In the name of God

HOMEWORK #3

CAD

Dr.Khodadadi

Spring 1402

Designer:

Alireza Bayatkhani

Kamyar Moradian

Deadline:

1402/03/12

Notes:

1. Do the homework's with your own information
2. Any similarity between solutions reduces 1 point from overall point of your assignments.
3. 25 percent of your score will be decreased per day after the announced deadline.

سوالات زیر را با زبان VHDL پیاده سازی کنید. در پایان برای هر سوال تمامی فایل های vhdl تون رو اعم از کد و تست بنچ تو یک فایل مرج کنید(همه ی کد ها پشت سر هم باشند) و بفرستید(در صورت عدم تطابق با کد اصلیتون و یا در صورت عدم ارسال نمره اون سوال رو از دست میدید)

و در بخش همه ی پاسخ ها تمامی فولدر های پروژه هاتون رو با داک و عکس ویو فرم ها رو زیپ کنید و ارسال کنید.

Q1)

Subprograms in VHDL, such as procedures and functions, are used to define common operations within many designs and to define common declarations, such as port names and constants values, in one file, instead of declaring in each file.

Design a VHDL package that implements complex matrix operations using procedures and functions. The package should contain custom data type for complex numbers and also supports matrices of different dimensions, and the user should be able to specify the dimensions as generic parameters.

Subprograms included in this package is as follows:

- Function to add two given matrices.
- Function to multiply two given matrices.
- Procedure to fill a matrix randomly and finally return the result as an out signal.
- Procedure to print out the result.

Q2)

Recursion is a method of defining functions in which the function being defined calls itself. The idea is to execute a task in a loop, in a self-similar way. However you should note that recursive functions in VHDL can consume a significant number of resources in FPGA implementations. It is essential to consider the trade-offs between resource usage and design complexity when using recursive functions in VHDL.

Design a package that consists of following recursive functions:

- BCD to binary: Design a function that gets a std_logic_vector which is a BCD number and returns its binary representation. 0001 0010 -> 12 (1100)
- Binary reverser: Design a function that gets a std_logic_vector as input and returns a std_logic_vector which is the reverse form of input. 11000 -> 00011

For binary reverser function, consider a scenario that we want to reverse a binary number with length of 5. Synthesize code for such scenario and draw schematic view of the design.

slide 7 -> page 36

Answer of this problem should contain both the source code and requested schematic view.

Q3)

Consider a 30*30 pixels grayscale image(2D array of integers(0 – 255)) and a 3*3 filter (2D array of real (0-1))

We are going to filter this image using a convolution using a convolution unit that gets a 3*3 window of image and the filter and produces a Real number between 0 to 255 using the formula below:

$$Result = (Window[0,0] * Filter[0,0] + Window[0,1] * Filter[0,1] + Window[0,2] * Filter[0,2] + Window[1,0] * Filter[1,0] + Window[1,1] * Filter[1,1] + Window[1,2] * Filter[1,2] + Window[2,0] * Filter[2,0] + Window[2,1] * Filter[2,1] + Window[2,2] * Filter[2,2])/9$$

But instead of using only one convolution unit which results in taking 28*28=784 cycles to completely filter the image we want to have the option of using up to 28 convolution units(you can use 1,2,4,7,14,28 units only). For example, if we use 28 units in our hardware, we can filter the image in only 28 cycles and get the feature map.

Here is what you are going to do:

Make a package and define the image and filter and the types, functions and procedures you are going to use

Make the convolution unit with 2 inputs and 1 output(you can also pass the whole image to unit with index instead of passing window)

Make the Main components with the generic integer on N that indicates number of Convolution units that we are going to use

Show the result feature map(29*29 of Real)(bonus for saving the result into a txt file)

About using multiple units its up to you how to handle it

You can see how convolution is done with one unit using this link.

https://github.com/KamyarMoradian/CAD-CNN-Hardware/tree/master/CNNHardware/CNNHardware.srcs/sources_1/new

Implement the following question with VHDL language:
Consider a 3030 pixels grayscale image(2D array of integers(0 – 255)) and a 33 filter (2D array of real (0-1)).
We are going to filter this image using a convolution using a convolution unit that gets a 33 window of image and the filter and produces a Real number between 0 to 255 using the formula below:
= ([0,0] [0,0] + [0,1] [0,1] + [0,2]
[0,2] + [1,0] [1,0] + [1,1] [1,1] +
[1,2] [1,2] + [2,0] [2,0] + [2,1]
[2,1] + [2,2] [2,2]) / 9
But instead of using only one convolution unit which results in taking 2828=784 cycles to completely filter the image we want to have the option of using up to 28 convolution units(you can use 1,2,4,7,14,28 units only). For example, if we use 28 units in our hardware, we can filter the image in only 28 cycles and get the feature map.
Here is what you are going to do:
Make a package and define the image and filter and the types, functions and procedures you are going to use.
Make the convolution unit with 2 inputs and 1 output(you can also pass the whole image to unit with index instead of passing window).
Make the Main components with the generic integer on N that indicates number of Convolution units that we are going to use.
Show the result feature map(29*29 of Real) and save the result into a txt file.
About using multiple units its up to you how to handle it.