**CSC 343 Spring 2021**

## Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

## *What to Submit:*

Please post on direct private channel to Instructor just Figures with **SCREENSHOTS** of waveforms. The waveforms should demonstrate the correctness of storage operation, asynchronous set, and clear of storage elements described in Part1,2,3. Figure Captions should state **IN ONE SENTENCE** why the design is correct. The file name has to have your last name and title, signals have to have your last name as a prefix.

***No report,*** nor video is required to submit.

***No grade*** will be given for this Self-Check lab.

A Check Mark **will be assigned. The criteria used for Check Mark** (✓) A **check mark**, **checkmark** or **tick** (✓) is a **mark** used to indicate the concept "yes" (e.g. "yes; this has been verified", "yes; that **is the** correct answer", "yes; this has been completed".

This tutorial review has three parts:

## *What to DO:*

**Part 1.** **Using D Latch Component, simulation and verification with Model-Sim**

**Part 2.** **Design Master-Slave D-FF using D-Latches and Simulation with Model-Sim**

**Part 3.** **Using D Flip Flop As A Component Symbol and Simulation with Model-Sim**

**CSC 343 Spring 2021**

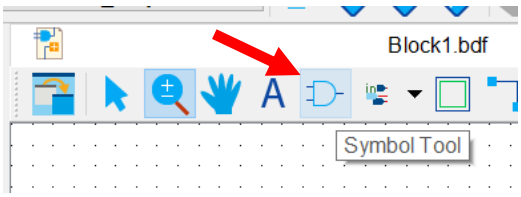
# Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

## Part 1. Using D Latch Component, simulation and verification with Model-Sim

The **D Latch** is a primitive storage device that utilizes an internal feedback mechanism to "remember" previous inputs. We will explore its behavior in detail, later in the guide. For now, find it in the symbol tool menu and insert it into a block diagram.



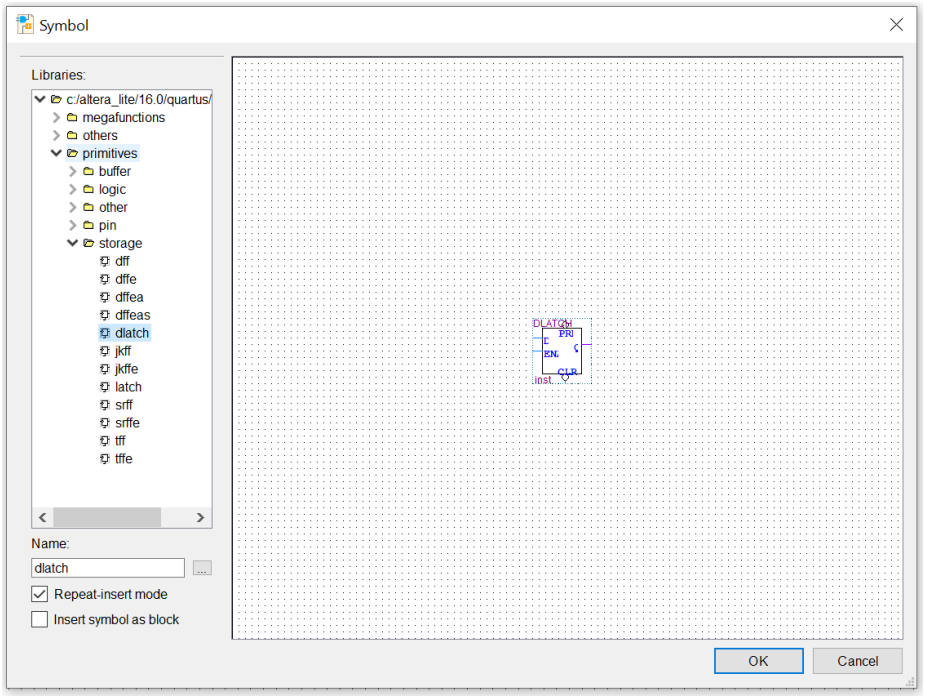



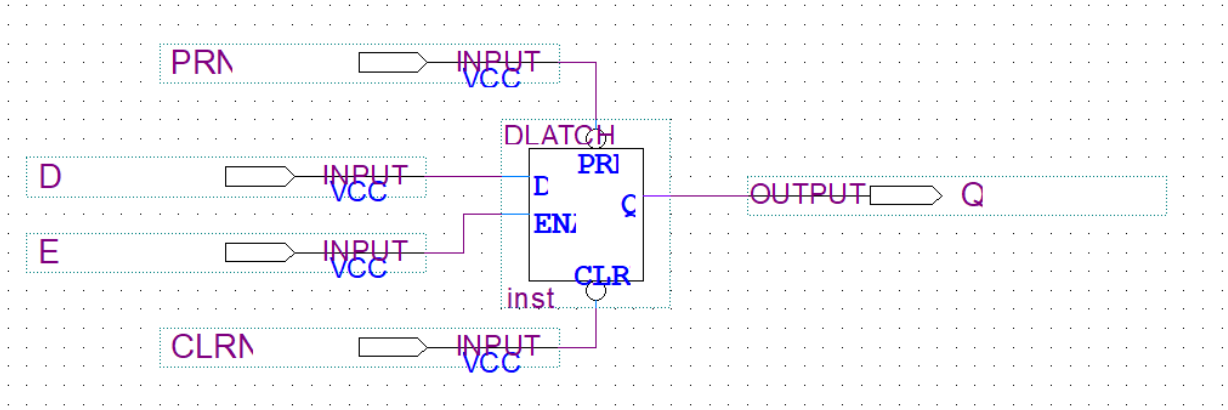

**Figure 1.1. D-Latch symbol component.**

Create 4 inputs and 1 output, and set the diagram up something like this.

# Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

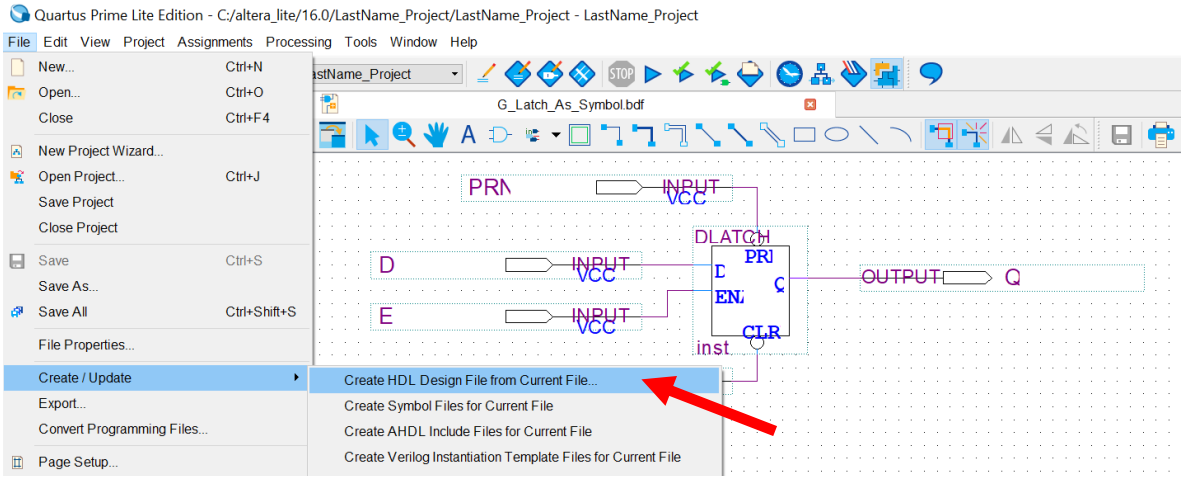



It may be complicated at first glance. Essentially, the enable input **E** acts as a switch that allows the output to be affected by **D**, the data input.

When **E** is at value 0, the value of **Q** will not move regardless of changes in the **D** input. When **E** is at value 1, **Q** will mirror the value of **D** in real-time. The moment of interest here is when **E** changes from 1 to 0 – at that moment, **Q** will freeze its value until **E** becomes 1 again. In the ModelSim simulation we will see all this happen.

**CLRN** and **PRN** are neat additions to any device like this. They come with many of the built in symbols and they are used to reset the output **Q**. **CLRN** resets the output to low (0), while **PRN** resets the output to high (1). Note these switches don't depend on **D** or **E**, at all. We will check this out in ModelSim later too.

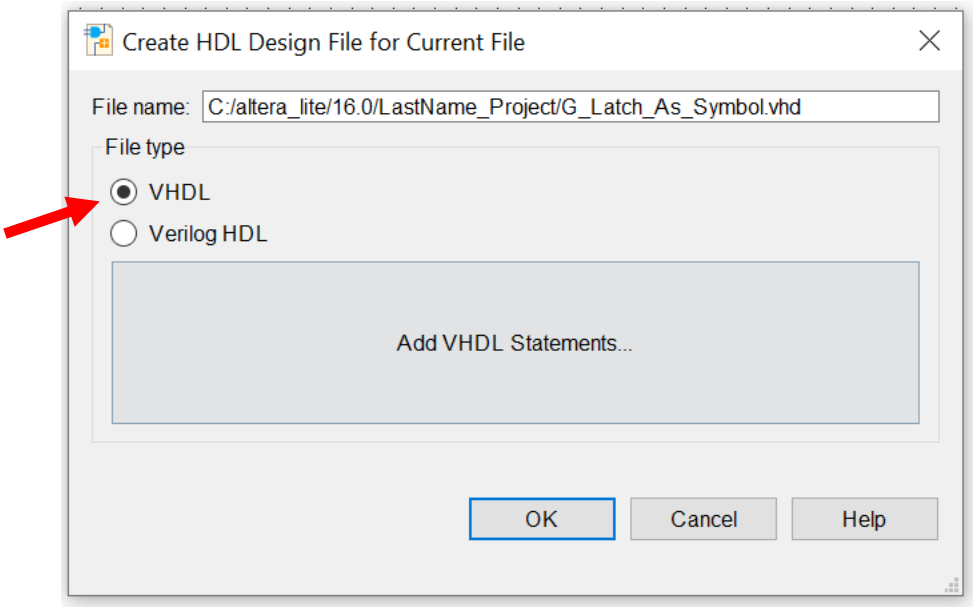
You may compile this block, but we will create VHDL from this diagram anyway.



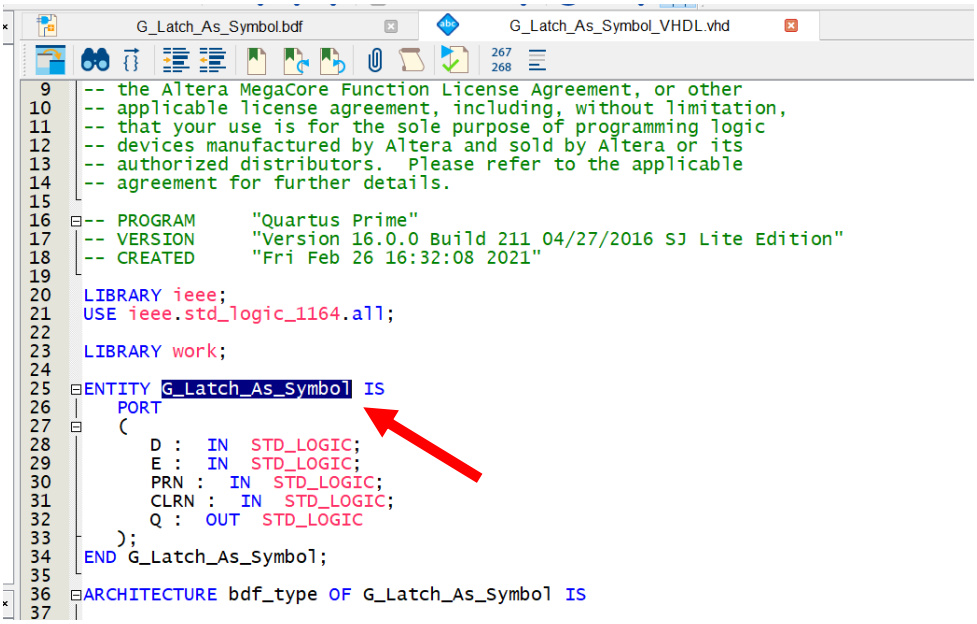
# Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***




Like before, we are unable to change the output file's name and this will cause a compile issue since the block diagram has the same name. I recommend to find the file and rename it manually. Once it is renamed, open it and notice the **ENTITY** name hasn't changed.

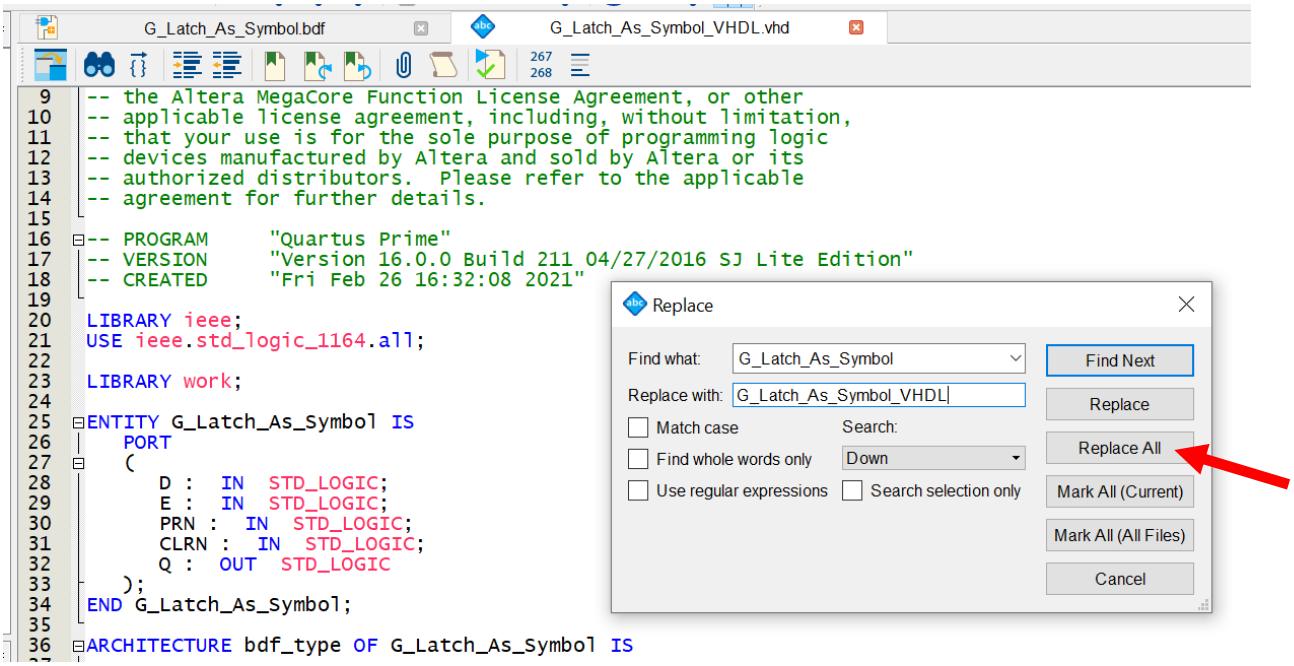



## Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

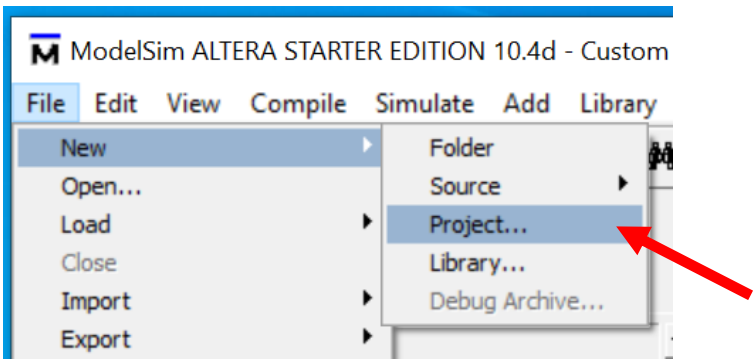
We will use the same trick as last time; find the feature **Edit > Replace…** and **Replace All** instances where the name is wrong.




You can set it as **Top-Level Entity** and compile to make sure nothing has gone wrong, however if you are confident it's correct we will compile it in ModelSim anyway.

## Wave Simulation in ModelSim

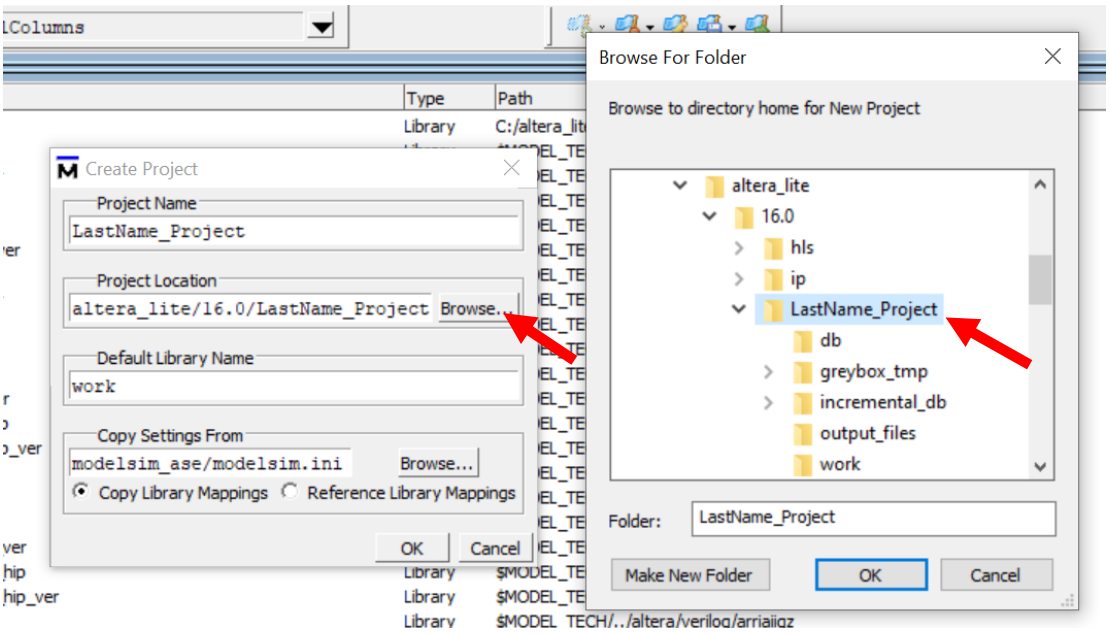
In ModelSim, create a new project. **Browse…** for the directory of the D Latch VHDL file, add it as an existing file, etc. as in previous guides. Then compile the D Latch VHDL file.

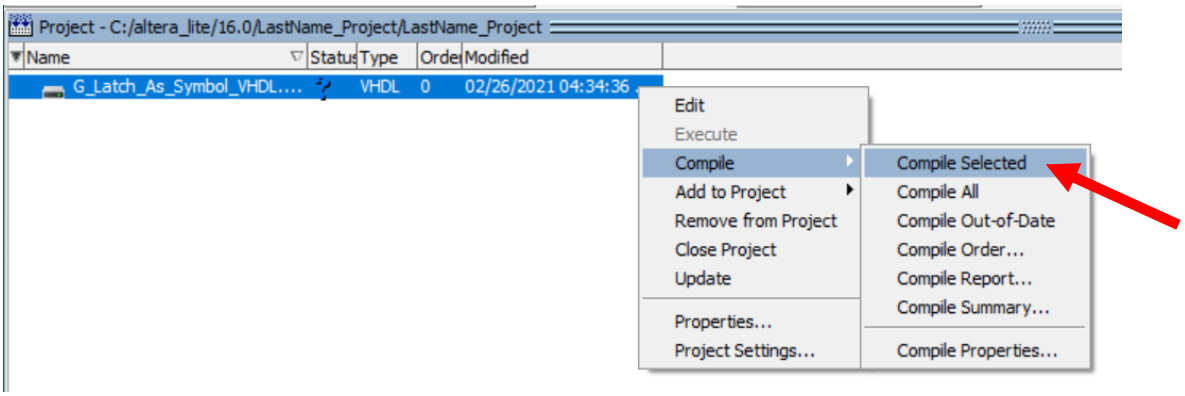


# Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***



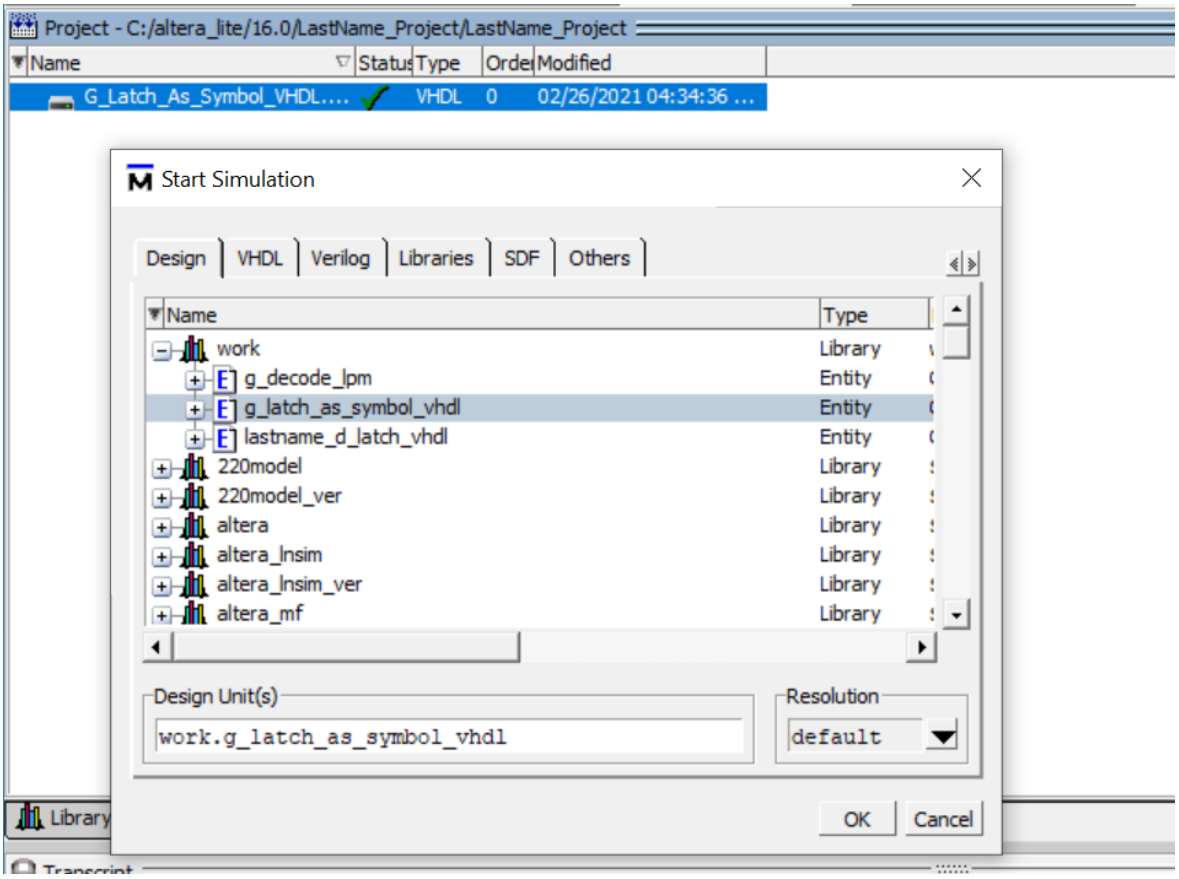



## Self-Check Laboratory Exercise 4A

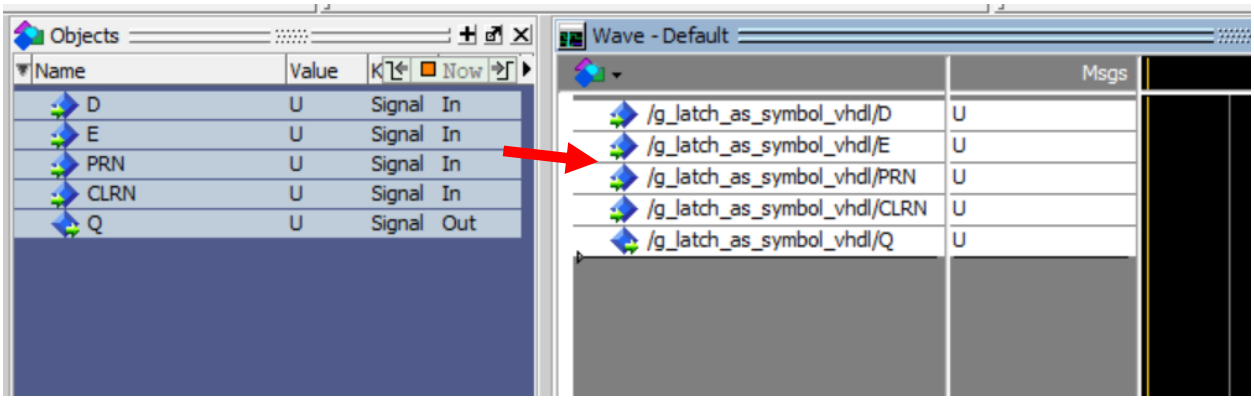
Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

Then start the simulation with **Simulate > Start Simulation.** Select your D Latch VHDL file.




As always, drag the non-Internal signals into the wave tab.

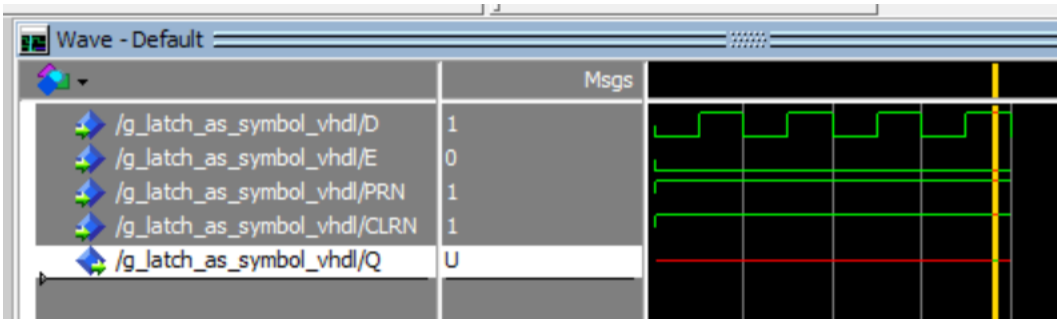


## Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

For simplicity, in the first two simulations I simply had **E** off one time, and on the next. I forced **CLRN** and **PRN** to stay at 1 so they do not affect the output.




We observe that **Q** remains in an Undefined state (red line).

Now with **E** forced to 1.


We observe that **Q** follows the value of **D**.

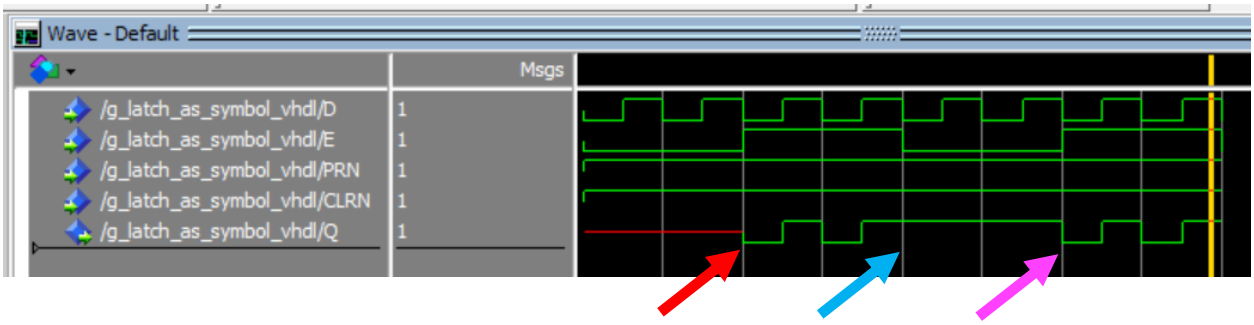
More simulations on the next page.

## Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

In this third simulation, I used the **Clock** feature to keep **E** and **D** oscillating at different rates.

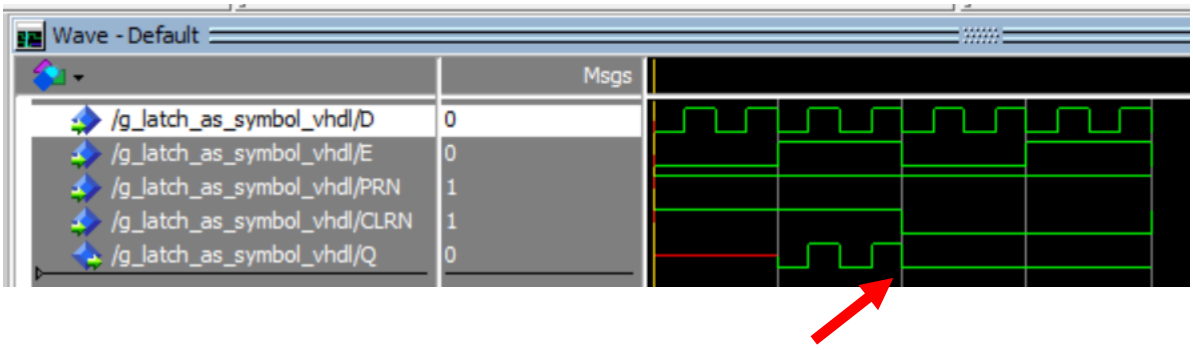



At the time of the red arrow, **E** first moves from 0 to 1. **Q** leaves the Undefined state, and begins mirroring **D**.

At the time of the blue arrow, **E** moves from 1 to 0. We observe that the entire time **E** remains at 0, **Q** holds its value and does not enter an Undefined state.

At the time of the pink arrow, **E** moves from 0 back to 1. We observe that **Q** leaves its "latched" state (frozen value) and begins to mirror **D** again.

I repeated this third setup, this time with **CLRN** going from 1 to 0 halfway through the simulation.




We observe that **Q** drops to 0 and stays there (as long as **CLRN** is still 0). If we had done this same procedure with **PRN**, then **Q** would jump to 1 instead.

## Part 2. Design Master-Slave D-FF using D-Latches and Simulation with Model-Sim (Tutorial Review)

We can create our own **D Flip Flop** from D Latches. This kind of technique is called **Master-Slave**, it summarizes the relationship between the 2 latches used.

# Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

Start off by adding a blank block diagram schematic to your project. Create 2 of the built-in symbol **DLatch**, a **not** gate, 4 inputs, and 1 output. You can find the **DLatch** in the symbol menu like we did last time.

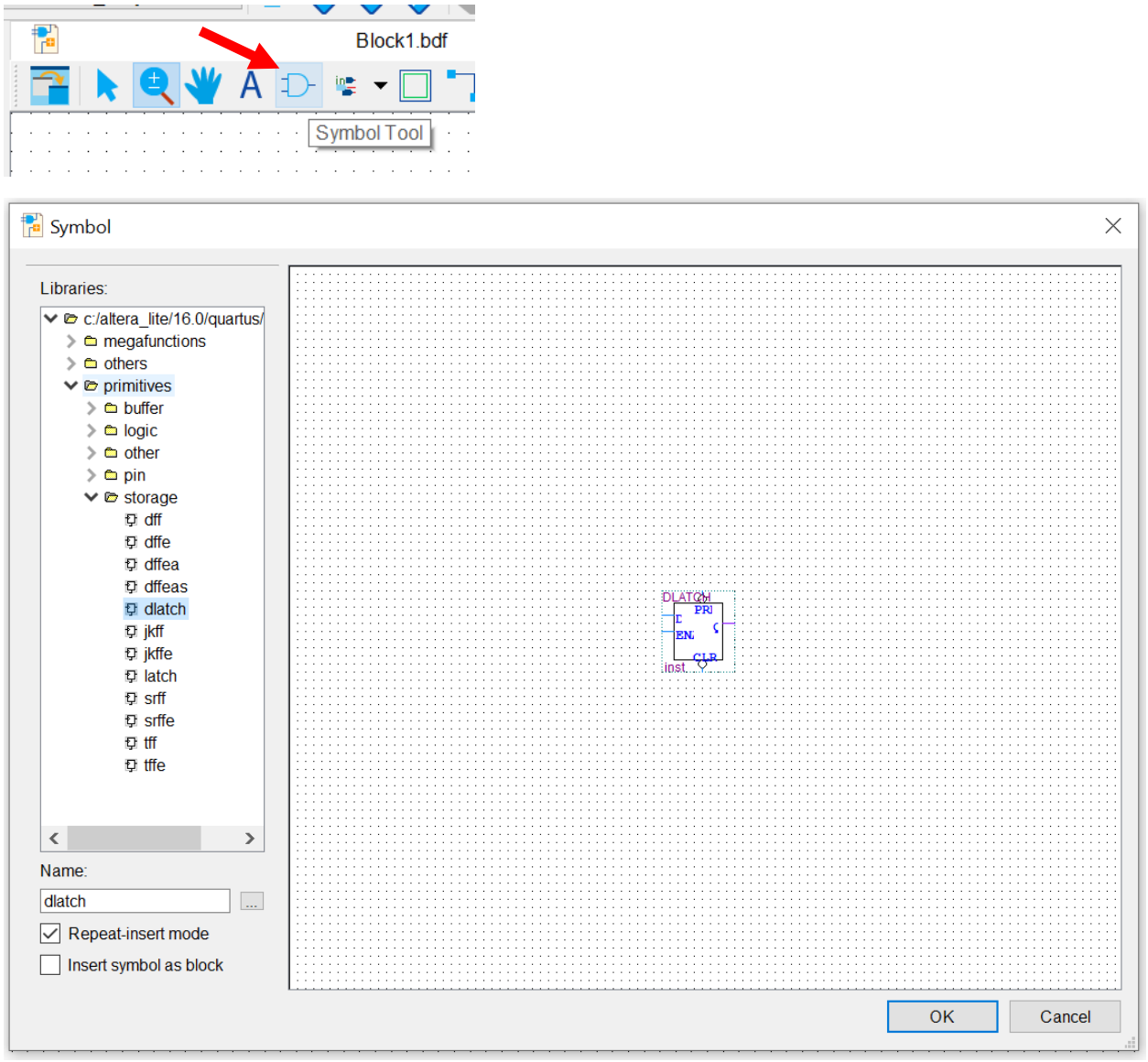



**Figure 2.1. D-Latch symbol component.**

Connect the pieces as such, notice that the **CLK** is inverted for the first latch but not the second one. This yields a positive-edge triggered response, by inverting the second latch's clock signal instead we would have a negative-triggered one.

# Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

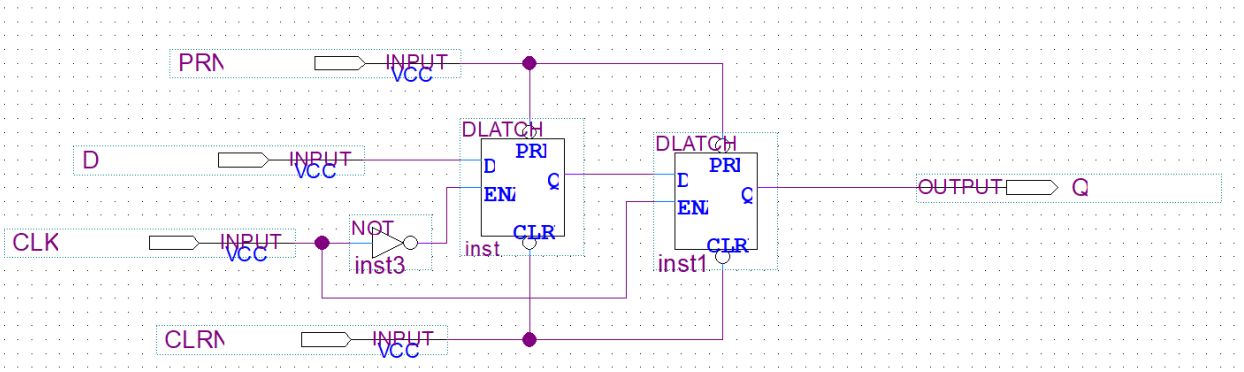
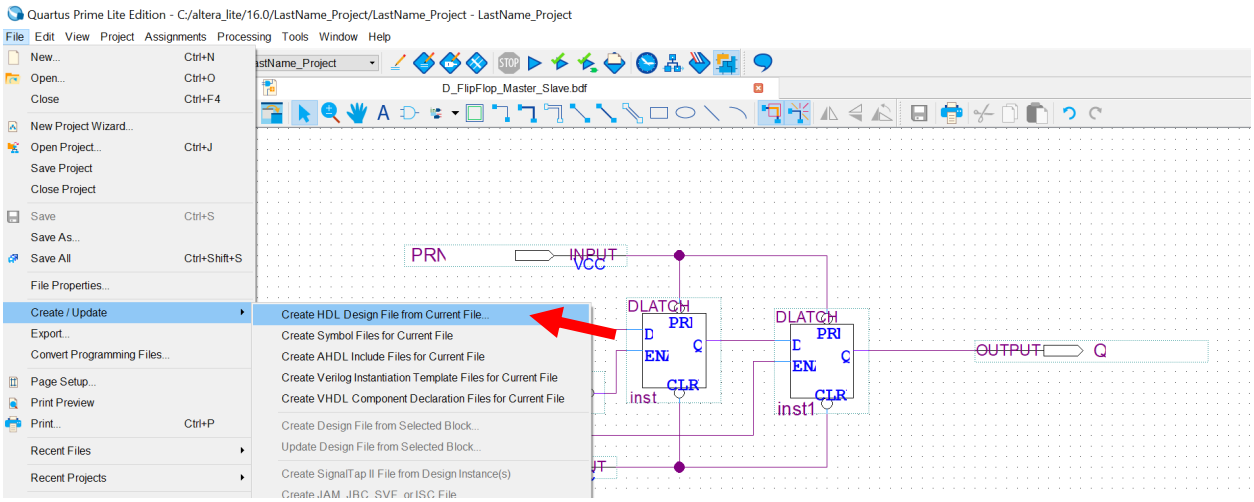



**Figure 2.2. Schematics of Master-Slave D-FlipFlop.**

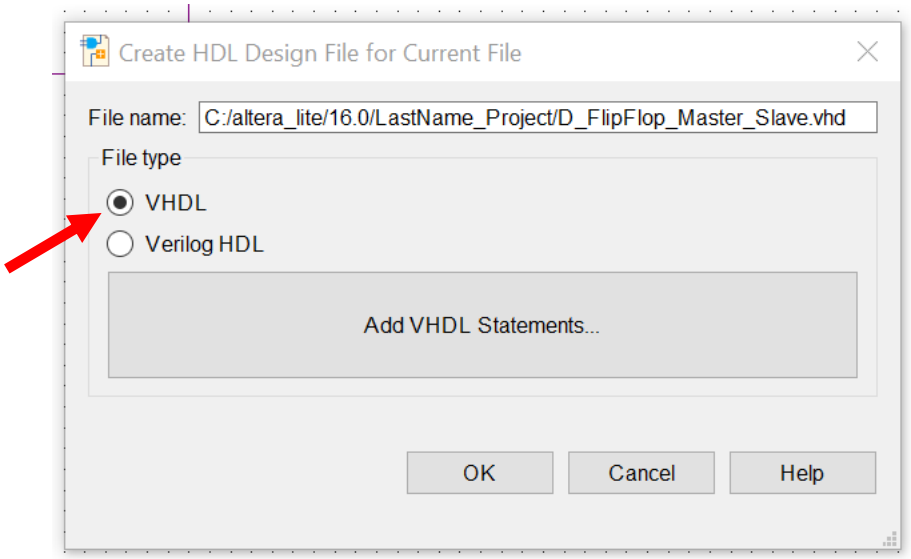
Create VHDL from this design.



# Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

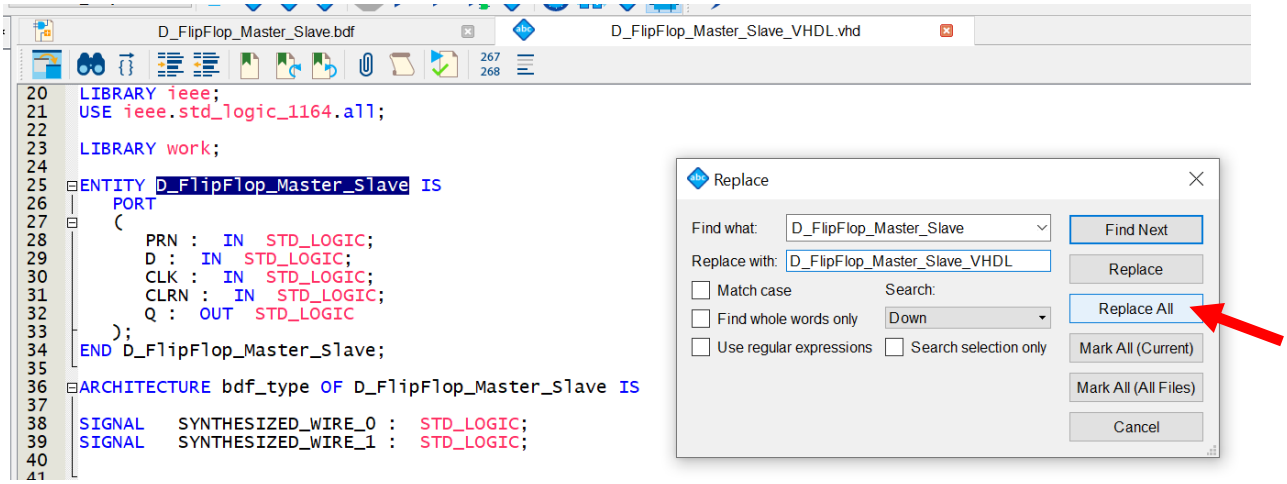
***Instructor:*** ***Professor Izidor Gertner***




As usual we cannot change the file name, and this will lead to a compile error later since the block diagram has the same name. I recommend to use our usual trick of manually renaming the VHDL file then opening it and replacing all instances where the filename is still the old one.

First, find the folder of your project and manually rename the file.


Then, open the file in Quartus (make sure **Add file to current project** is ticked). Find the feature **Edit > Replace…** and **Replace All** instances where the name is wrong.



## Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

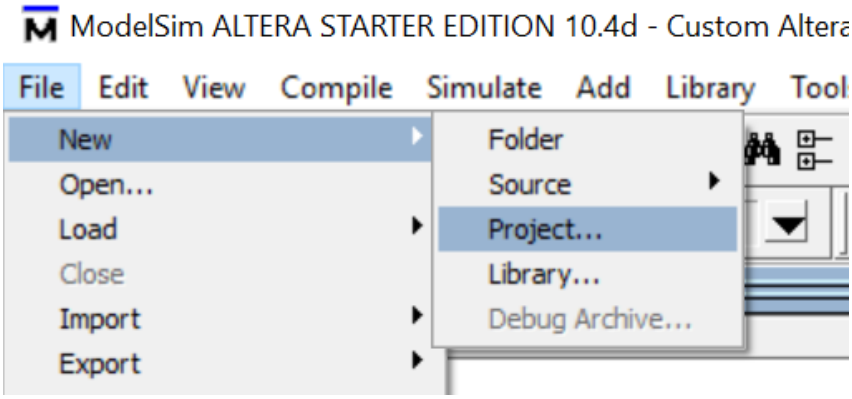
***Instructor:*** ***Professor Izidor Gertner***

You can compile the file but we will do it in ModelSim.

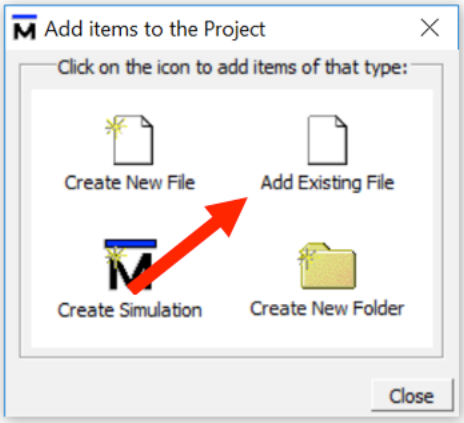
# ModelSim Simulation

Follow the usual steps in ModelSim (project folder select, add existing file).

First, if the project is not set up yet, create a new one and for the **Project Location**, you have to **Browse** to the folder where the VHDL file we created is.




Then add that file to the project with **Add Existing File**

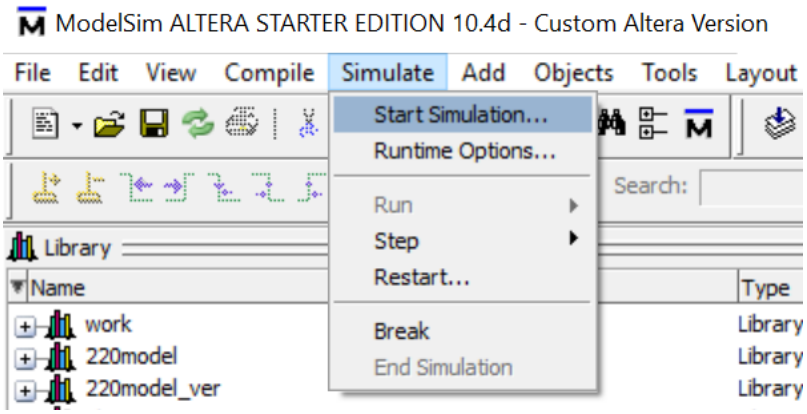


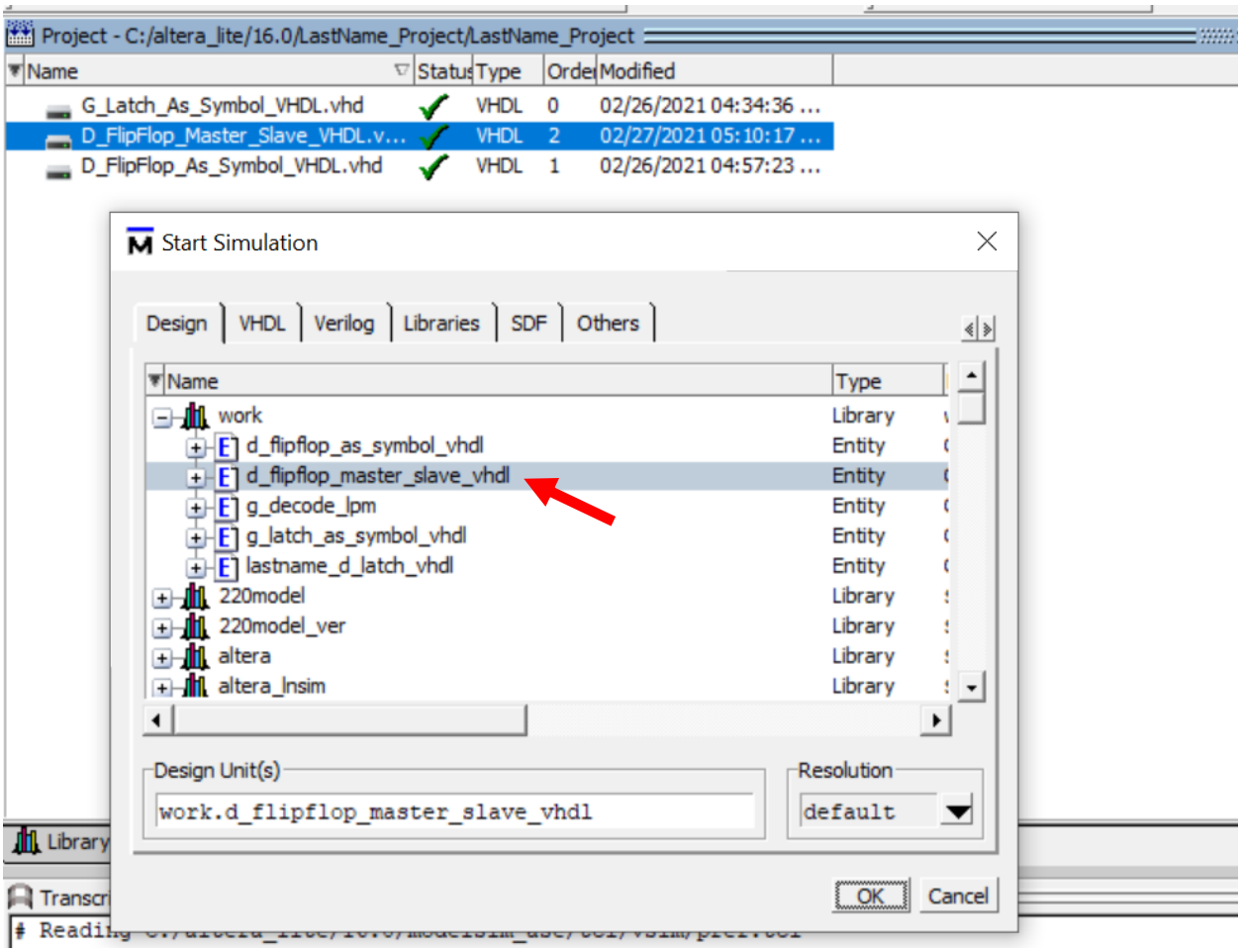
## Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

Start the simulation and select the file.



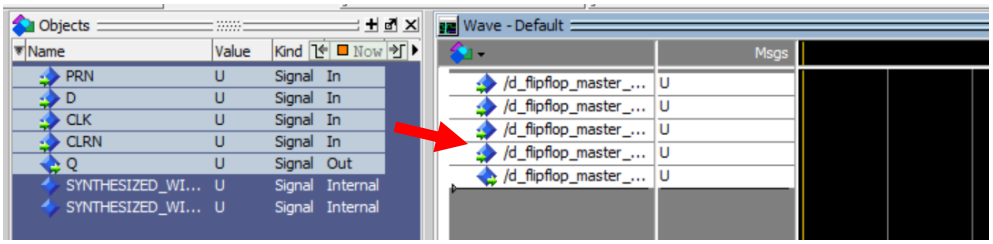



# Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

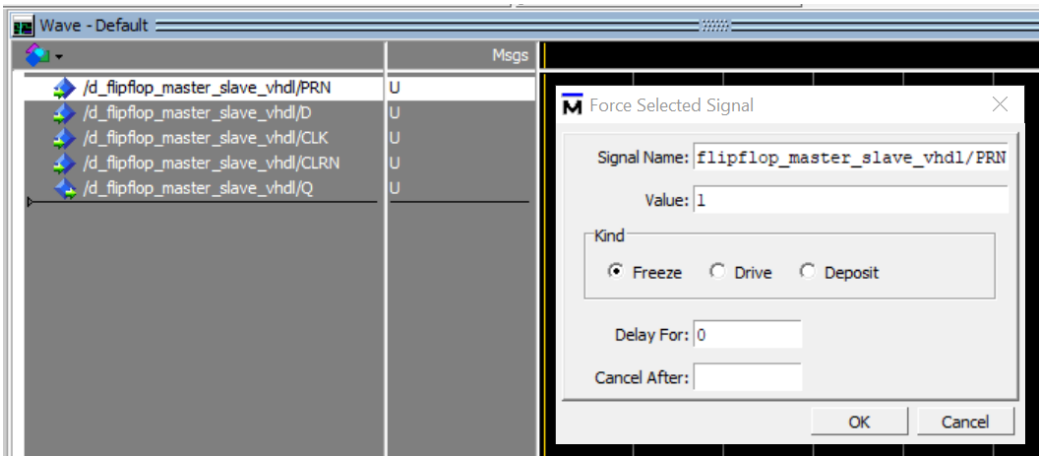
***Instructor:*** ***Professor Izidor Gertner***

As always drag the non-Internal signals to the wave.

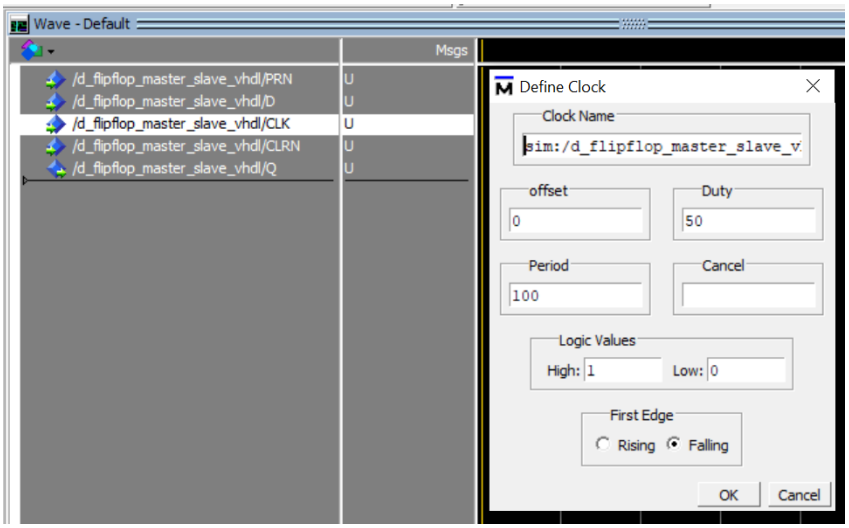



To observe what we need to I set the signals up like this.

**PRN** I forced at 1 so it doesn't interfere. I did the same for **CLRN** too.




I kept the clock at its default settings, but with a falling first edge.

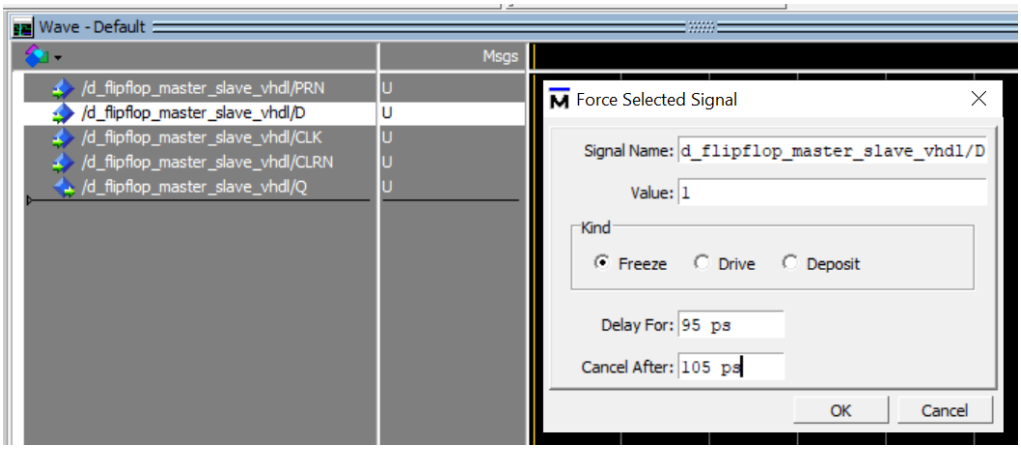


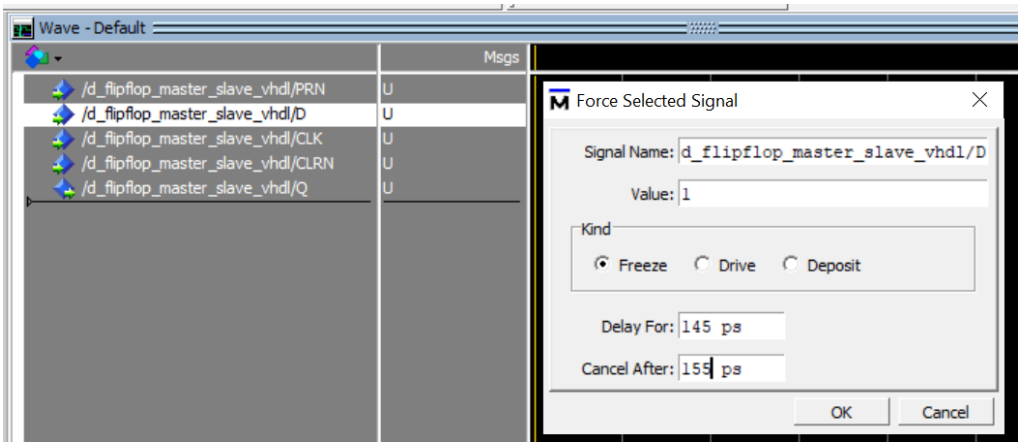
# Self-Check Laboratory Exercise 4A

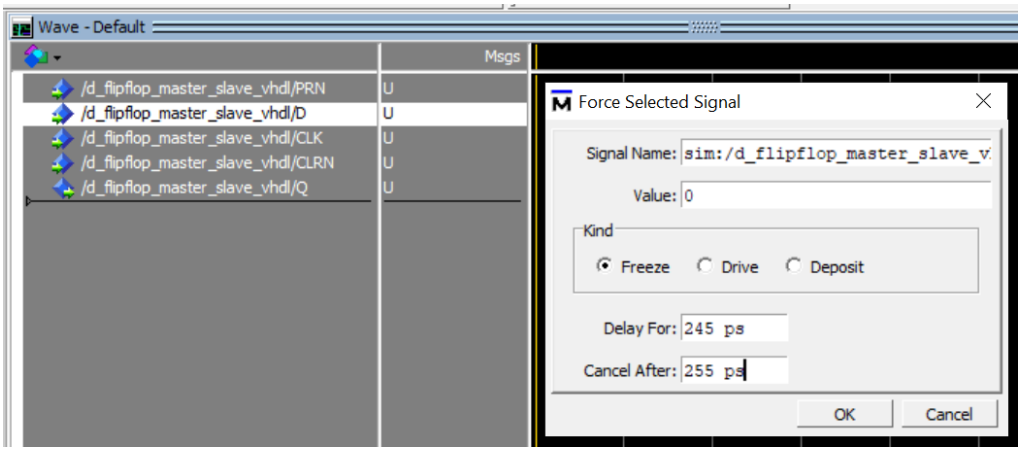
Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

I set up three different forces at different times for the data input. They are surrounding the clock edge – the first one is an attempt on a falling clock edge (1 to 0), the other two are attempts on rising clock edges (0 to 1).



## Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

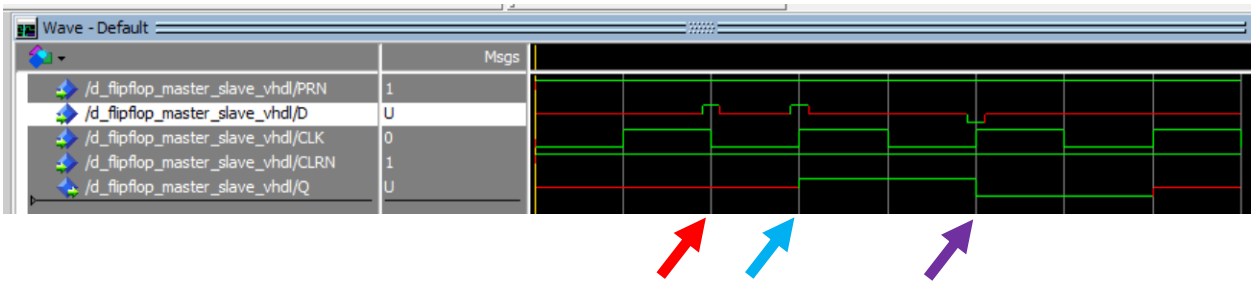



**Figure 3. Simulation of Master-Slave D-FlipFlop.**

**PLEASE SUBMIT JUST THIS WAVEFORM WITH YOUR LAST NAME AS A PREFIX TO EACH SIGNAL.**

We observe that this design behaves exactly like the D Flip Flop we made from the built-in component symbol.

At the time of the red arrow, there is no change in output **Q** because we designed this flip flop to be positive-edge triggered.

At the times of the blue arrow and purple arrow, we see that the value of **D** is reflected in the output as the clock rises from 0 to 1.

## Part 3

### Using D Flip Flop As A Component Symbol and Simulation with Model-Sim

The **D Flip Flop** is another primitive storage device. Unlike a latch, a flip flop circuit is edge-triggered. What this means is, there is no **E** (enable) input that indicates when the output **Q** should mirror the value of **D**. Instead there is an oscillating clock input, that sets **Q** to the value of **D** at the moment the clock value changes.

Devices can be positive-edge triggered and negative-edge triggered, the names imply the functionality. This may be confusing to grasp in writing, but the behavior will quickly become clear when we run a few simulations later. For now find the symbol and drop it into a new block diagram.

# Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

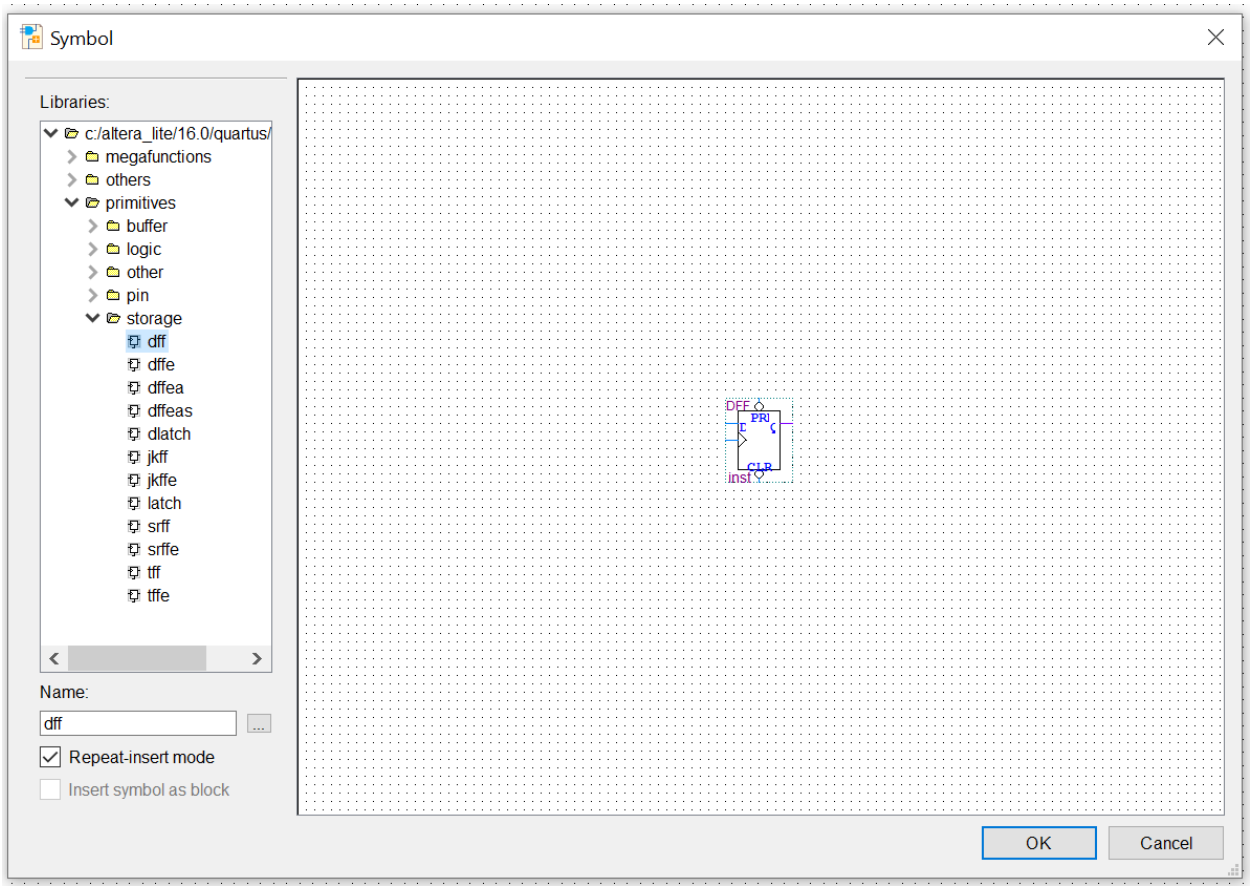


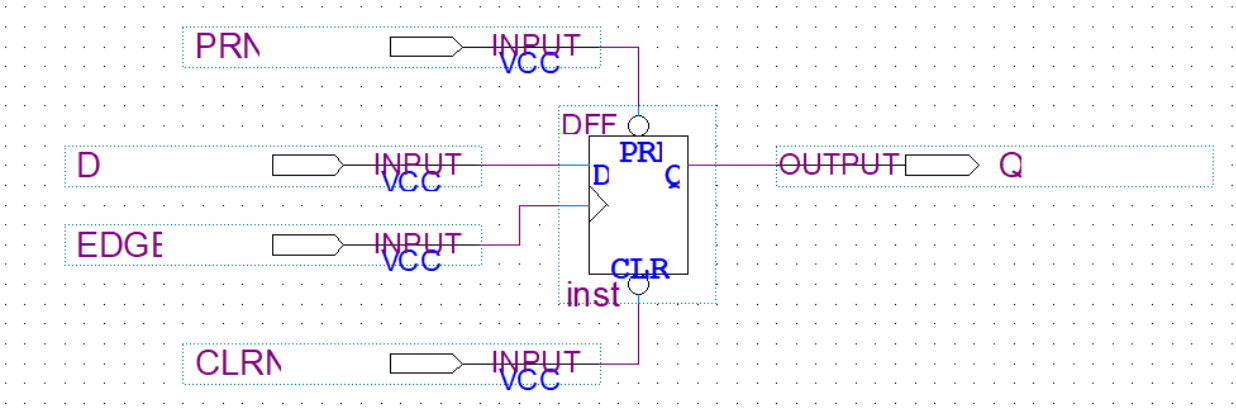

**Figure 3.1. D-FlipFlop symbol component. Please pay attention to triangle shape at the clock input.**

## Self-Check Laboratory Exercise 4A

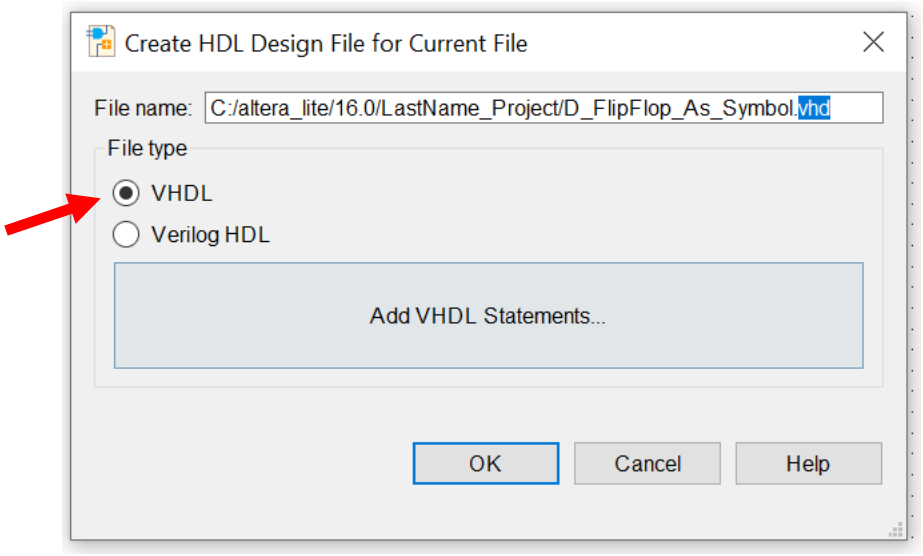
Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

Set it up similarly to the latch, except name the clock input something meaningful, perhaps **CLK**. I chose to name it **EDGE.**




As we did before create VHDL from this diagram.



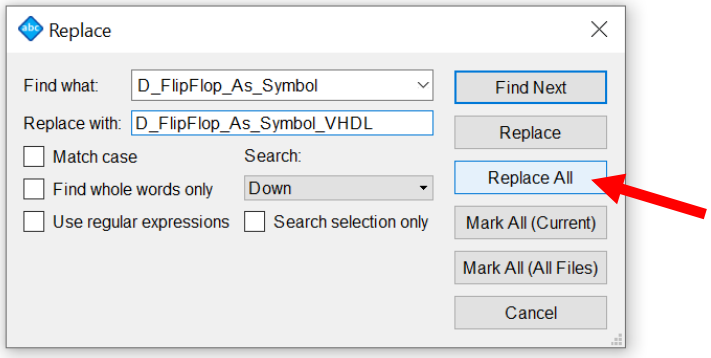

Once again we cannot change the file name, so we will use the usual trick with **Edit > Replace...** and **Replace All** to prevent the similar filenames compilation error.

## Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

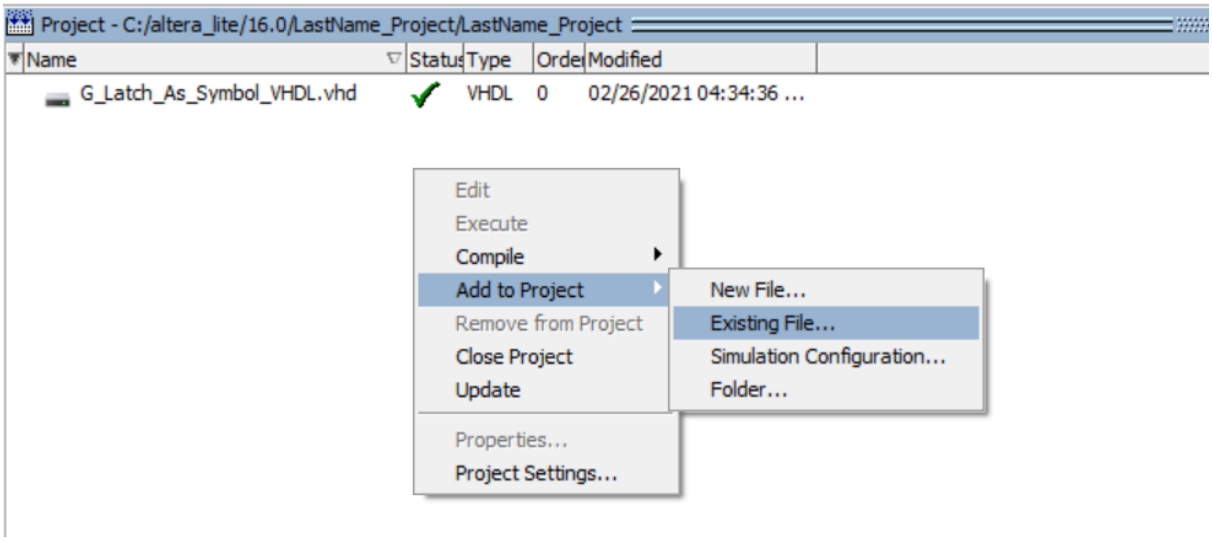
```
25  ENTITY D_FlipFlop_As_Symbol IS
26      PORT
27      (
28          CLRN :  IN  STD_LOGIC;
29          PRN :  IN  STD_LOGIC;
30          D :  IN  STD_LOGIC;
31          EDGE :  IN  STD_LOGIC;
32          Q :  OUT  STD_LOGIC
33      );
34  END D_FlipFlop_As_Symbol;
35
36  ARCHITECTURE bdf_type OF D_FlipFlop_As_Symbol IS
37
38
39
40  BEGIN
41
42
43
44  PROCESS(EDGE,CLRN,PRN)
45  BEGIN
46  IF (CLRN = '0') THEN
47      Q <= '0';
48  ELSIF (PRN = '0') THEN
49      Q <= '1';
50  ELSIF (RISING_EDGE(EDGE)) THEN
51      Q <= D;
52  END IF;
53  END PROCESS;
54
55
```




You can compile it but we will do that in ModelSim anyway.

## Wave Simulation in ModelSim

In this case I am still in the same ModelSim project as the latch. This is OK. Either way, add the flip flop as an existing file.

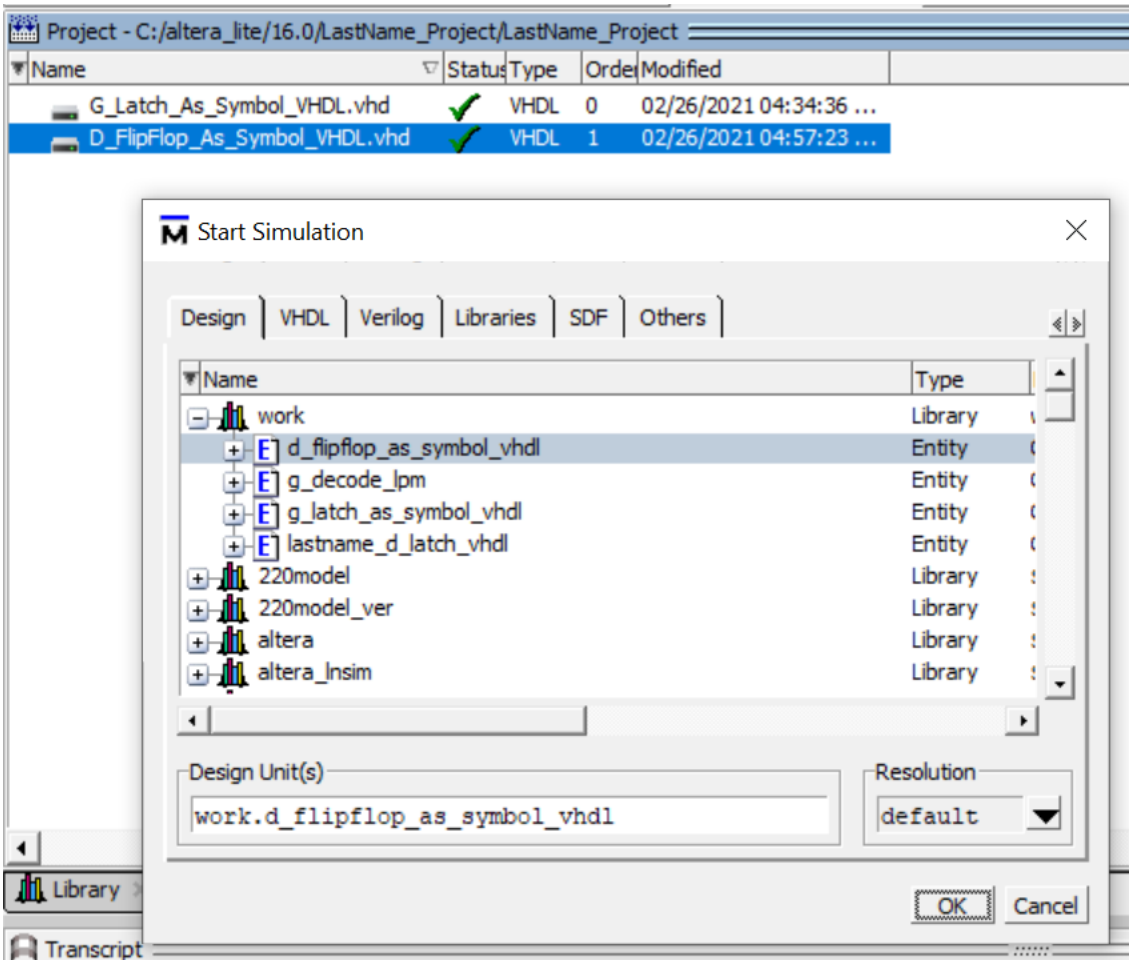


## Self-Check Laboratory Exercise 4A

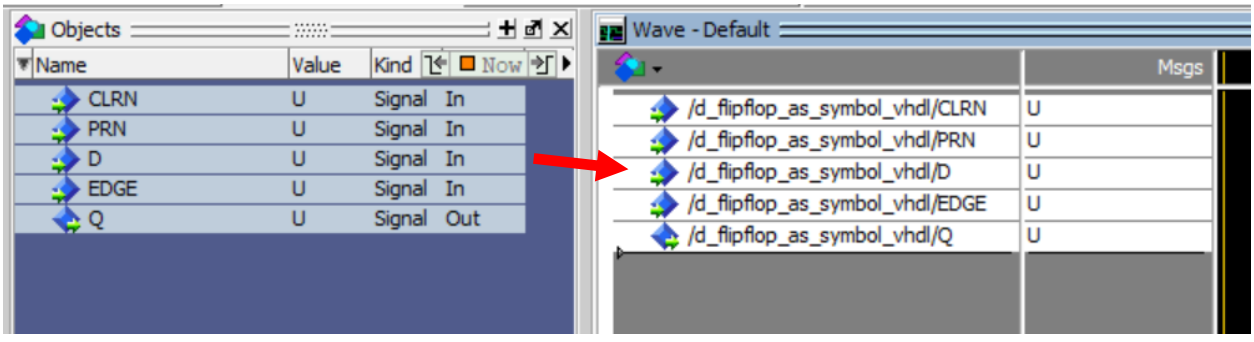
Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

Compile it (green check), then start the simulation.




Drag all the signals into the wave tab.



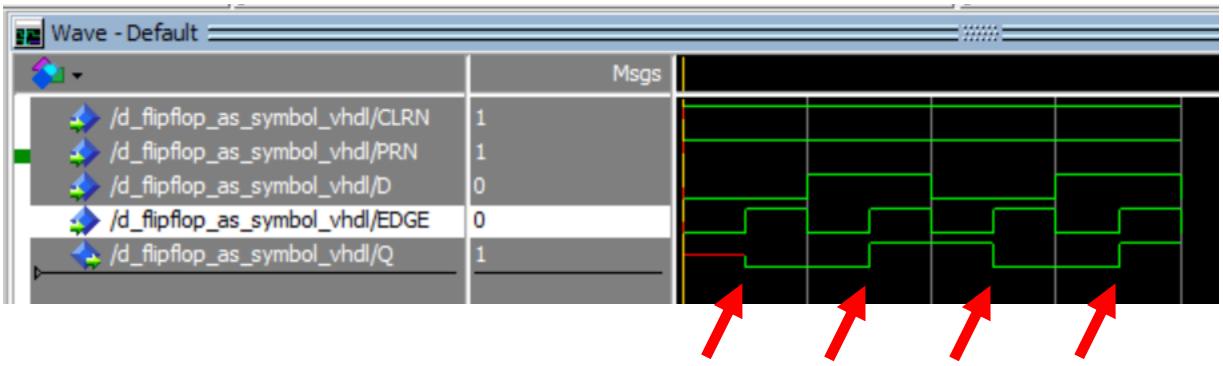
## Self-Check Laboratory Exercise 4A

Tutorial on D Latch, Master-Slave D-FF, D Flip Flop using components in Storage library.
Simulation and verification with ModelSim Simulation

***Instructor:*** ***Professor Izidor Gertner***

I forced **CLRN** and **PRN** to be 1 so they don't interfere with the simulation. They will behave predictably as I described earlier, but you can test them out on your own if you want.

I had the **EDGE** (clock) oscillating at twice the speed as **D**, this is an effective setup to observe the behavior.




At the time of the red arrows, the value of **Q** is set to **D**. Also notice at each arrow that **EDGE** (clock) is changing from 0 to 1. So, only when the **EDGE** (clock) is moving from 0 to 1, the value of **D** is transmitted to **Q.** This behavior can be summarized as, a **positive-edge triggered flip flop**.