**2023 Digital IC Design**

**Homework 4:** **Atrous Convolution**

1. **Introduction:**

Atrous convolution, is a technique that expands the kernel by inserting holes between its consecutive elements. In simpler terms, it is the same as convolution but it involves pixel skipping, so as to cover a larger area of the input.

Below are three main advantages using Atrous convolution

1. Increased receptive field without increasing parameters ( e.g. 3x3 atrous convolution kernel with dilation=2 has the same receptive field of 5x5 regular convolution kernel )
2. Can capture feature at multiple scales by altering dilation hyperparameter.
3. Reduced spatial resolution loss compared to regular convolutions with larger filters

Atrous convolutions have been used successfully in various applications, such as semantic segmentation, where a larger context is needed to classify each pixel, and audio processing, where the network needs to learn patterns with longer time dependencies.

The goal of this assignment is to design a atrous convolutional circuit, which consists of two layers, Layer 0 and Layer 1 as Figure 2 shows.

Layer 0 takes a grayscale image (64x64) as input and applies “replicate” padding to the input, resulting in a 68x68 grayscale image. Afterward, Layer 0 performs atrous convolution operation ( stride=1, dilation=2 ) on the 68x68 grayscale image with a 3x3 kernel, followed by a ReLU function.

Layer 1 applies max-pooling to downsample the output image of Layer 0 and round up the result of max-pooling to the nearest integer.

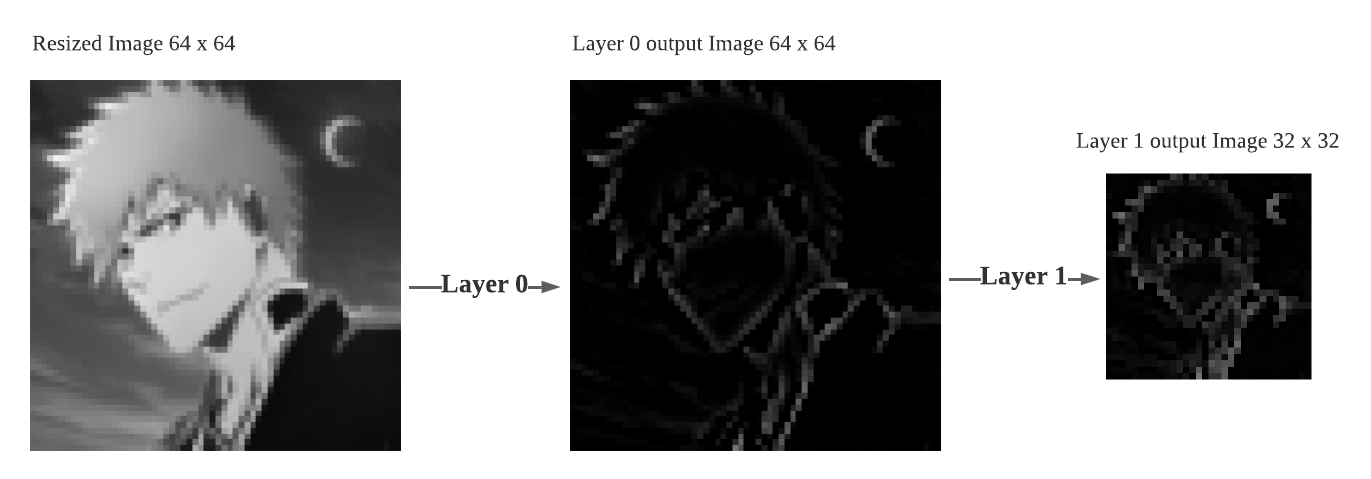
More detailed circuit functionalities will be described in subsequent sections.

Figure 1. Atrous Convolution Example

1. **Design Specifications:**

**2.1 Block Overview:**

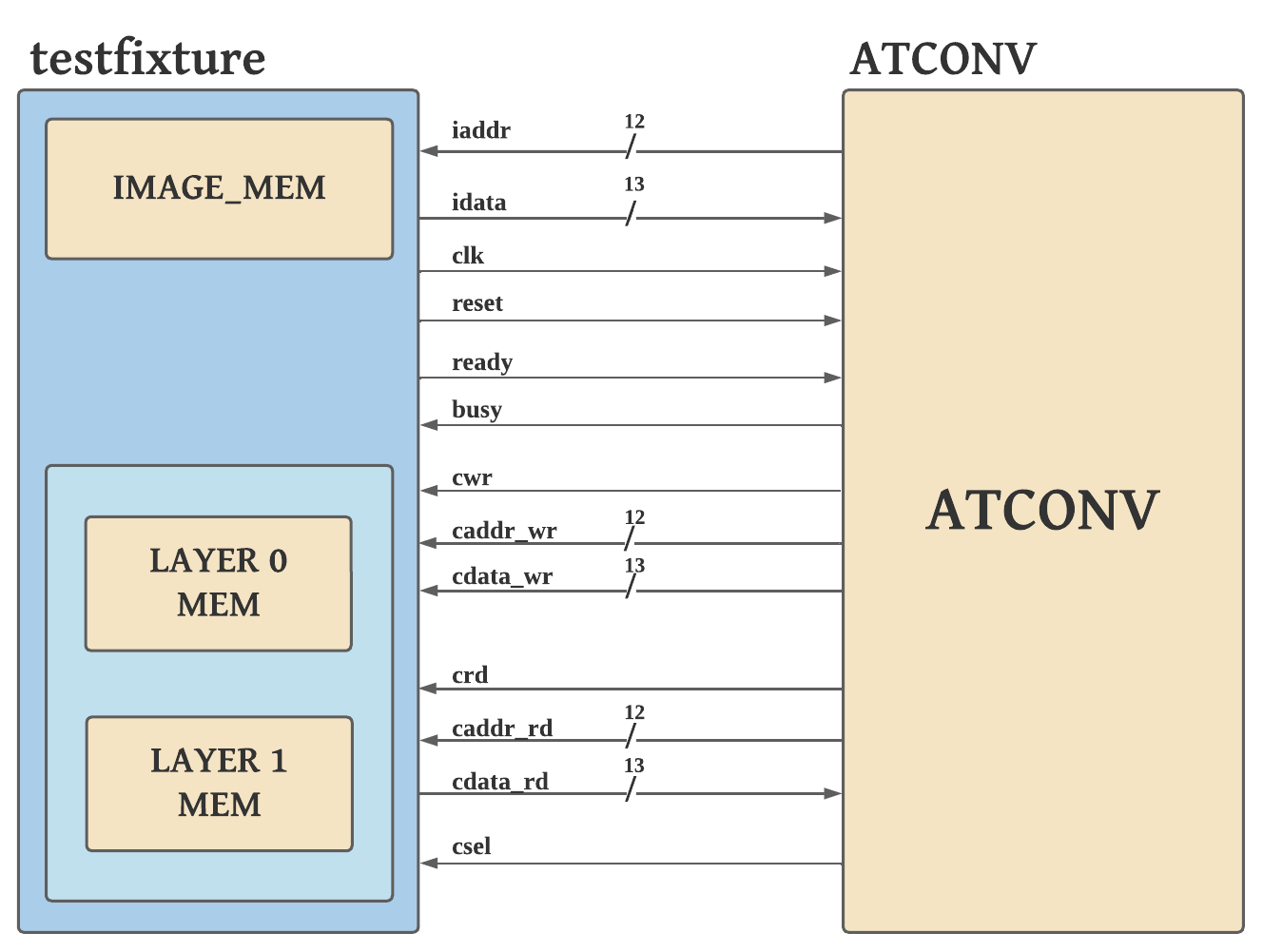
****

Figure 2. System Block Diagram

**2.2 I/O Interface:**

|  |  |  |  |
| --- | --- | --- | --- |
| Signal Name | I/O | width | Description |
| *clk* | I | 1 | System clock signal. This system is synchronized with the positive edge of the clock. |
| *reset* | I | 1 | Active-high asynchronous reset signal. |
| *ready* | I | 1 | Image memory ready signal. When ready signal is high, the 64x64 grayscale image is loaded to the memory. The "iaddr" signal can effectively obtain "idata" only when the ready signal is high. |
| *busy* | O | 1 | After ready signal is set to high, ATCONV should pull the busy signal high before requiring data with “iaddr”. After ATCONV finishing all the tasks, pull the busy signal low and testfixture would check the result. |
| *iaddr* | O | 12 | Input grayscale image’s pixel address signal. Specifying the request grayscale image pixel address. |
| *idata* | I | 13 | Input grayscale image’s pixel data signal. The pixel data consists of 9 bits integer part (MSB) and 4 bits fractional part (LSB). Testfixture will use “idata” to transmit the pixel data from the address indicated by “iaddr” to ATCONV. |
| *csel* | I | 1 | Memory selection signal.  1’b0: read/write Layer 0 memory  1’b1: read/write Layer 1 memory |
| *crd* | O | 1 | Read enable signal. If the signal is set to high, it indicates a read operation is required. Testfixture will read the data from the address “caddr\_rd” of the selected memory and store it in “cdata\_rd”. |
| *caddr\_rd* | O | 12 | Read address signal. Specifying the request data address of the selected memory. |
| *cdata\_rd* | I | 13 | Read data signal. The data read from the selected memory would be transmitted to ATCONV with “cdata\_rd”. Each data consists of 9 bits integer part (MSB) and 4 bits fractional part (LSB). |
| *cwr* | O | 1 | Write enable signal. If the signal is set to high, it indicates a write operation is required. Testfixture will write “cdata\_wr” to the address “caddr\_wr” of the selected memory. |
| *caddr\_wr* | O | 12 | Write address signal. Specifying the address of the selected memory to be written with data “cdata\_wr”. |
| *cdata\_wr* | O | 13 | Write data signal. The data to be written to selected memory address “caddr\_wr”. “cdata\_wr” consists of 9 bits integer part (MSB) and 4 bits fractional part (LSB). |

Table I. I/O interface of the design

**2.3 Function Description:**

**2.3.1 Atrous convolutional circuit behavior overview**

Input image is a 64 x 64 grayscale image stored in testfixture’s memory.

In Layer 0, there are three main tasks to accomplish.

1. Replicate padding the 64x64 grayscale input image in order to maintain the same image size after atrous convolution
2. Apply Atrous convolution to the padded grayscale image
3. Implement ReLU function on the result of astrous convolution

In Layer 1, there are two main tasks to accomplish.

1. Max-pooling the output of Layer 0 ( stride=2, kernel\_size=2x2 )
2. Round up the result of Max-pooling to the nearest integer

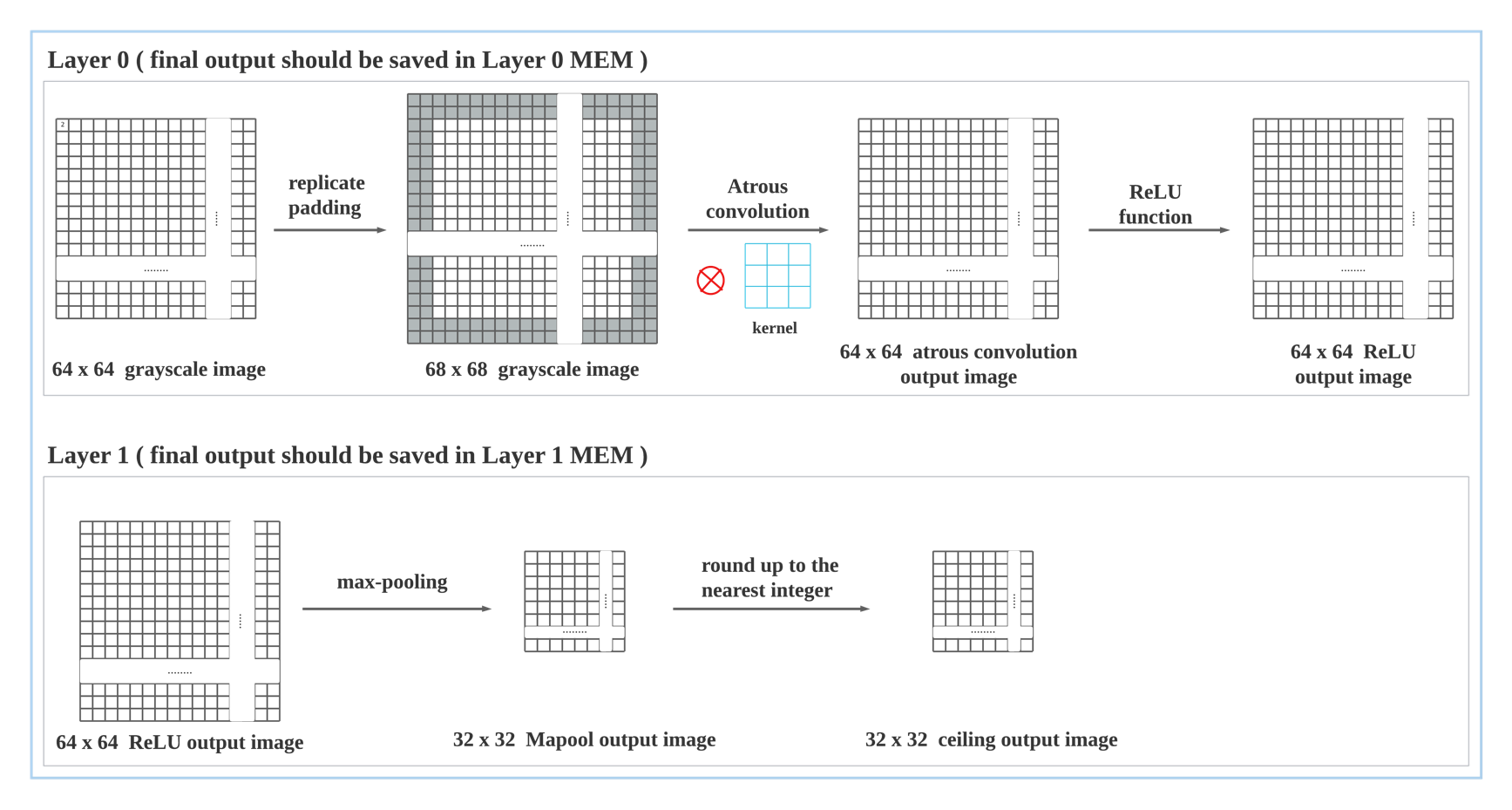
The result of each Layer should be saved in Layer 0 Mem and Layer 1 Mem respectively. Testfixture would check the functionality of ATCONV after busy signal is set to low.

Figure 3. Atrous convolutional circuit behavior overiew

**2.3.2 Memory Mapping Relations**

1. Input Grayscale Image and IMAGE\_MEM

**一張含有 資料表 的圖片

自動產生的描述**

Figure 4. Input Grayscale Image memory mapping

1. Layer 0 Result and Layer 0 Mem

**一張含有 資料表 的圖片

自動產生的描述**

Figure 5. Layer 0 result memory mapping

1. Layer 1 Result and Layer 1 Mem

**一張含有 資料表 的圖片

自動產生的描述**

Figure 6. Layer 1 result memory mapping

**2.3.3 Replicate padding Example:**

For maintaining the same size (64x64) after atrous convolution, replicate padding the input grayscale image to 68x68 is required.

Below is an example of how to replicate padding works on grayscale image.

When padding parameter equals 1, each boundary would replicate the boundary value to create 1 padding pixel at each side of the boundary. Similarly, when padding parameter equals 2, each boundary would replicate the boundary value to create 2 padding pixels at each side of the boundary.

In this assignment, the padding parameter equals 2.

**一張含有 文字, 日曆 的圖片

自動產生的描述**

Figure 7. replicate padding example

**2.3.4 Atrous Convolution Example:**

To further extrapolate how Atrous Convolution works, the definition of the term “dilation” is given below.

**( Dilation:** controls the spacing between the kernel points. )

In this assignment, the dilation parameter equals 2, while in traditional convolution, the dilation parameter equals 1. The following example will use a 3x3 kernel as an illustration, with a dilation rate of 2. The atrous convolution would be implemented on the padded 3x3 image (7x7) and the stride is set to 1.

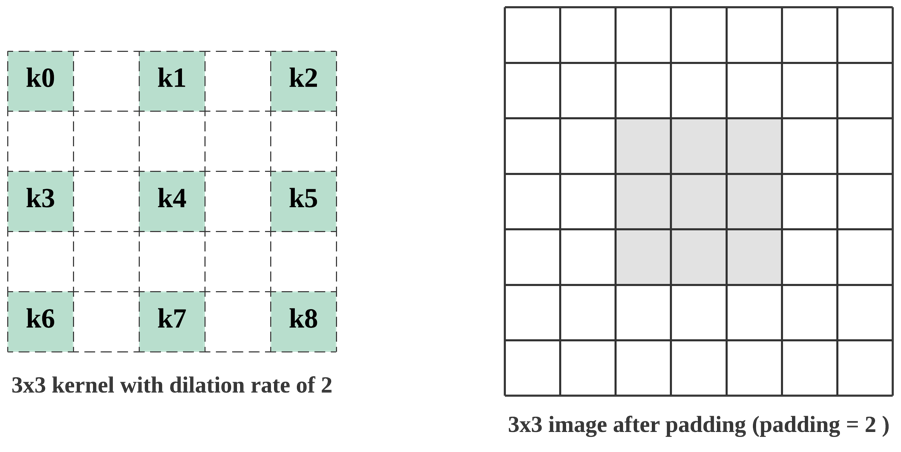


Figure 8. kernel and image used in example

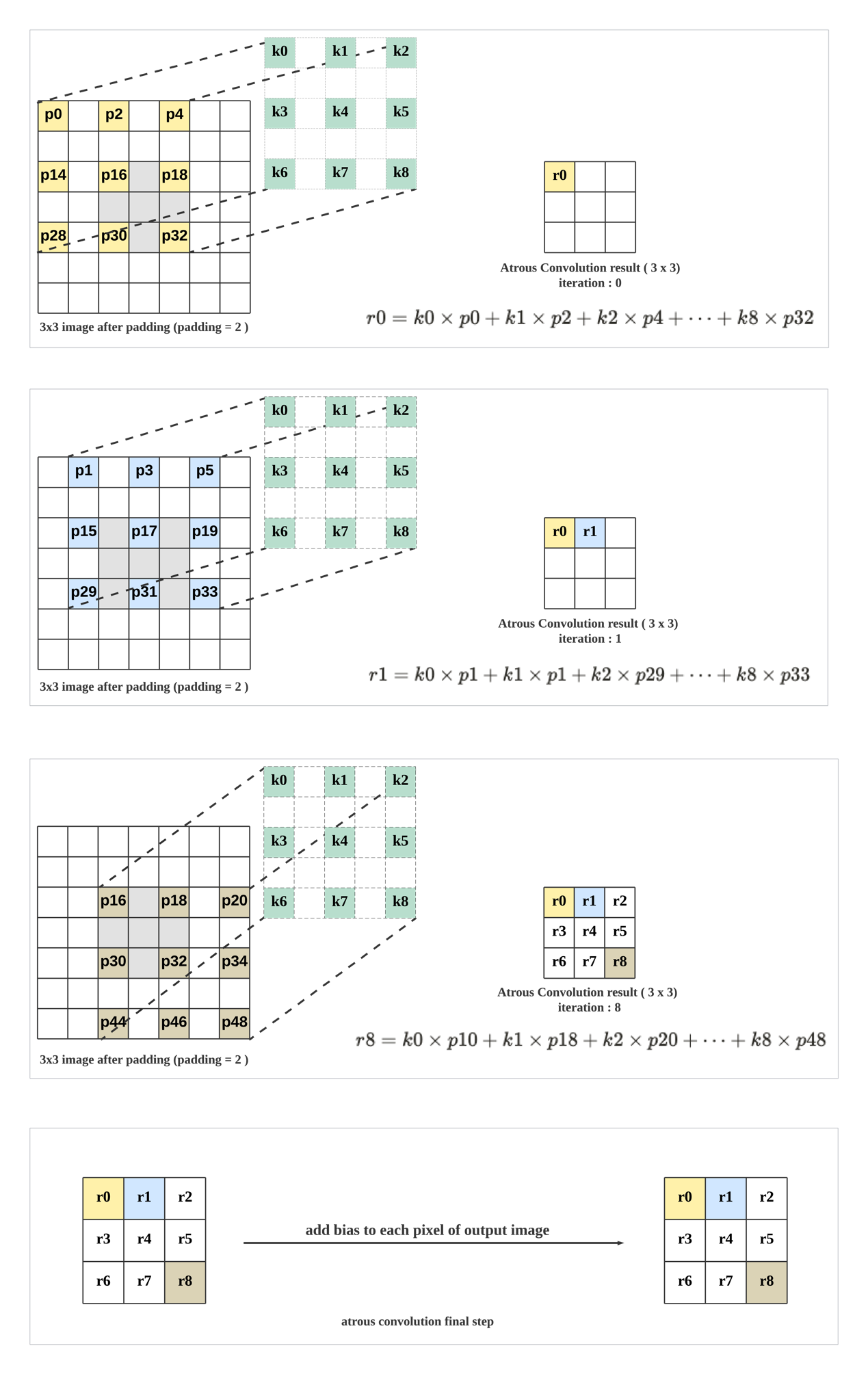


Figure 9. Atrous convolution example on 3x3 padded image (iteration 0, 1, 8)

From the example above, we can notice that the output of atrous convolution and the original image have the same size since we’ve padded the image to 7x7.

Below are the kernel and bias value used in this assignment. Meanwhile, the stride of this assignment equals 1.

**一張含有 日曆 的圖片

自動產生的描述**

Figure 10. Kernel and bias used in this assignment

**2.3.5 ReLU Function:**

In the last step of Layer 0, ReLU function is applied to the output of atrous convolution. The definition of ReLU function is shown below.

**2.3.6 max-pooling Example:**

Max-pooling is adopted in the first step of Layer 1. How max-pooling works would be explained below. In this assignment, the kernel size of max-pooling equals 2x2 and the stride of max-pooling equals 2. The example below adopts the same hyperparameters used in this assignment.

一張含有 文字, 電子產品 的圖片

自動產生的描述

Figure 11. Max-pooling example

**2.4. Timing Specifications:**

**2.4.1 ATCONV system busy control**

When the Input grayscale Image is ready, ATCONV should pull the busy signal high before further action. After ATCONV finish all the tasks, pull the busy signal low to inform testfixture for scoring the result.

**一張含有 圖表 的圖片

自動產生的描述**

Figure 12. system busy control

**2.4.2 Input grayscale image pixel data retrieval control**

When the Input grayscale image ready, alternating iaddr could retrieve the corresponding idata on the negedge of clk.

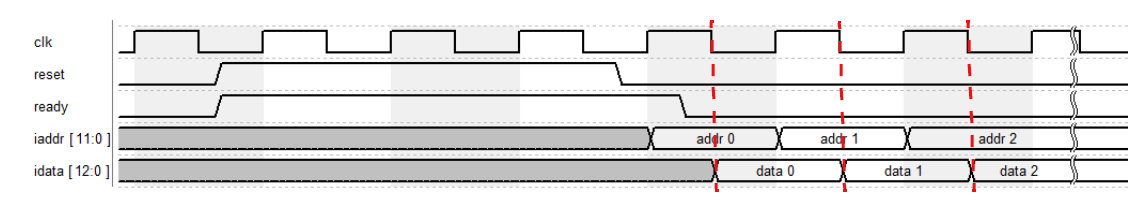
****

Figure 13. input image pixel retrieval control

**2.4.3 Layer 0/1 Memory data reading control**

When the crd signal is set to 1, testfixure would transmit the corresponding cdata\_rd to ATCONV on the negedge of clk. (be aware of caddr\_rd and csel while retrieving data. )

**一張含有 圖表 的圖片

自動產生的描述**

Figure 14. memory data reading control

**2.4.4 Layer 0/1 Memory data writing control**

When the cwr signal is set to 1, ATCONV would write cadata\_wr to the selected memory’s address ( caddr\_wr ).

**一張含有 圖表 的圖片

自動產生的描述**

Figure 15. memory data writing control

**3. Software Verification:**

You need to write a program to verify your circuit. You can write your program by using C, C++ and python.( python is recommended ) The requirements and functionality of this program should be as following:

1. Be able to read any jpg or png image. The file path for reading should be ./image.jpg or ./image.png.

2. Convert RGB image to gray scale.

3. Resize the image into 64x64 and output the resized image as img.dat.

4..implement the algorithm in layer 0 and output the result of layer 0 as layer0\_golden.dat.

5. implement the algorithm in layer 1 and output the result of layer 1 as layer1\_golden.dat.

6. The format of three output files should be the same asthe img.dat, layer0\_golden.dat and layer1\_golden.dat which TA provided. The output files should be ableto be read by the testbench and simulate correctly.

( hint: you should transform IEEE754 floating point into fixed point number and save it using binary representation. )

7. Please name your program in the format: main.c, main.cpp or main.py.

8. If your program needs to be executed with special libraries or in special

environment, please provide a Readme.txt to explain how to execute it.

**( notice: pytorch is allowed to be used, the behavior should be same as your Verilog code. )**

**4. File Description:**

|  |  |
| --- | --- |
| File Name | Description |
| ATCONV.v | The top module of your design. |
| testfixture.v | The testbench file. |
| img.dat | Input grayscale image data. |
| layer0\_golden.dat | answer of layer 0 output data. |
| layer1\_golden.dat | answer of layer 1 output data. |

**5. Scoring:**

**4.1 Function simulation [ %]**

All of the result should be generated correctly, and you will get the following message in ModelSim simulation. If unable to complete all tasks, you will receive partial credit of 0. \* your score.

一張含有 資料表 的圖片

自動產生的描述

Figure 16. functional simulation result example

**4.2 Gate-Level Simulation [ %]**

4.2.1 Synthesis:

Your code should be synthesizable. After it is synthesized in Quartus, a file named ATCONV.vo will be obtained.

4.2.2 Simulation:

All of the result should be generated correctly using ATCONV.vo, and you will get the following message in ModelSim simulation. If unable to complete all tasks, you will receive partial credit of 0. \* your score.

一張含有 資料表 的圖片

自動產生的描述

Figure 17. gate level simulation result example

**4.3 Performance [ %]**

The performance is scored by the total logic elements, total memory bit, and embedded multiplier 9-bit element your design used in gate-level simulation and the simulation time your design takes. The score will be decided by your ranking in all received homework. (The smaller, the better)

Scoring = (Total logic elements + total memory bit + 9\*embedded multiplier 9-bit element) × (Total cycle)

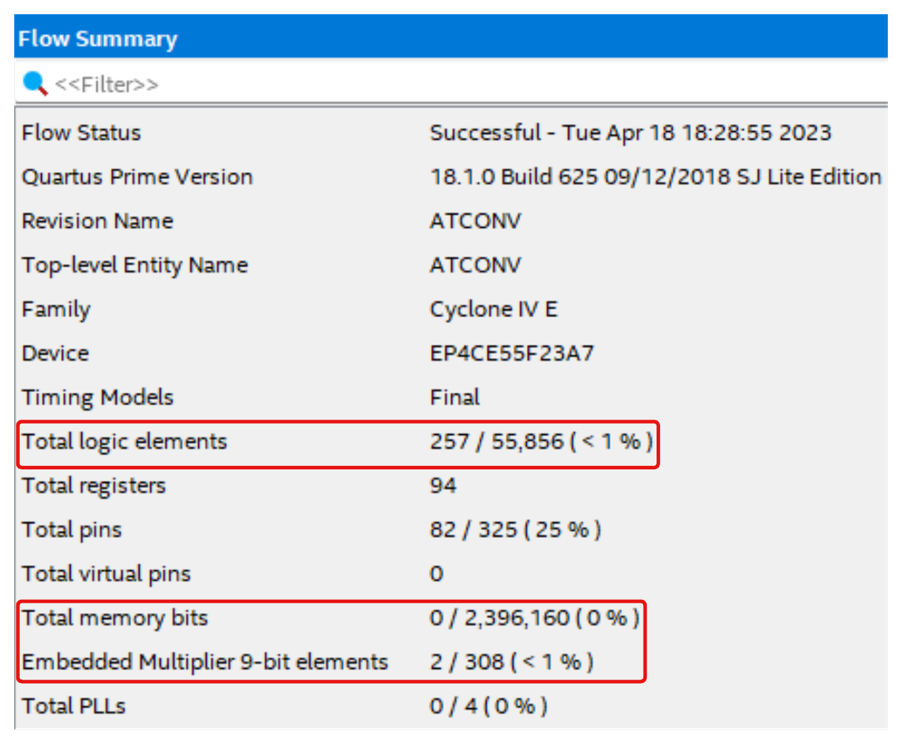


Figure 18. Quartus synthesis result example

**5. Submission:**

**5.1 Submitted files**

You should classify your files into two directories and compress them to .zip format. The naming rule is HW4\_studentID\_name.zip. If your file is not named according to the naming rule, you will lose five points.

|  |  |
| --- | --- |
|  | **RTL category** |
| \*.v | All of your Verilog RTL code |
|  | **Gate-Level category** |
| \*.vo | Gate-Level netlist generated by Quartus |
| \*.sdo | Gate-Level netlist generated by Quartus |
|  | **Documentary category** |
| \*.pdf | The report file of your design (in pdf). |

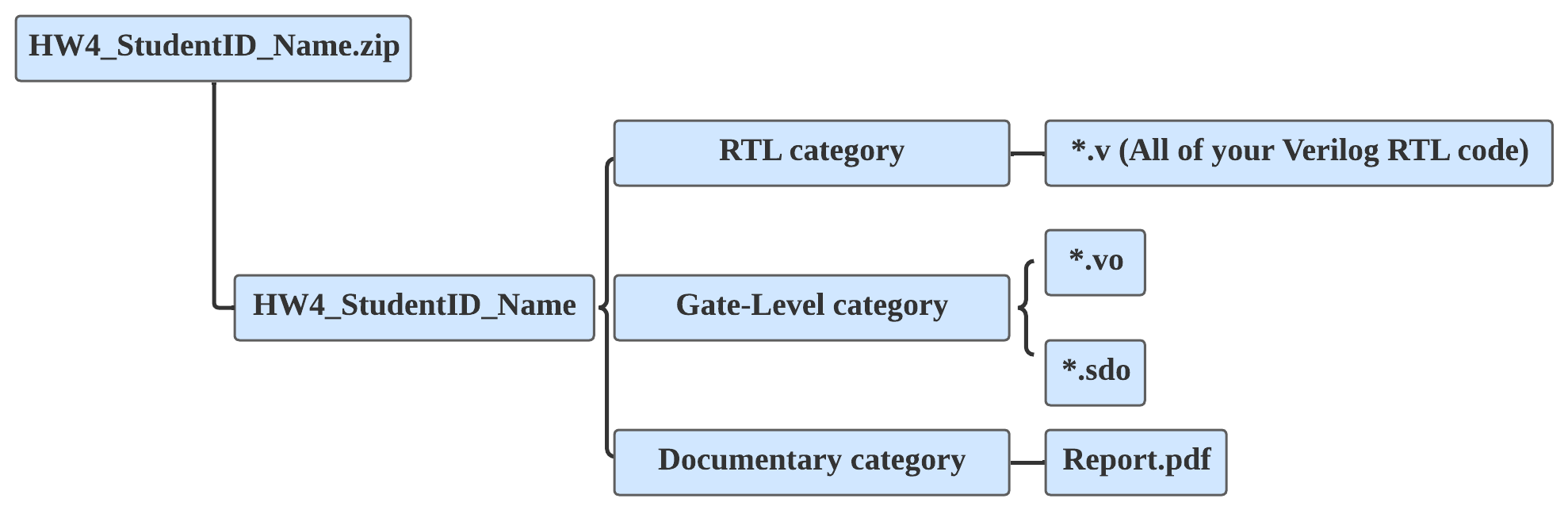
****

Figure 19. assignment directory

**5.2 Report file**

Please follow the spec of report. You are asked to describe how the circuit is designed as detailed as possible.

**5.3 Note**

Please submit your .zip file to folder HW4 in moodle.

Deadline:

If you have any problem, please contact TA by email

P76114537@gs.ncku.edu.tw