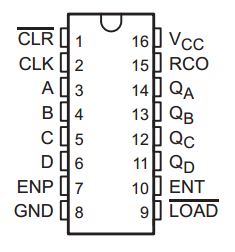
Task 2:

Download the Data Sheets of the **SN74ALS163 Synchronous 4-Bit Binary Counter** from Texas Instruments' Web site. Use the available **CAD tools to design, simulate and compile a functionally** equivalent circuit on your **Xilinx Spartan-6 chip**. However, downloading the bit file to your Nexys 3 Board is **NOT** required.

# Algorithm of SN74ALS163 Synchronous 4-Bit Binary Counter

The task elaborates the functionally equivalent circuit of SN74ALS163 Synchronous 4-Bit Binary Counter. To execute this task, we have some inputs and outputs. Look the figure below.



**SN74ALS163**

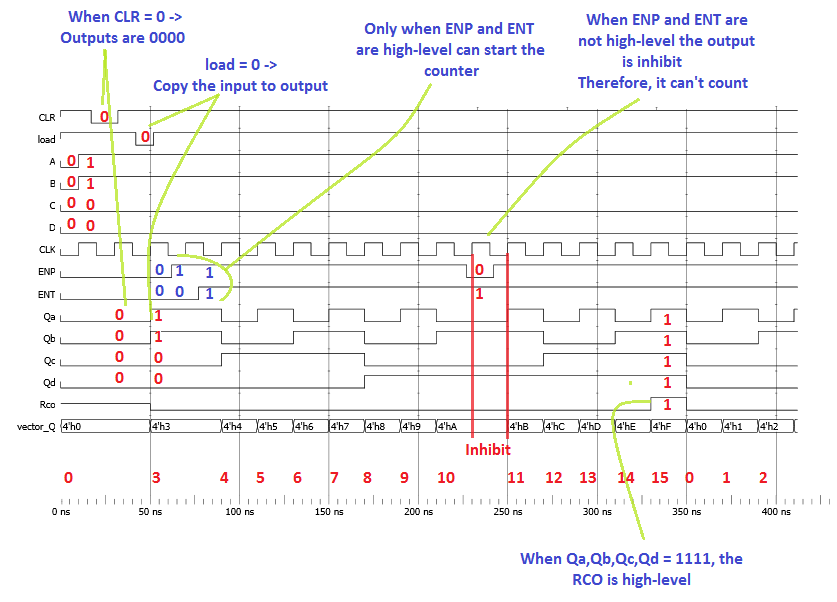
The counter can be present to any number between 0 and 15. Furthermore, it can start with any number between these numbers. It is possible because we can set up a low level at the load (LOAD). Consequently, the initial count will be the value in the input A, B, C, D. The CLR input when active in low can clear the count to 0000. Also, we have the inputs ENP and ENT that are conditional to count because both must be high to count and ENT is a conditional to enable RCO. RCO, always produces a high-level pulse while the count is 15.

I developed a finite state machine to reproduce the functionally equivalent circuit of SN74ALS163. To test the operation, I created a file .do to test the correct operation.

* I checked the CLR. When it is the low-level, the output must be 0000;
* I checked the LOAD. When the load is in the low-level, the input: A, B, C, and D is copied to the output: Qa, Qb, Qc, and Qd.
* I checked the RCO because it should be high-level when the output is 1111 = “15”.
* I checked the inputs: ENP and ENT because the counter just can work when the ENP and ENT if high-level. In case it doesn’t happen, the output cannot change.
* Finally, checked the all counter states.

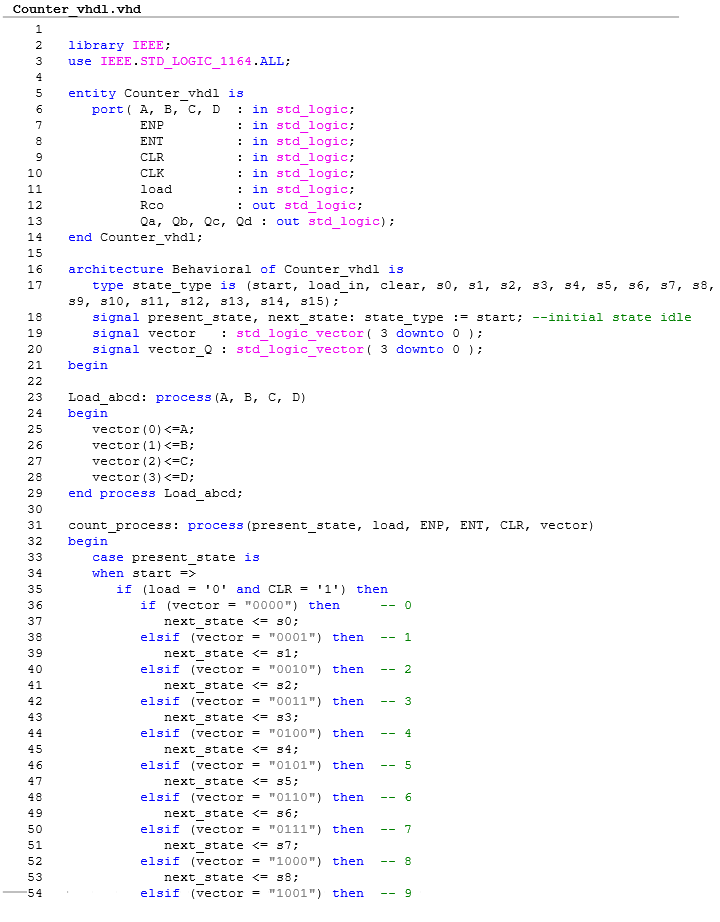
# Simulation - Wave

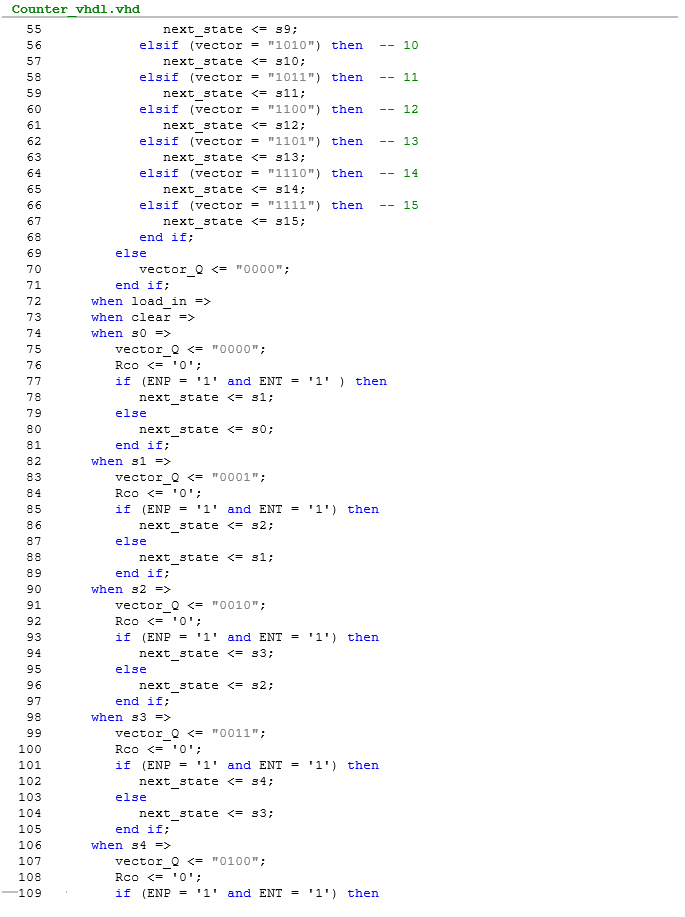
Look the wave below:

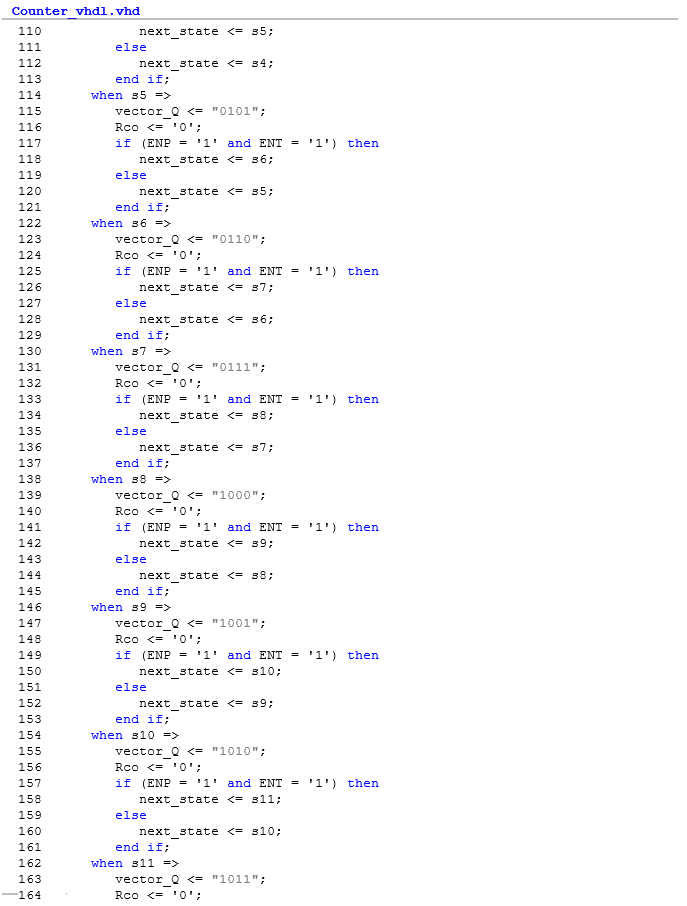


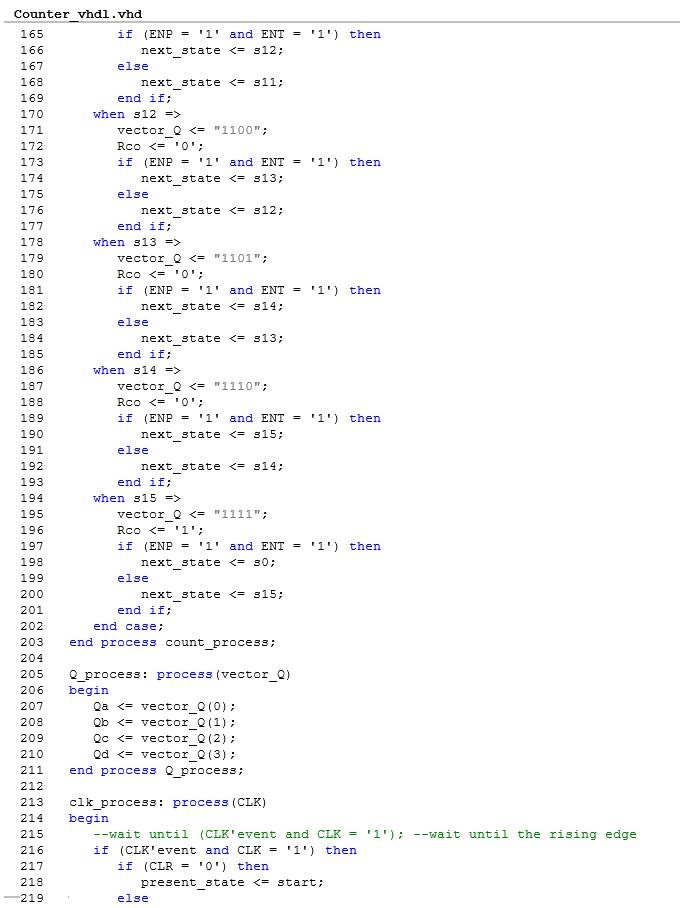
# 

# File .vhd





****

****

# 

# Simulation: File .do

**####################################**

**# Macro for the SN7ALS163 #**

**####################################**

restart

# Generates de clock with T = 20ns

force CLK 0 0, 1 10ns -r 20ns

**# Initial Information**

force A 0

force B 0

force C 0

force D 0

force ENP 0

force ENT 0

force load 1

force CLR 1

**# Initial condition for A, B, C, D**

run 10ns

force A 1

force B 1

force C 0

force D 0

**# Clear test**

run 7ns

force CLR 0

run 15ns

force CLR 1

**# Load of the initial condition**

run 10ns

force load 0

run 10ns

force load 1

**# Test of the inputs ENP and ENT**

run 10ns

force ENP 1

run 15ns

force ENT 1

**# Maintain the count**

run 150ns

# Test the Inhibit

force ENP 0

run 15ns

force ENP 1

run 170ns

# File .ucf

# PlanAhead Generated physical constraints

NET "A" LOC = T10;

# PlanAhead Generated IO constraints

NET "A" IOSTANDARD = LVCMOS33;

# PlanAhead Generated physical constraints

NET "B" LOC = T9;

# PlanAhead Generated IO constraints

NET "B" IOSTANDARD = LVCMOS33;

# PlanAhead Generated physical constraints

NET "C" LOC = V9;

# PlanAhead Generated IO constraints

NET "C" IOSTANDARD = LVCMOS33;

NET "CLK" IOSTANDARD = LVCMOS33;

# PlanAhead Generated physical constraints

NET "CLR" LOC = M8;

NET "D" LOC = N8;

NET "ENP" LOC = U8;

NET "ENT" LOC = V8;

NET "load" LOC = T5;

# PlanAhead Generated IO constraints

NET "load" IOSTANDARD = LVCMOS33;

NET "ENP" IOSTANDARD = LVCMOS33;

NET "ENT" IOSTANDARD = LVCMOS33;

NET "D" IOSTANDARD = LVCMOS33;

NET "CLR" IOSTANDARD = LVCMOS33;

# PlanAhead Generated physical constraints

NET "Qa" LOC = U16;

NET "Qb" LOC = V16;

# PlanAhead Generated IO constraints

NET "Qa" IOSTANDARD = LVCMOS33;

NET "Qb" IOSTANDARD = LVCMOS33;

NET "Qc" IOSTANDARD = LVCMOS33;

NET "Qd" IOSTANDARD = LVCMOS33;

NET "Rco" IOSTANDARD = LVCMOS33;

# PlanAhead Generated physical constraints

NET "Qc" LOC = U15;

NET "Qd" LOC = V15;

NET "Rco" LOC = M11;

NET "CLK" LOC = C9;