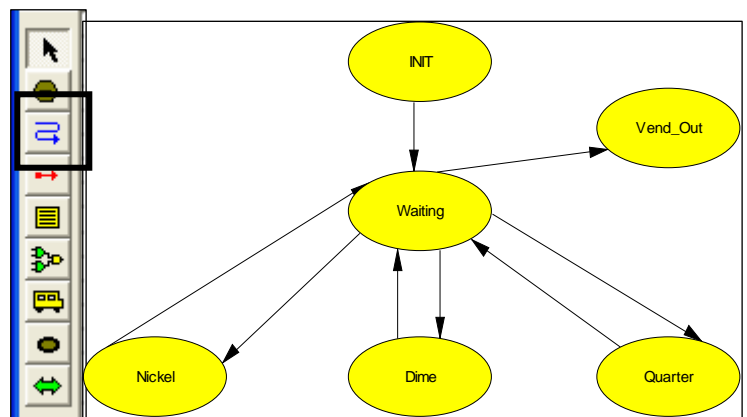
7. **Add a Reset** (where the diagram will start) using the highlighted tool. **Click** near and then **click on** the state you want to reset to.



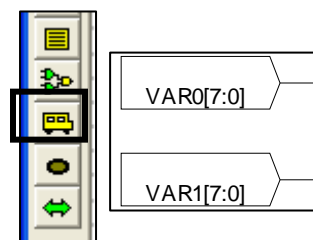
8. When this menu appears **select asynchronous (Yes)**. For this class, you will always use an asynchronous reset.



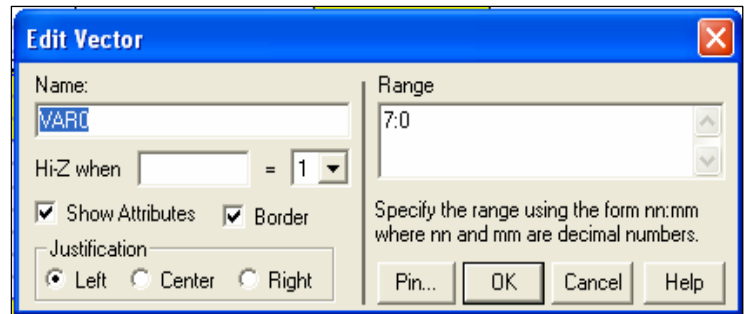
9. Each state will need transitions to and from itself. Transitions only occur on the rising edge of the clock. With the highlighted tool, **add the transitions** shown by **clicking on one state and clicking on the state it will transition to**. Later, conditions will be added to the transition so StateCAD knows under what inputs the transition will occur.



10. Next we will need to create **bus**, or **vector**, variables. This diagram will use two eight-bit vectors. One **internal vector** to count the money inserted and one **output vector** to display change. To do this **click twice on the screen using the highlighted tool**.

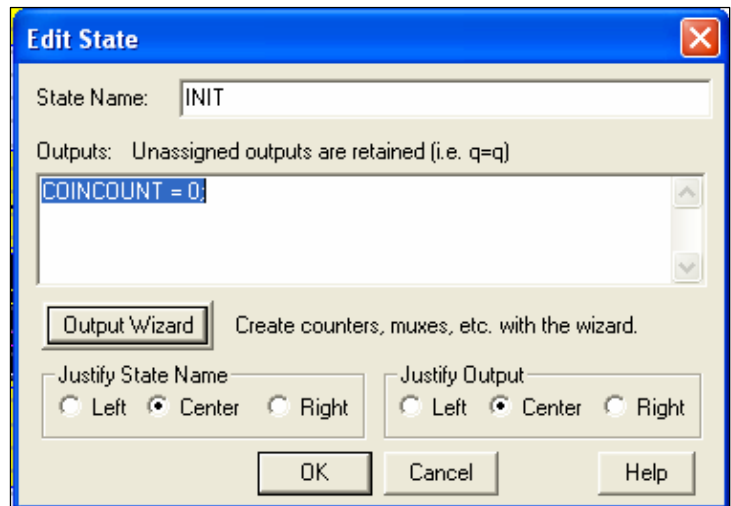


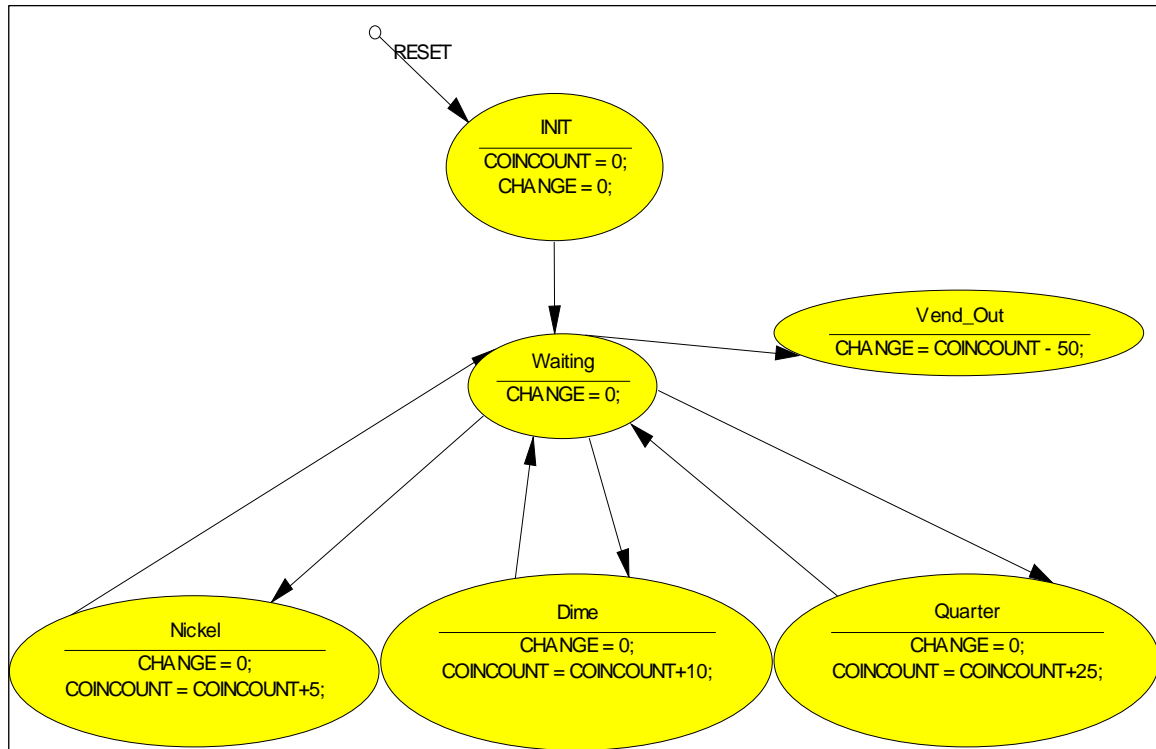
11. **Double click** on one of the vectors to bring up this menu. Here you can **rename** and **change the size** of the vector. The size is already eight bits, but change the name to something meaningful. This tutorial will use **CHANGE** and **COINCOUNT**.



12. We will need to specify our vector's values during the states. **Double click** the **INIT** state and **enter** the desired **vector name** and **value** inside the menu.

Syntax: For the **INIT** state, **COINCOUNT** will need to equal all zeros in binary. Since it is a vector, we can simply say **COINCOUNT = 0;** But we can also specify each bit of the vector in double quotes as **COINCOUNT = "00000000";** This tutorial will use the decimal method.





13. **Enter** the above vector values into your states. **COINCOUNT** will need to start at zero and increment appropriately when it reaches the nickel, dime, or quarter state. **CHANGE** will need to be zero in every state except for the Vend_out state when **CHANGE = COINCOUNT – 50**; The picture above illustrates these outputs.

14. To define the transitions, **double click** on the **transition arrow** to bring up this menu. For the transition from the **waiting** state to the **Vend_out** state, we want this to occur when **COINCOUNT** is greater than or equal to 50.

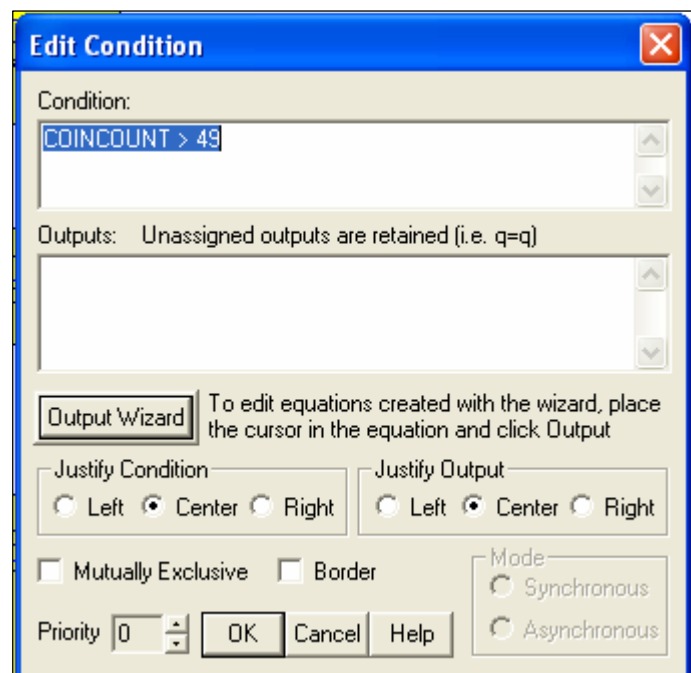
Syntax: No semicolon is used for transitions. The comparison operations are similar to **C** syntax.

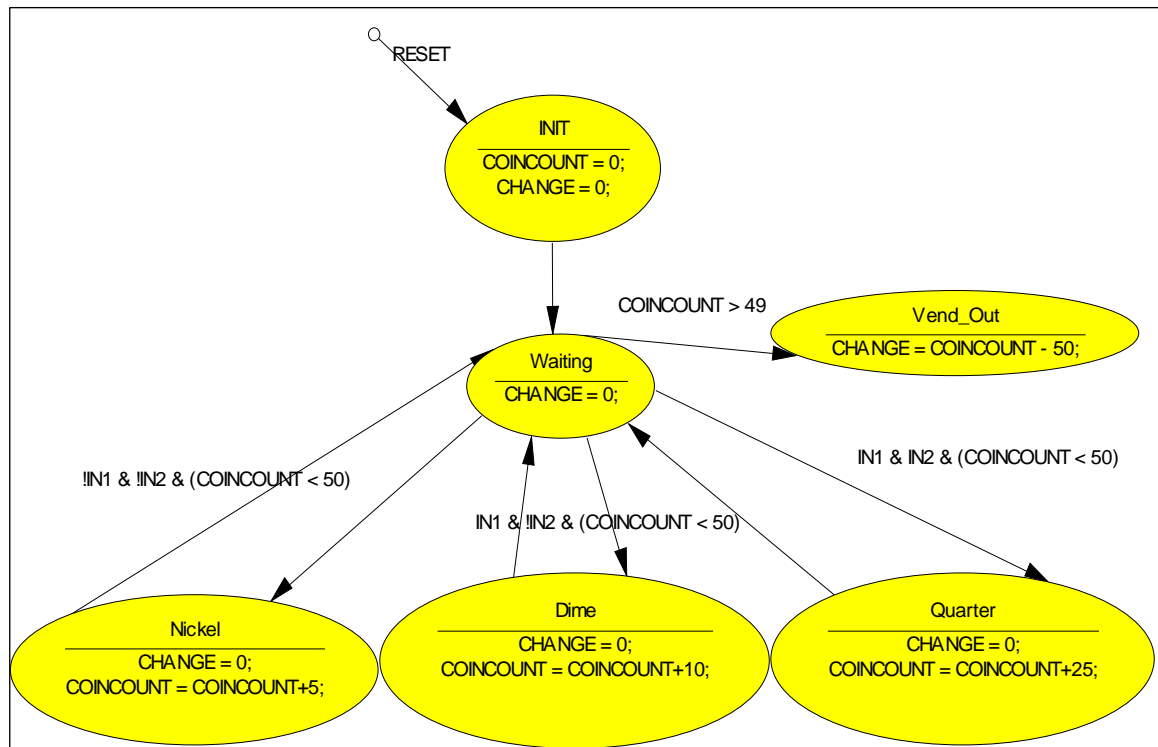
> is greater than

< is less than

== is equal to

Also, a transition with no conditions means that the transition will occur on the next clock tick.

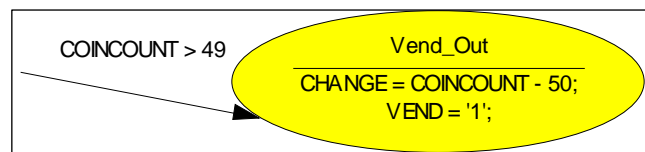




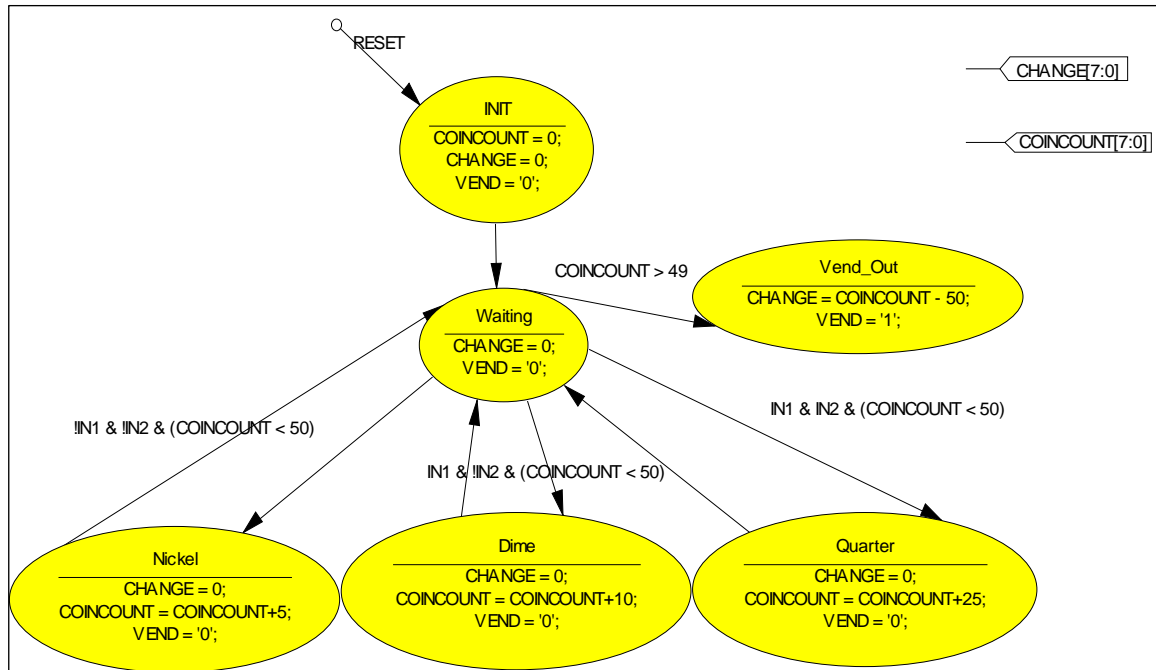
15. As illustrated above, the rest of the transitions from the waiting state depend on the two inputs and **COINCOUNT** being less than 50. Since a nickel is signified as both inputs being logic zero, the transition will be **!IN1 & !IN2 & (COINCOUNT < 50)**. You may have noticed that **IN1** and **IN2** are not declared. This is because it is not necessary to declare inputs that are not vectors. StateCAD will automatically add them to your variable list.

Syntax: Notice the use of parenthesis on the less than comparison. This is because StateCAD can only perform **Boolean operations** on **single bits**. The statement **COINCOUNT < 50** is converted to Boolean using the parenthesis.

16. You will need an output for letting the vending machine know that enough money has been entered, this tutorial will use **VEND**.



Syntax: This is not a vector, so you do not need to declare it. Since it is one bit, however, you will need to assign the logic value in single quotes. So when you want the variable, **VEND**, to be zero you will assign **VEND = '0'**; The variable will be zero everywhere except for the **Vend_out** state.

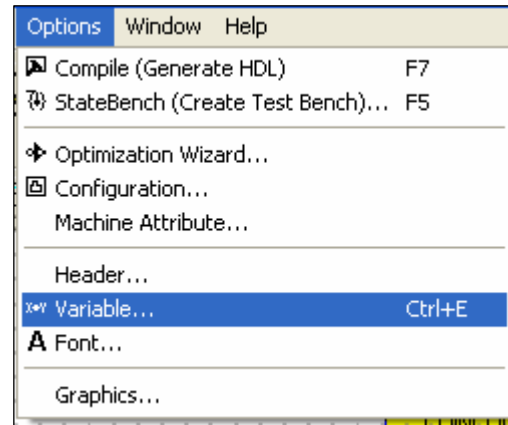


17. The **final diagram** should look like the figure above. If it does not, make the appropriate changes.

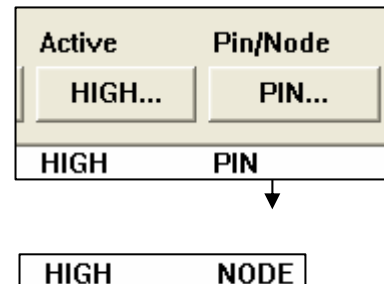
Setting Up Variables and Using Statebench

Now that you have completed the diagram you will need to set up the variables as nodes, inputs, or outputs. Nodes are variables internal to the State Diagram. For our tutorial, COINCOUNT will be a node.

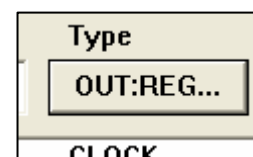
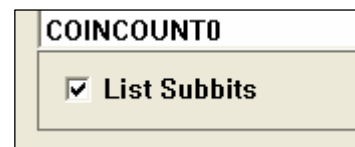
1. To change variable settings, click on **options>Variable**.



2. Inside this menu, you can access all the parameters of your inputs and outputs. Change **COINCOUNT** to a **node**, click on **COINCOUNT** and click on **PIN...** under **Pin/Node**. It should change to a node when you deselect it.

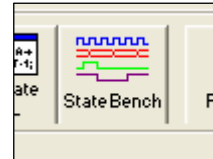


3. Instead of viewing the variables as vectors, you can click the **list Subbits** box and view every variable in your diagram. In the **Type** column, you can change the variable into an **OUT:REG** or an **OUT:COM**. **OUT:REG** stands for a registered output. This means that the output **only changes values on a clock tick**. All the outputs you make **should be registered** to prevent static hazards. If they are not, change them.

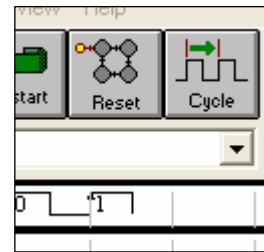


4. Before exiting the menu, be sure that all inputs and outputs are **active high**.

5. Before generating the **HDL** you will want to check the functionality of your design. Press the **Statebench** button on the top toolbar menu. You may receive warnings from StateCAD. Warnings do not necessarily mean your design is bad. For this tutorial, ignore the warnings.

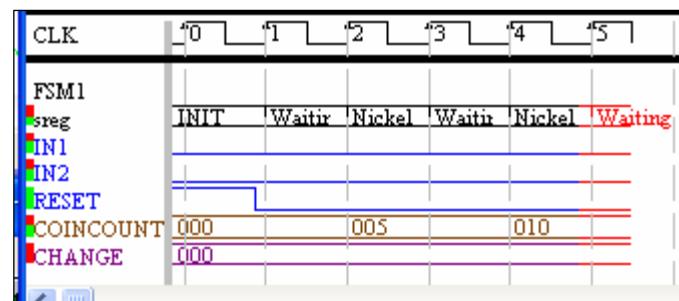


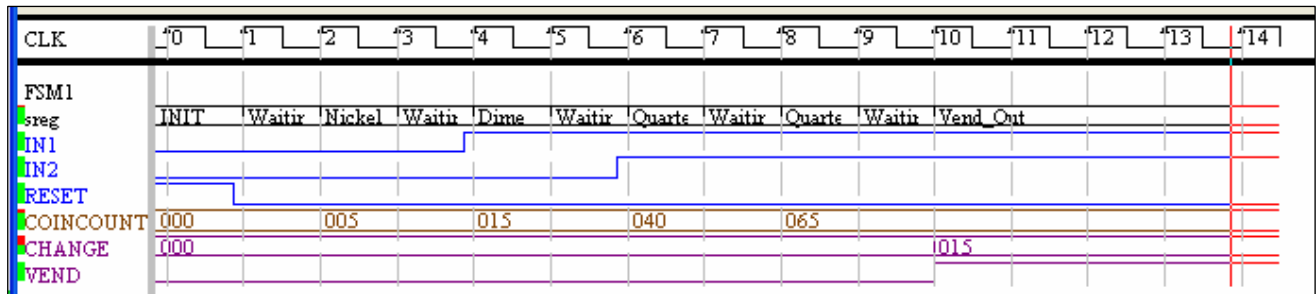
6. With **Statebench** you can go through your design one clock cycle at a time, **changing inputs** and **verifying output**. To begin press the **Reset** button and begin cycling your diagram.



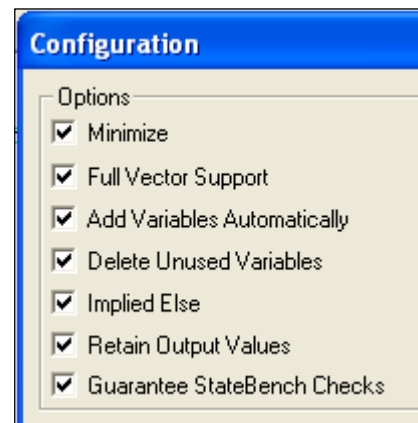
7. To change an input **double click** on the **right most edge** of the cycle. The edge is highlighted in red.

As you cycle through you will also notice that the state you are in is highlighted in a separate window above the testbench. Any imminent transitions are also highlighted as the state makes them.





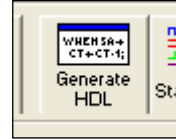
8. Above is the fully specified test bench for the vending machine with a nickel, dime, and two quarters inserted. The change is indeed 15 and vend is taken to logic one.
9. Close Statebench. You will not need to save the testbench.
10. You will need to make one last change to your outputs. **Go to options>configuration.** Make sure that the boxes shown are checked.



11. **Save** the state diagram by **clicking the save file button** in the top left corner. Save the file into the same file that your project is saved in.

Syntax: When naming the file, you will need to adhere to **Dos naming conventions**. Only eight characters, no spaces- only letters, numbers, underscore.

12. **Exit the configuration menu** and click on the **Generate HDL** button. Ignore warnings and close the HDL file when it opens.

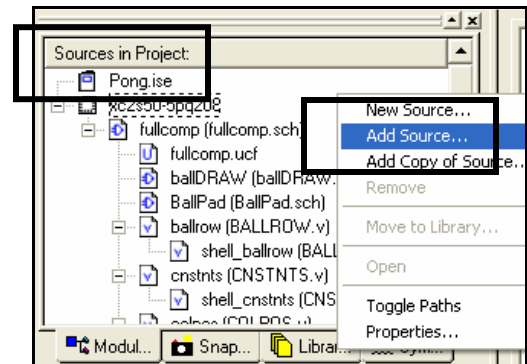


This creates a **VHDL file** in the file the StateCAD diagram has been saved in.

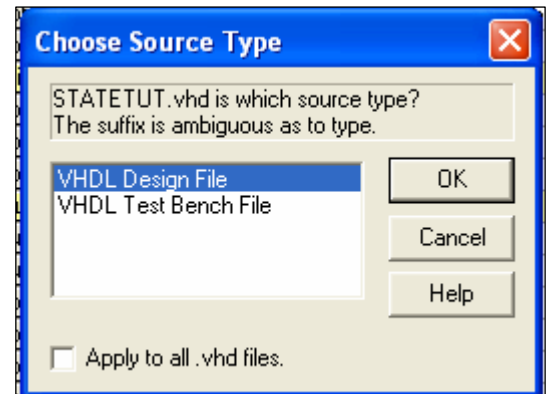
13. **Exit StateCAD.**

Creating the Schematic Symbol

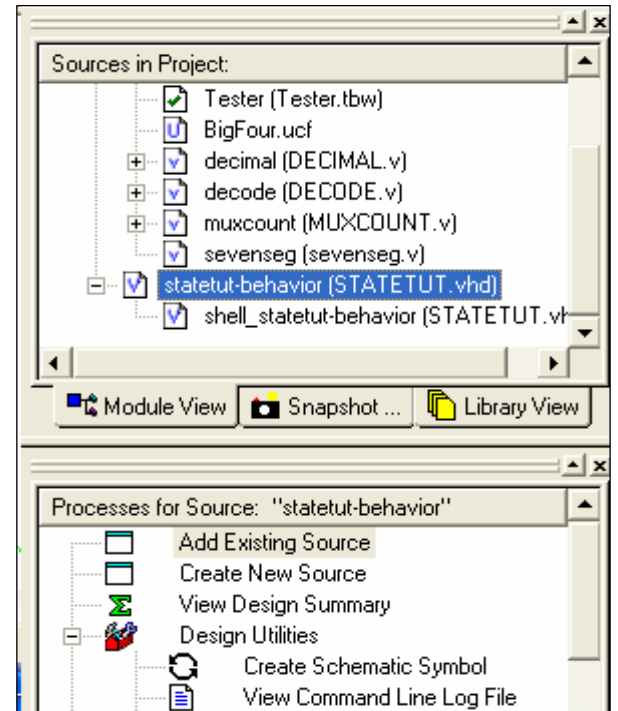
1. **Right Click** inside the **Sources in Project** window and select **add source**.



2. **Double Click** on your .vhd file and select **VHDL Design File** when prompted about source type. Ours is called STATETUT.vhd



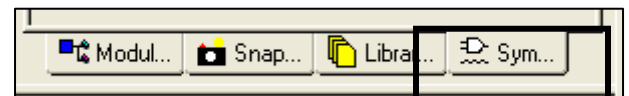
3. **Highlight STATETUT.vhd** in the **Sources in Project** window by clicking once on the module name.



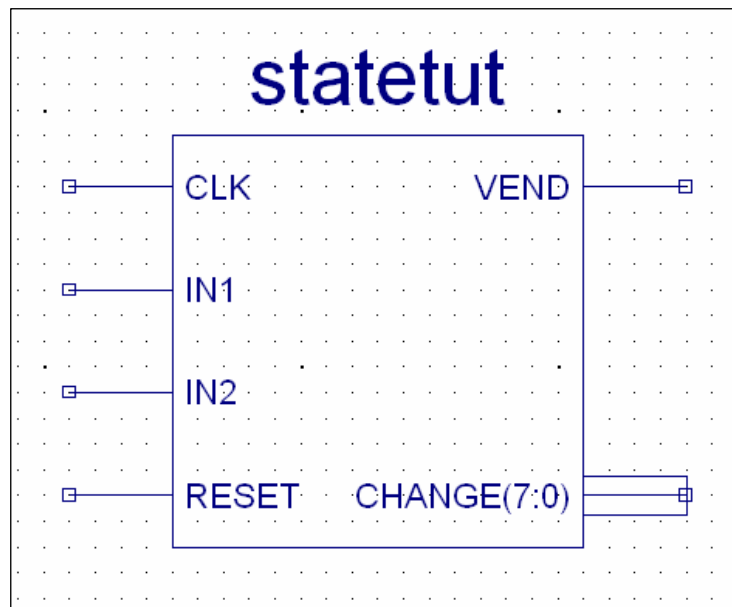
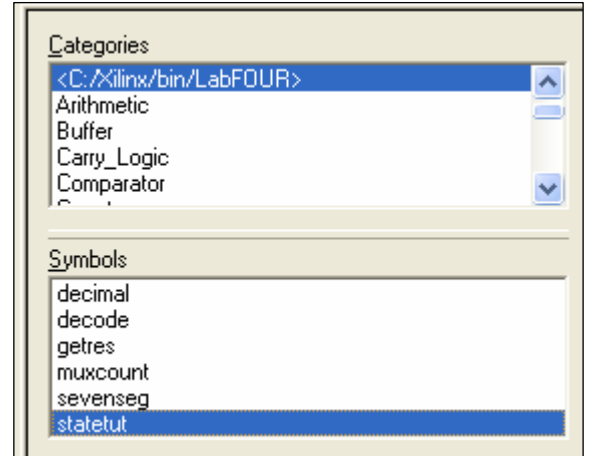
4. In the **Processes for Source** window, select **Create Schematic Symbol**. *The question mark next to the line will change to a check if done properly.*

5. Open the schematic you wish to place the symbol in.

6. **Select the symbols tab** under the **sources in project** window and **expand** the window using the **arrow** in the upper right corner.



- Under the **categories** list select the **folder and directory** you placed the Verilog file into.
- Under the symbols list, **select STATETUT**. You can now place the symbol inside your schematic.



- The symbol, by default, should look similar to the above.

For Further information try the XILINX tutorial at the link below

<http://tutor.al-williams.com/fsm-1.htm>