Indian Institute of Technology Bombay

EE214 - Digital Circuits Lab

Mid Semester Exam

Timing: 5:15 PM - 8:30 PM **Date: September 21, 2022**

Exam Instructions:

1. Download the **Question Paper** and **Tracefile** containing **zip folder** from **Moodle** by **5:15 PM**.
2. Turn off your internet and keep your laptop and mobile phone in **airplane mode** after the download.
3. If anyone is caught using internet between the duration of exam **5:15 PM to 8:30 PM strict action** will be taken.
4. **Password for the zip folder will be provided at 5:15 PM**.
5. Read the problem statement carefully and complete the **handwritten design** between **5:15 PM** to **6:30 PM**.
6. **Handwritten design** should comprise of **circuit/block diagram to show the design flow**, **boolean equations** and **justifications for the boolean equations by inspection (as the case may be)** to explain your approach.
7. **Handwritten design** will be **timestamped** by the **respective TA** and the **deadline** is **6:30 PM (sharp)**.
8. Write the **VHDL description** and perform **RTL simulation** by **8:30 PM sharp**. Implementation on Xen10 Board will not be there for Midsem.
9. **No alterations to the handwritten design is allowed after 6:30 PM. If somebody does that they will lose the time in the VHDL coding**
10. **If you wish to change the logic of the design you can do so in your VHDL code but you will only get the marks for the design part based on whatever you have written in pen-paper design before 6:30 PM**.
11. **No time extension will be allowed for VHDL coding and RTL simulation (6:30 PM to 8:30 PM)**
12. Use **Structural Modelling only**.
13. You are allowed to use only the components in **Gates.vhdl** and **your own VHDL descriptions in the experiments/homework problems** so far.
14. You can switch on your internet at 8:30 PM only for uploading the files.
15. The **Final Design (VHDL Design files, qpf file and scanned copy of handwritten design)** in a zip format only (file format: **Rollnumber_Name.zip**) should be uploaded in **Moodle** between **8:30 PM to 8:45 PM (sharp)**.
16. **Hard copy of the handwritten design** will be collected by your **respective TA** at **8:45 PM (sharp)**.
17. Demonstrate your **RTL simulation** to your **respective TA** using the given tracefile for the overall design (if you finish before **8:30 PM**).
18. After **8:45 PM** for all **RTL Simulation demonstrations** you need to **download the uploaded project folders from Moodle** and get it verified by **your TA** one by one.

All the Best!

Problem Statement

In computer computations it is often necessary to compare numbers. Two four-bit signed numbers in 2's complement representation X = $x_3x_2x_1x_0$ and Y = $y_3y_2y_1y_0$ can be compared using subtractor circuit which performs X - Y and the result is S = $s_3s_2s_1s_0$.

Note: The range of **N-bit 2's complement representation** is between -2^{N-1} to $(2^{N-1}-1)$.
Inputs will be in the range of **-8 to 7** for **four bit** input case.
Input format (-8 to 7) : -8$\longrightarrow$ 1000, $-7 \longrightarrow$ 1001, ..., $-1 \longrightarrow$ 1111, $0 \longrightarrow$ 0000, $1 \longrightarrow$ 0001, ..., $7 \longrightarrow$ 0111

Design [20 Marks]

Design a subtractor circuit which performs **X - Y** using Full Adders and logic gates giving the result **S** [2M].

Note: All necessary logic gates and Full Adder units are already present in **Gates.vhdl** provided within the zip file.

Get below three output flags with the following specification after the subtraction:

- **Z=1** if the result is 0; otherwise **Z=0** [4M]
- **N=1** if the result is negative; otherwise **N=0** [4M]
- **V=1** if arithmetic overflow occurs; otherwise **V=0** [4M]

Note: Arithmetic overflow occurs when the magnitude of a number exceeds the range allowed by the size of the bit field. **Example of overflow** : When inputs are X=0100 (i.e. 4) and Y=1010 (i.e. -6) to the subtractor the output is 1010 as shown below in table(1). The output will be depicted by the system as -6 not 10 as it is outside the bound of -8 to 7. Whenever overflow occurs the number wraps around. It can be seen that we are subtracting Y (negative number) from X (positive number) and the result obtained is negative indicating overflow. We need a flag to tell the system that the output has overflowed. V flag detects such overflow cases. Hence, V=1 for input 4 and -6 as given in the example.

0100
- 1010
Taking 2's complement of the subtrahend (2'complement of -6 is 6):
0100
+ 0110
1010

Table 1: Subtraction of 4 and -6

Using the three output flags generated **Z, N** and **V** design logic for the below flags:

- **L=1** if **X** $\leq$ **Y**, otherwise **L=0** [3M]
- **G=1** if **X** $\geq$ **Y**, otherwise **G=0** [3M]

Note: The expression for **L** and **G** need to be in terms of **Z, N** and **V**.

Given below are some of the example input and output combinations:

(i) Inputs : X = 0100 (4) and Y = 1010 (-6)
Expected outputs : Z=0, N=1, V=1, L=0, G=1

(ii) Inputs : X = 0010 (2) and Y = 0101 (5)
Expected outputs : Z=0, N=1, V=0, L=1, G=0

(iii) Inputs : X = 1101 (-3) and Y = 1100 (-4)
Expected outputs : Z=0, N=0, V=0, L=0, G=1

(iv) Inputs : X = 0001 (1) and Y = 0001 (1)
Expected outputs : Z=1, N=0, V=0, L=1, G=1

(v) Inputs : X = 1101 (-3) and Y = 0110 (6)
Expected outputs : Z=0, N=0, V=1, L=1, G=0

VHDL description and Simulation [20 Marks]

- Describe your designed circuit in VHDL using **Structural modelling only**.
- **Using behavioral modelling will fetch 0 marks**
- Files to be edited are given inside the folder **VHDL_Files**
- Main design file is ZNVLG.vhdl where you need to describe the VHDL code
- Full Adder is already provided inside the package Gates.
- Do proper port mapping in DUT.vhdl file
- Edit the number of inputs and outputs in Testbench.vhdl
- Simulate your design using the generic testbench to confirm the correctness of your description.
- To do this, use the tracefile given below and modify the testbench given to you appropriately.

Tracefile format: ($< X3\ X2\ X1\ X0\ Y3\ Y2\ Y1\ Y0\ > \ < Z\ N\ V\ L\ G >$ 11111)

Copy **TRACEFILE_ZNVLG.txt** provided in the EE214_Midsem_2022 folder as **TRACEFILE.txt** and place in your design folder for running the RTL simulation.

Correct VHDL description and RTL Simulation fetches:

- X-Y block [4M]
- Z flag [2M]
- N flag [2M]
- V flag [4M]
- L flag [4M]
- G flag [4M]

Note:

For partial marking run step by step when you finish the design of block/flag.

(i) For partial marking in case you want to test only **X-Y block** please use the below tracefile by modifying main design file, DUT.vhdl and Testbench.vhdl in a separate project folder.

[6M] (Design : 2M & VHDL Description and RTL Simulation : 4M)

Tracefile format: ($< X_3\ X_2\ X_1\ X_0\ Y_3\ Y_2\ Y_1\ Y_0\ > \ < S_3\ S_2\ S_1\ S_0 >$ 1111)

Copy **TRACEFILE_X-Y.txt** provided in the EE214_Midsem_2022 folder as **TRACEFILE.txt** and place in your project folder for running the RTL simulation.

Please keep a separate copy of this folder once you get all test cases success before moving on to the VHDL design for flags as there will be changes in DUT.vhdl and Testbench.vhdl only for this case.

(ii) For partial marking in case you want to test only Z and N flags please use the below tracefile by modifying main design file by giving V=0, L=0 and G=0. [18M] (Design : 10M & VHDL Description and RTL Simulation : 8M)

Tracefile format: ($< X_3\ X_2\ X_1\ X_0\ Y_3\ Y_2\ Y_1\ Y_0\ > \ < Z\ N\ 0\ 0\ 0 >$ 11111)

Copy **TRACEFILE_ZN000.txt** provided in the EE214_Midsem_2022 folder as **TRACEFILE.txt** and place in your project folder for running the RTL simulation.

(iii) For partial marking in case you want to test only Z, N, V flags please use the below tracefile by modifying main design file by giving L=0 and G=0. [26M] (Design : 14M & VHDL Description and RTL Simulation : 12M)

Tracefile format: ($< X_3\ X_2\ X_1\ X_0\ Y_3\ Y_2\ Y_1\ Y_0\ > \ < Z\ N\ V\ 0\ 0 >$ 11111)

Copy **TRACEFILE_ZNV00.txt** provided in the EE214_Midsem_2022 folder as **TRACEFILE.txt** and place in your project folder for running the RTL simulation.

For verification of individual flags additional tracefiles are provided for evaluation purposes inside Additional_TRACEFILE_for_evaluation.zip