EECS LAB Summer Training 2024

Lab01 Exercise

Design: Code Calculator

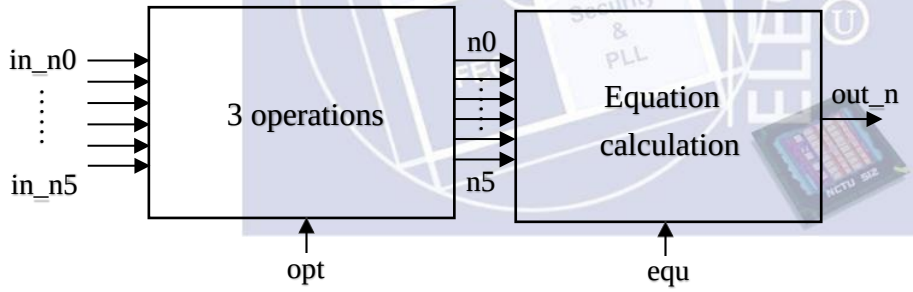
Data Preparation

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

Design Description and Examples

At the final stage of NYCU Millionaire, you are asked to answer a question based on a series of simple mathematic 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 dollar. "Ready… Start…".

You will receive a sequence with 6 numbers {**in_n0, in_n1, in_n2, in_n3, in_n4, in_n5**} a **3-bit opt** signal and a **1-bit equ** 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. Signed/Unsigned	If **opt[0] is 1**, the 6 numbers will be regarded as **2's complement signed** value, which means that there MSB is signed bit. For example, in_n0=4'b1010, then its value is -6. in_n0=4'b0010, then its value is 2 If **opt[0] is 0**, the 6 numbers will be regarded as **unsigned** value. For example, in_n0=4'b1010, then its value is 10 in_n0=4'b0010, then its value is 2

2. Sort	If **opt[1] is 1**, sort the sequence from the **largest to the smallest.** For example, {2, -1, 3, 5, 4, -3} becomes {5, 4, 3, 2, -1, -3}. If **opt[1] is 0**, sort the sequence from the **smallest to the largest.** For example, {2, -1, 3, 5, 4, -3} becomes {-3, -1, 2, 3, 4, 5}.
3. Normalization:	If **opt[2] is 1,** subtract the average of the largest number and the smallest number from every elements in sequence. For example, the original sequence is {1, 2, 3, 4, 6, 9}. The value (1+9)/2 should be subtracted from each element in the sequence, and the sequence will become{-4, -3, 2, -1, 1, 4}. If **opt[2] is 0,** don't normalize. For example, the original sequence is {1, 2, 3, 4, 6, 9} remains {1, 2, 3, 4, 6, 9}. (round-down the average first if it is not integer)

After these three operations, you will get a sequence {**n0, n1, n2, n3, n4, n5**}. Finally, the output answer can be obtained by one of the following equations

1. Eq0 : ((n0 – n1 * n2 + n5) / 3) (round-down the answer if it is not integer) For example, if result is -3.75, round down to -3, if result is 5.5, round down to 5
2. Eq1 : | n3 * 3 – n0 * n4 |
(**Hint: Try to use behavior modeling description instead of gate level description**)

The summary of the description and specifications are as followings:

Input Signal	**Bit Width**	**Description**
in_n0	4	The first number of code. If opt[0] is 0, ranged from **0~15** If opt[0] is 1, ranged from **-8~7**.
in_n1	4	The second number of code. If opt[0] is 0, ranged from **0~15** If opt[0] is 1, ranged from **-8~7**.
in_n2	4	The third number of code. If opt[0] is 0, ranged from **0~15**

		If opt[0] is 1, ranged from **-8~7**.
in_n3	4	The forth number of code. If opt[0] is 0, ranged from **0~15** If opt[0] is 1, ranged from **-8~7**.
in_n4	4	The fifth number of code. If opt[0] is 0, ranged from **0~15** If opt[0] is 1, ranged from **-8~7**.
in_n5	4	The sixth number of code. If opt[0] is 0, ranged from **0~15** If opt[0] is 1, ranged from **-8~7**.
opt	3	Operator for different mode. The operation will be encode as following: opt[0]: 1: Signed. 0: Unsigned opt[1]: 1: Sort from largest to smallest. 0: Sort from smallest to largest. opt[2]: 1: Normalize 0: Don't normalize
equ	1	Decide which equation to be calculated. . If equ is 0 : Eq0 If equ is 1 : Eq1

Output Signal	**Bit Width**	**Description**
out_n	9	The answer. Ranged from -256~255

Examples:

1. **Initial numbers {4'b0101, 4'b1111, 4'b0110, 4'b1101, 4'b1011, 4'b0011} with opt = 3'b110, equ=1'b1:**

   $\{4'b0101, 4'b1111, 4'b0110, 4'b1101, 4'b1011, 4'b0011\} \ -(unsigned)-> \{5, 15, 6, 13, 11, 3\} \ -(sort\ from\ largest\ to\ smallest)-> \{15, 13, 11, 6, 5, 3\} \ -(Normalize)-> \{6, 4, 2, -3, -4, -6\} \ -(Eq1)-> 15$

2. **Initial numbers {4'b0111, 4'b1001, 4'b1010, 4'b0101, 4'b0010, 4'b1110} with opt = 3'b011, equ=1'b1:**

   $\{4'b0111, 4'b1001, 4'b1010, 4'b0101, 4'b0010, 4'b1110\} \ -(signed)-> \{7, -7, -6, 5, 2, -2\} \ -(sort\ from\ largest\ to\ smallest)-> \{7, 5, 2, -2, -6, -7\} \ -(Don't\ Normalize\ )-> \{7, 5, 2, -2, -6, -7\} - (Eq1)-> 36$

3. **Initial numbers {4'b1111, 4'b0011, 4'b1101, 4'b0110, 4'b1111, 4'b1110} with opt = 3'b001, equ=1'b0:**

$\{4'b1111, 4'b0011, 4'b1101, 4'b0110, 4'b1111, 4'b1110\} \ -(signed)-> \{-1, 3, -3, 6, -1, -2\} -(sort\ from\ smallest\ to\ largest)-> \{-3, -2, -1, -1, 3, 6\} -(Don't\ Normalize)-> \{-3, -2, -1, -1, 3, 6\} -(Eq0)-> 0$

Inputs

1. The input signals in_n0, in_n1, in_n2, in_n3, in_n4 and in_n5 are 4-bit inputs
2. The input signal opt is a 3-bit input indicated whether to do the operations and which equation to use to get the final result.
3. The input signal equ is a 1-bit input indicated which equation to be used to calculate.

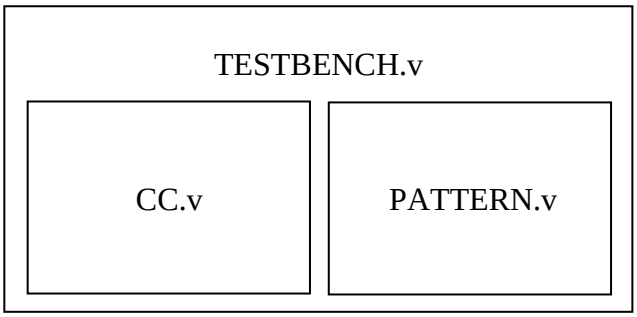
Outputs

The output signal **out_n** is a signed number ranged from **-256~255**. 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, in_n5, opt, equ
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 in 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



Note

1. 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

   (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

   You can key in **./09_clean_up** to clear all log files and dump files in each folder

Example Waveform

Input and output signal: