## Lab 3: Designing the Toy Processor Datapath

**CSEE 4280 – ADVANCED LOGIC DESIGN**

**SPRING 2018**

## Pre-Lab Due (30 points): Jan 26 Friday 1:30 pm

***Report Due (70 points): Feb 1 Thursday 5 PM.***

***No Demo Due***

## Total Points: 100

## Contents

Part 1: Building an 8-bit Register (Part of Datapath – Instruction/Data Register)

Part 2: Let’s make a Counter! (Part of Datapath – Program Counter)

Part 3: Putting together the Datapath of the Toy Processor

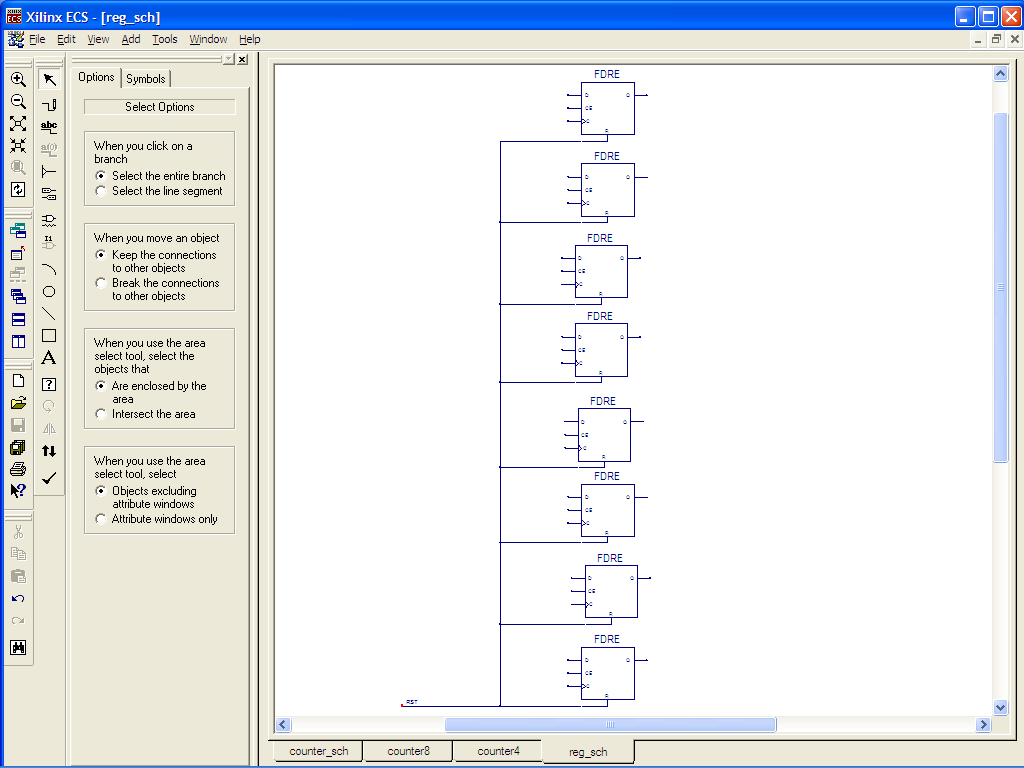
**Part 1**

**Building an 8-bit Register**

**Why do we do that? Because these registers will be your Instruction Register (IR), Data Registers including Accumulator Register in your complete datapath.**

We will build an 8-bit register using eight 1-bit registers. The 1-bit register symbol is called **FDRE**.

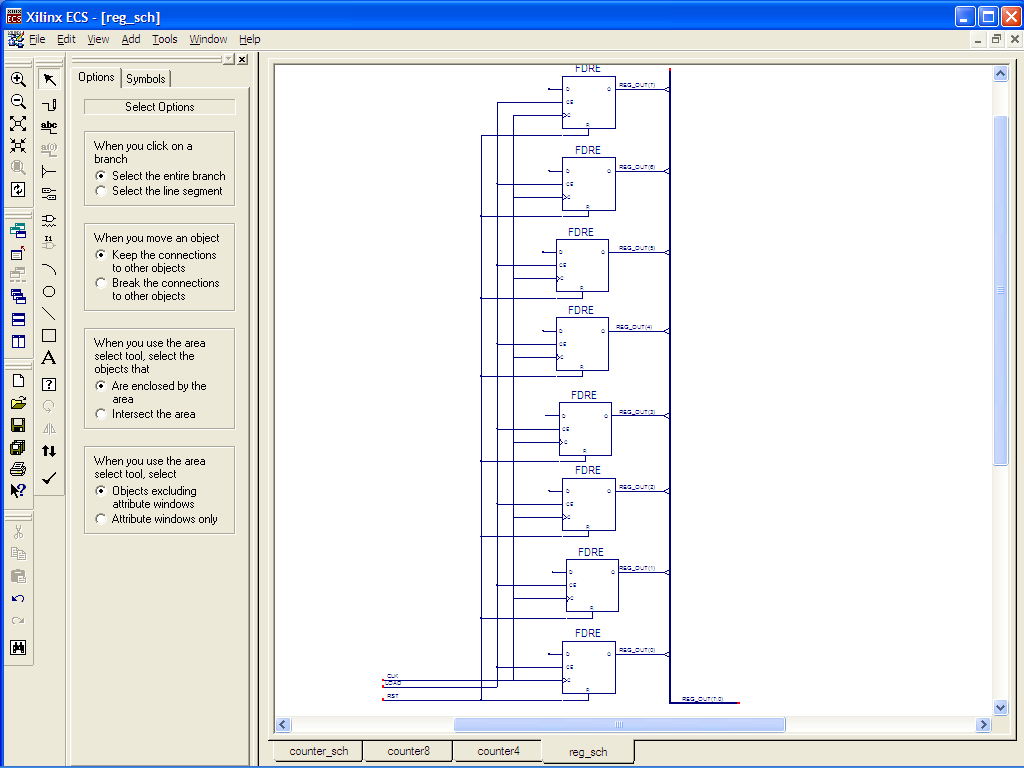
1. Create a new schematic and name it **reg\_sch.sch.**
2. On your drawing sheet, arrange eight FDRE symbols from the ***ISE*** library as shown below. Connect their R ports with a net. Name the net RST.  
   ***Note:*** *The FDRE symbol is made up of a D-flip-flop and a 2-1 mux. When CE is high, it loads the current value of D. If CE is low, D is ignored. If RST is high the FDRE is reset*

*.*   
  
*8 FDRE with R-ports connected to net RST.*

1. Similarly connect the CE pins to a single net named LOAD to allow for a common enable signal for all eight components. Do the same for the clock pins C. Name the clock net CLK.
2. Connect the output ports Q to a bus called REG\_OUT(7:0).   
   ***Note:*** *Remember that the bus tap nets must be named with the pattern REG\_OUT(7)…REG\_OUT(0).*

**REG\_OUT(7:0)**

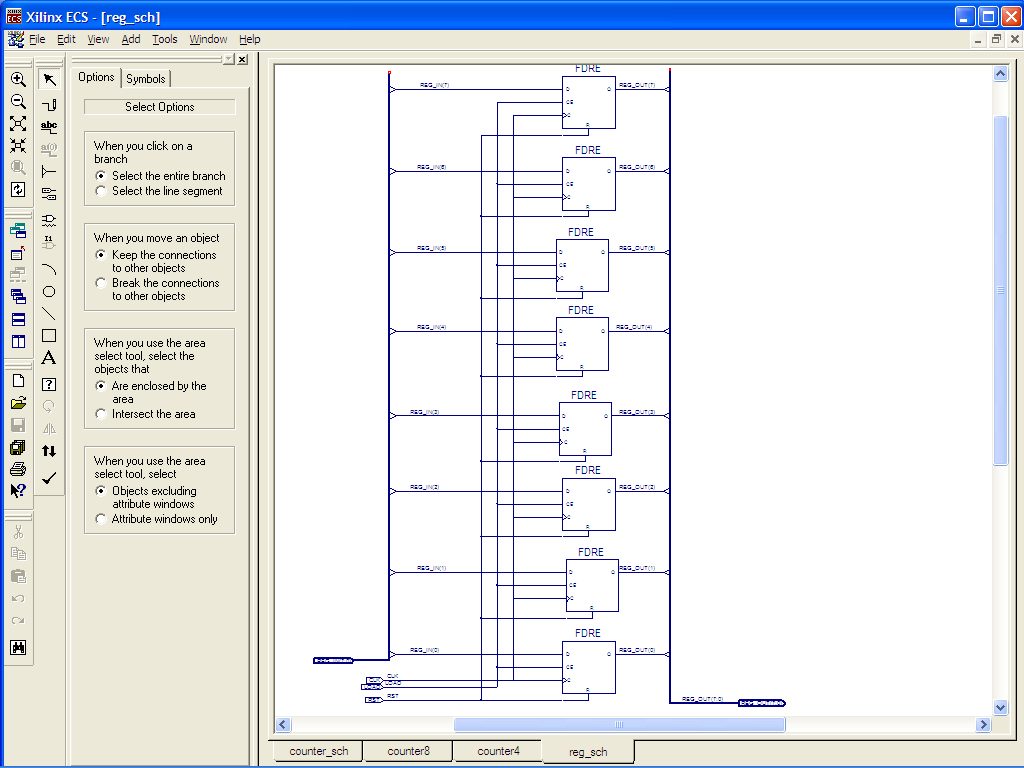
Your schematic should look something like the following now:

  
*REG\_OUT(7:0) bus connected, CLK, LOAD, RST connected  
.*

**CLK**

**LOAD**

**RST**

1. Connect the D ports in a bus named REG\_IN(7:0).
2. Add I/O ports to the schematic. CLK, RST, LOAD and REG\_IN(7:0) are all inputs. REG\_OUT(7:0) is an output. Your final schematic will look like this:  
     
   *Finished register with inputs CLK, LOAD, RST, REG\_IN(7:0) and output REG\_OUT(7:0)*

**CLK**

**LOAD**

**RST**

**REG\_IN(7:0)**

**REG\_OUT(7:0)**

**Creating a Test bench waveform**

1. Creat a testbench **reg\_tb.v**. If asked by Associate Source that “Select a source with which to associate the new source”, choose the **reg\_sch**.

***Please write your own testbench.***

***ISIM* : Running the simulation**

1. Choose the **Simulation** option on top of the **Design** window to enter into simulation mode.
2. In ***Behavioral window,*** highlight **reg\_tb.v.** The ***Processes for Current Source*** will display an expandable toolbox tree ***ISim* Simulator.** Expand ***ISim* Simulator** and double-click **Simulate Behavioral Model.** This will run the ***ISim***simulator process.   
     
   What is the expected output?

4. Does your register update REG\_OUT to the current REG\_IN only when LOAD is high? Does RST reinitialize all inputs and outputs? Make sure everything is working correctly. The results will be part of the lab report. Please keep all waveforms and testbench.

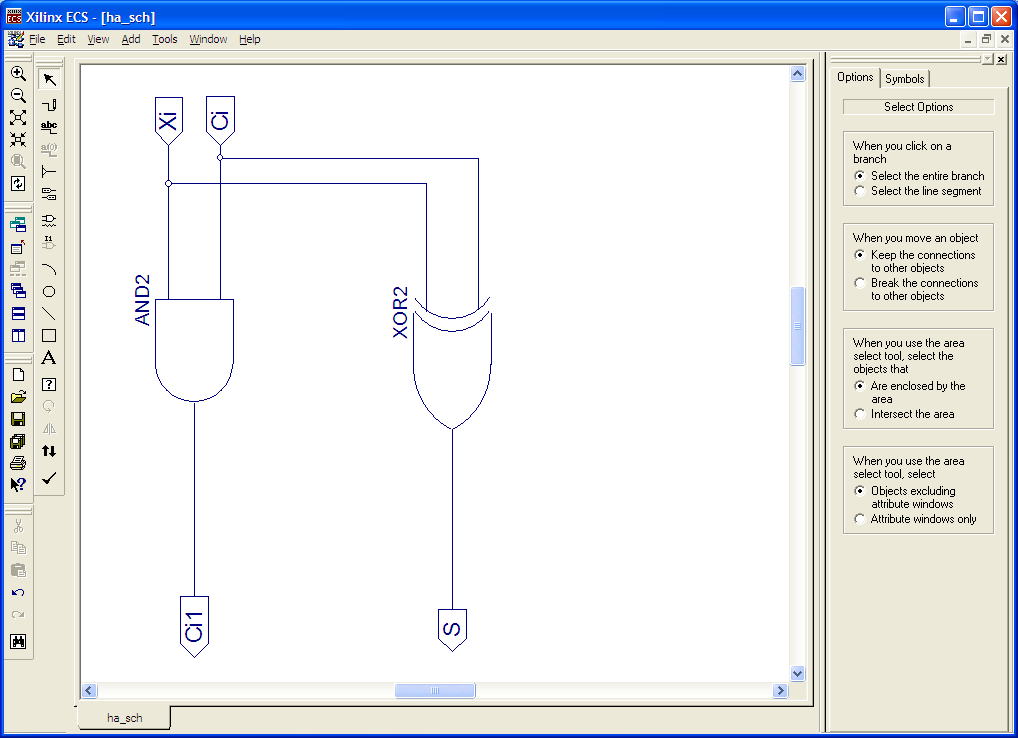
You just finished building the 8-bit register.

**Part 2  
Let’s make a Counter!**

**Why do we do that? Because this counter is going to be your Program Counter in your datapath.**

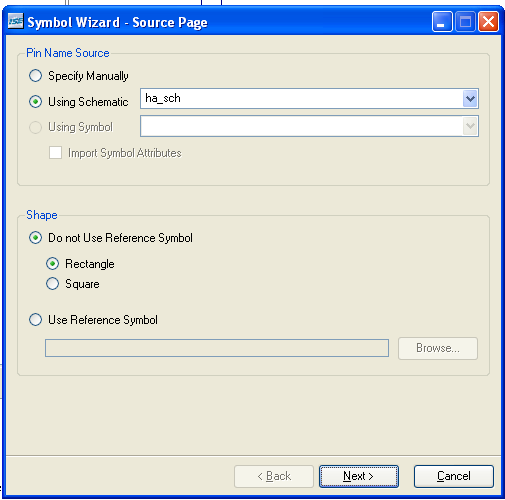
We will first build a half-adder.

1. Click **Project->New Source** in ***Project Manager.*** It is going to be a schematic circuit. Name it **ha\_sch.sch.**
2. In ***ECS***, draw the half-adder circuit as shown below:

  
*Half-adder circuit*

Cout

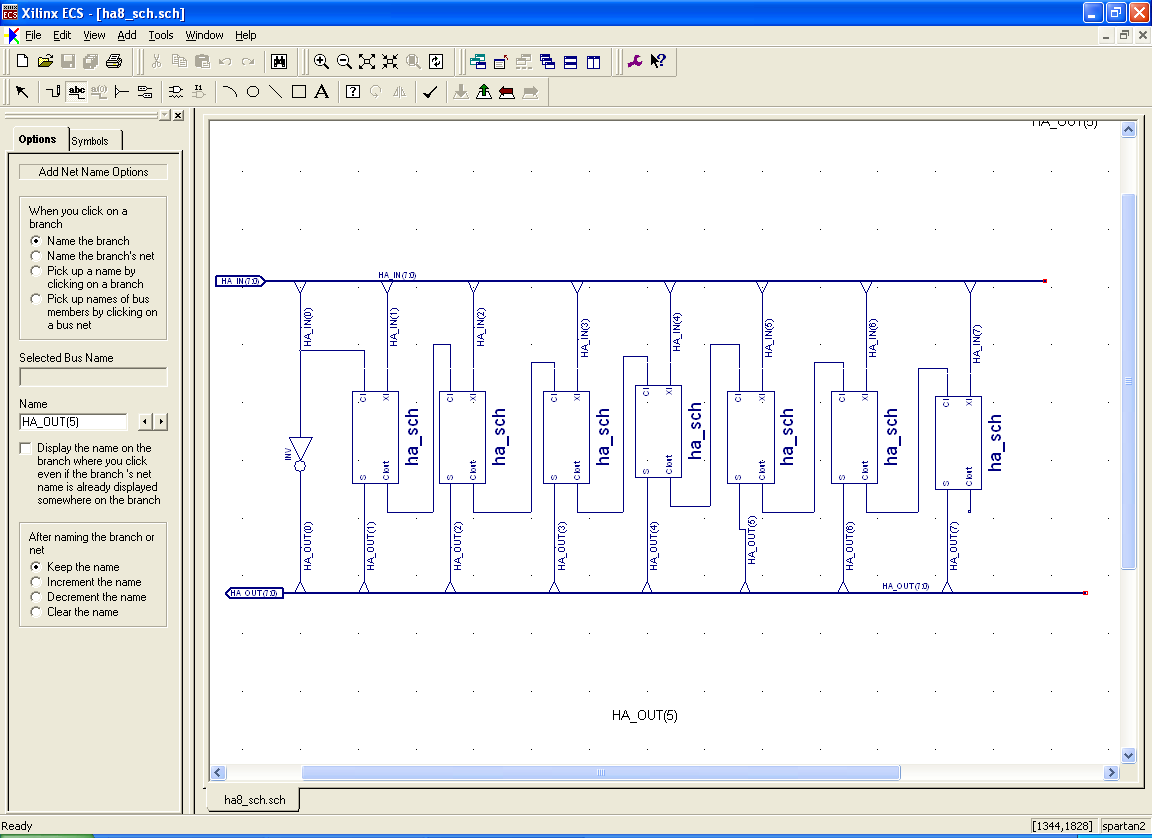
1. Still in ***ECS***, (i.e. make sure the schematic of ha\_sch.sch is open.) go to **Tools->Symbol Wizard.** The following dialog box will pop up:

  
*Symbol Wizard*

1. In the ***Pin Name Source*** groupbox, select **Using Schematic** and select the half-adder schematic, **ha\_sch** from the drop-down menu. In the ***Shape*** groupbox, select **Do not use reference symbol**. You may either choose **Rectangle** or **Square.** That determines the shape of your symbol. Click **Next** until **Finish.**

Next we are going to build a half-adder array. This array will be used in making the counter.

1. Click **Project->New Source** in ***Project Manager.*** It is going to be a schematic circuit. Name it **ha8\_sch.sch.**
2. In ***ECS***, draw the half-adder circuit as shown below Place 7 **ha\_sch** symbols onto a new schematic sheet. Instead of using an 8th **ha\_sch** on the least significant bit, we will use an inverter (NOT) gate since the half-adder effectively inverts its value.
3. Connect each Ci pin on one symbol to the Cout pin on the symbol to its left. This means the Cout pin on the rightmost **ha\_sch** will be left unconnected.
4. Connect each Xi and the input of the INV symbol to an input bus named HA\_IN(7:0).
5. Connect each S and the output of the INV symbol to an output bus named HA\_OUT(7:0).   
   ***Note:*** *Make sure that INV is connected to the LSB of the input and output buses.*

***Note:*** *Cout on the MSB of the half-adder is unconnected.*

**HA\_IN(7:0)**

**HA\_OUT(7:0)**

1. Save **ha8\_sch.sch.** Create a symbol from **ha8\_sch.sch** by invoking the ***Symbol******Wizard*** from the ***Tools*** menu.

Now you will need to build the counter.

1. In ***ISE,*** create a **New Source** of type schematic and name it **counter\_sch.sch.**
2. Build your counter based on your design (**part of pre-lab**). The inputs/outputs of this counter are (including **CLK**):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RST** | **LOAD** | **COUNT** | **COUNT\_IN(7:0)** | **COUNT\_OUT(7:0)** |

The behavior of this counter is as following (**You need to decide the what COUNT\_OUT are**). Please do not forget the **CLK**.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RST** | **LOAD** | **COUNT** | **COUNT\_IN(7:0)** | **COUNT\_OUT(7:0)** |
| 1 | 0 | 0 | COUNT\_IN(7:0) | ? |
| 1 | 0 | 1 | COUNT\_IN(7:0) | ? |
| 1 | 1 | 0 | COUNT\_IN(7:0) | ? |
| 1 | 1 | 1 | COUNT\_IN(7:0) | ? |
| 0 | 0 | 0 | COUNT\_IN(7:0) | Hold Previous Value |
| 0 | 1 | 0 | COUNT\_IN(7:0) | ? |
| 0 | 0 | 1 | COUNT\_IN(7:0) | ? |
| 0 | 1 | 1 | COUNT\_IN(7:0) | ? |

***Note:*** *LOAD takes precedence over COUNT*.

1. Create a test bench called **counter\_tb.v (**shown below, associated to **counter\_sch**)**.** Use the following table to verify the correctness of your counter:

**//counter\_tb.v**

**`timescale 1ns/1ps**

**module counter\_tbw\_tb\_0;**

**reg CLK = 1'b0;**

**reg COUNT = 1'b0;**

**reg [7:0] COUNT\_IN = 8'b00000000;**

**reg LOAD = 1'b0;**

**reg RST = 1'b0;**

**wire [7:0] COUNT\_OUT;**

**parameter PERIOD = 200;**

**parameter real DUTY\_CYCLE = 0.5;**

**parameter OFFSET = 100;**

**initial // Clock process for CLK**

**begin**

**#OFFSET;**

**forever**

**begin**

**CLK = 1'b0;**

**#(PERIOD-(PERIOD\*DUTY\_CYCLE)) CLK = 1'b1;**

**#(PERIOD\*DUTY\_CYCLE);**

**end**

**end**

**counter\_sch UUT (**

**.CLK(CLK),**

**.COUNT(COUNT),**

**.COUNT\_IN(COUNT\_IN),**

**.LOAD(LOAD),**

**.RST(RST),**

**.COUNT\_OUT(COUNT\_OUT));**

**initial begin**

**PLEASE WRITE YOUR OWN TEST VECTORS**

**end**

**endmodule**

Test and make sure it does match the functions of the above table.

**Now we have all the components necessary to build our datapath.**

**Part 3**

**Part 3**

**Putting together the Datapath**

Finally, use all the components that you designed in the directory of ToyProcessor to build the Datapath. More specifically, create a new schematic, which you should call “Datapath”, and instantiate the necessary components. We will not simulate this datapath by itself, so be very careful making the correct connections. Note that the **Cnext** on **alu\_sch** should be left disconnected. The following figure lists all the necessary components for the complete datapath, along with their inputs/outputs. **You can draw the schematics and the connections. Please keep the inputs/outputs naming consistent with the figure (use capital letters). (ADDORSUB = 1 means SUB)**



Lab Report/Code Requirements

Lab Report Format

Each team should submit one report using the format provided below. You should provide a detailed description/discussion for each lab report item including all schematics, codes and simulation waveforms from testbench. The lab report should be in WORD or PDF format and submitted as a single file to the dropbox in ELC. **Please submit your lab reports in appropriate assignment folders on ELC before the deadline. 30% credit will be taken off if the submission is late by 0-24 hours. After 24 hours, late submission will not be accepted and zero credit will be given to the lab report.**

**1. Student Information (First page)**

Lab Number and Title:

Names of members in the team:

E-mails:

**Contributions: Please list contributions (in estimated percentages) of each member in the following categories.**

* **Pre-lab design and analysis**
* **In-lab module and testbench design**
* **In-lab testbench simulation and analysis**
* **In-lab FPGA synthesis and analysis**
* **Lab report writing**

**2. Lab Purpose (Second page)**

Provide a brief summary of the lab assignment.

**3. Implementation Details**

Provide a detailed description of the steps you followed to implement the lab. Be sure to include all relevant information. Recommended information to be included:

* **Pre-lab design and analysis**
* **Schematics of your design if available**
* **Module and testbench code**

**4. Experimental Results**

Describe the results of the lab assignment. Be sure to include any simulation waveforms for testbench. **You need to summarize the simulation results, supplemented by the evidence from Screen Captures of the waveforms. Please DO NOT just copy the waveforms as the results. You need explain why they correctly verify your design. When you attach the waveform screen captures, please crop them so that the variables names, time scale and waveforms are visible in your file.**

**5. Significance**

What is the importance of the lab? How will you use the information you learned in this lab in the future?

In which applications you can use this information?

**6. Comments/Suggestions**

Please provide any comments or suggestions you may have for this lab assignment. Please indicate any areas of difficulty you encountered for this lab assignment.

**7. Additional Lab Specific Information**

Attach any additional information requested for this lab, including schematics, simulation waveforms, or manually performed tasks.

**Grading Policy**

**Only one copy of the lab report from each team is needed. The grade for the report will apply to both team members, provided that each team member contributes roughly equally (50/50 – 40/60) to the lab. In the case where one team member contributes disproportionally lower to the lab, the instructor may reduce the grade of this member. Late submission after due date will receive partial credits (70% of credit for 0-24 hours late submission, no credit for more than 24 hours late submission).**

**Plagiarism: If plagiarism is found in reports, all of them will get ZERO points immediately, and face serious consequences.**