NCTU-EE IC LAB – Fall 2024

Lab01 Exercise

Design: Code Calculator

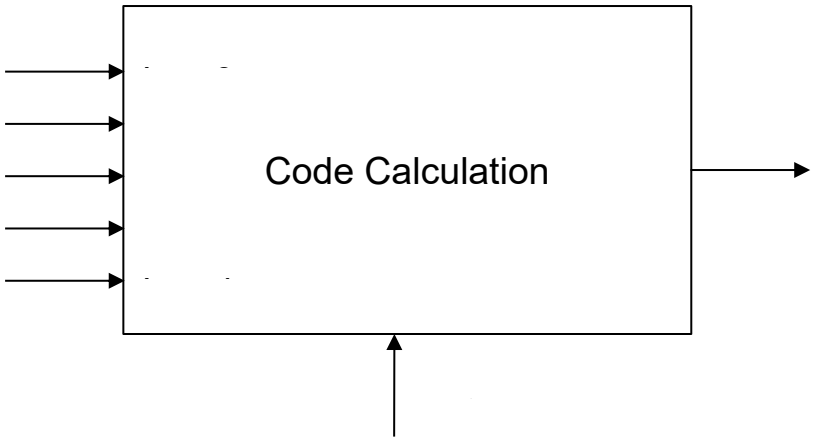
Data Preparation

1. Extract files from TA's directory:
   % tar xvf ~iclabTA01/Lab01.tar
2. The extracted LAB directory contains:
   a. Exercise/
   b. Practice/

Design Description and Examples

At the final stage of NYCU Millionaire, you are asked to answer a question based on a series of simple mathematical operations. The only challenge is the remaining time, 20 ns. If you answer it in time correctly, you will win a prize of a million dollars. "Ready… Start…".

You will receive a sequence with 5 numbers {**in_n0, in_n1, in_n2, in_n3, in_n4**} and a **3-bit opt** signal. Then you should calculate the result in the following order:



First, please do the 3 possible operations indicated by **opt** signal in the following order:

1. Sort	If **opt[1] is 1**, sort the sequence from the **largest to the smallest.** For example, {-1, 3, 5, 4, -3} becomes {5, 4, 3, -1, -3}. If **opt[1] is 0**, sort the sequence from the **smallest to the largest.** For example, { -1, 3, 5, 4, -3} becomes {-3, -1, 3, 4, 5}.
2. Normalization	If **opt[0] is 1,** subtract the average of the largest number

	and the smallest number from every element in sequence. For example, the original sequence is {1, 2, 4, 6, 9}. The value (1+9)/2 should be subtracted from each element in the sequence, and the sequence will become{-4, -3, -1, 1, 4}. If **opt[0] is 0,** don't normalize. For example, the original sequence {1, 2, 4, 6, 9} remains {1, 2, 4, 6, 9}. (round-down the average first if it is not an integer)
3. Calculation	After the two operations, you will get a sequence {**n0, n1, n2, n3, n4**}. Finally, the output answer can be obtained by one of the following equations If **opt[2] is 1, Eq :** | n3 * 3 – n0 * n4 | If **opt[2] is 0, Eq :** ((n0 + n1 * n2 + avg *n3) / 3) **About avg:** Based on the normalization result to calculate the average. For example, the sequence after normalization is {-4, -3, -1, 1, 4}, and you will get the average result, **avg = (-4 + -3 + -1 + 1 + 4)/5 .** (round-down both the answer of Eq and avg if it is not integer)

The summary of the description and specifications are as follows:

Input Signal	Bit Width	Description
in_n0	4	The first number of code, which ranged from **0~15.**
in_n1	4	The second number of code, which ranged from **0~15.**
in_n2	4	The third number of code, which ranged from **0~15.**
in_n3	4	The fourth number of code, which ranged from **0~15.**
in_n4	4	The fifth number of code, which ranged from **0~15.**

opt	3	Operator for different modes. The operation will be encoded as follows: opt[0]: 1: Normalize 0: Don't normalize opt[1]: 1: Sort from largest to smallest. 0: Sort from smallest to largest. opt[2]: 1: Eq : | n3 * 3 – n0 * n4 | 0: Eq : ((n0 + n1 * n2 + avg *n3) / 3)

Output Signal	**Bit Width**	**Description**
out_n	10	The answer. Ranged from -512~511

Inputs

1. The input signals in_n0, in_n1, in_n2, in_n3 and in_n4 are 4-bit inputs
2. The input signal opt is a 3-bit input indicating whether to do the operations and which equation to use for the final result.

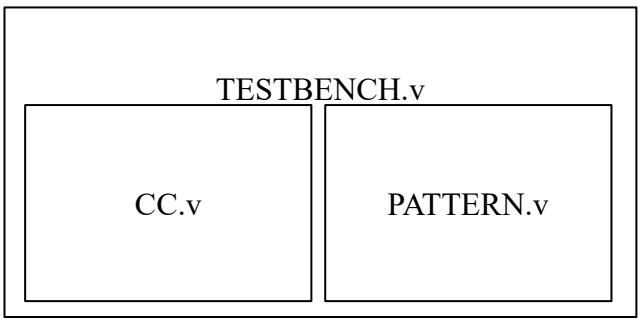
Outputs

The output signal **out_n** is a signed number ranging from **-512~511**. This represents the correct result.

Specifications

1. Top module name : CC (File name: CC.v)
2. Input pins : in_n0, in_n1, in_n2, in_n3, in_n4, opt
3. Output pins : out_n
4. After synthesis, check the "CC.area" and "CC.timing" in the folder "Report". **The area report is valid only when the slack at the end of "CC.timing" is "MET".**
5. The synthesis result **cannot** contain any **latch**.
   Note: You can check if there is a latch by searching the keyword "**Latch**" in 02_SYN/**syn.log**

Block Diagram



Grading Policy

The performance is determined by the area of your design. The less area your design has, the higher grade you get.

Function Validity: 70%

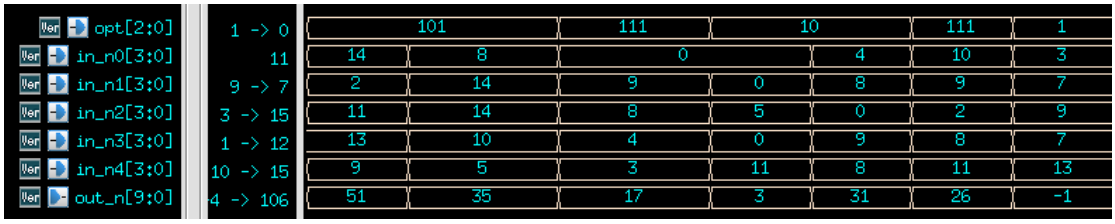
Performance: area 30%

Note

1. Please upload the following file on e3 platform before **12:00 at noon** on **Mar. 4**:
   CC_iclab??.v (?? is your account no.)
   Ex: CC_iclab99.v
   First demo deadline: **12:00 at noon** on **Mar. 4**
   Second demo deadline: **12:00 at noon** on **Mar. 6**
2. Template folders and reference commands:
   In RTL simulation, the name of template folder and reference commands is:
   01_RTL:
   ./01_run
   02_SYN/ (Synthesis):
   ./01_run_dc
   ./08_check
   (Check **latch** by searching the keyword "**Latch**" in 02_SYN/**syn.log**)
   (Check the design's timing in /Report/CC.timing)
   (Check the design's area in /Report/CC.area)
   03_GATE/:
   ./01_run
   09_SUBMIT:
   ./00_tar:
   After ./00_tar, you should add your cycle time.
   For example: ./00_tar 20
   Since in this lab performance is only based on area, changing cycle time by yourself is forbidden.
   ./01_submit
   You can only do this after ./00_tar, or maybe you will upload the wrong file.
   ./02_check
   Always run ./02_check after ./01_submit to make sure you upload the correct file.
   You can key in **./09_clean_up** to clear all log files and dump files in each folder

Example Waveform

Input and output signal:



opt[2:0]	1 -> 0	101		111	10		111	1
in_n0[3:0]	11	14	8	0		4	10	3
in_n1[3:0]	9 -> 7	2	14	9	0	8	9	7
in_n2[3:0]	3 -> 15	11	14	8	5	0	2	9
in_n3[3:0]	1 -> 12	13	10	4	0	9	8	7
in_n4[3:0]	10 -> 15	9	5	3	11	8	11	13
out_n[9:0]	4 -> 106	51	35	17	3	31	26	-1