**National Cheng Kung University**

**Department of Electrical Engineering**

Introduction to VLSI CAD (Spring 2021)

Lab Session 4

**Synthesis, Register Files and Convolution**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | | Student ID | |
| 陳旭祺 | | E24099059 | |
| **Practical Sections** | **Points** | | **Marks** |
| Prob A | 25 | |  |
| Prob B | 25 | |  |
| Prob C | 30 | |  |
| Report | 20 | |  |
| Notes: | | | |

**Due Date: 15:00, March 31, 2021 @ moodle**

**Deliverables**

1. All Verilog codes including testbenches for each problem should be uploaded.

NOTE: Please **DO NOT** paste source code in the report!

1. Noted! TA will use commands in Appendix A to check your design in SoC Lab. If TA can not compile your code with the commands, you will not get full credit.
2. **If you upload a dead body which we can’t even compile, you will get NO credit!**
3. **All Verilog file should get at least 90% SuperLint Coverage.**
4. All homework requirements should be uploaded in this file hierarchy or you will not get full credit.

NOTE: Please **DO NOT** upload waveforms!

1. The name of uploaded file should be Lab4\_StudentID.tar, and the student ID must be uppercase. (Ex. Lab4\_E24081234.tar)

**Lab4\_studentID.tar**

**Lab4\_studentID**

**studentID.doc**

**ProbA**

**ProbB**

**regfile.v**

**regfile\_tb.v**

**mini\_vending.v**

**mini\_vending\_tb.v**

**mini\_vending\_syn.v**

**mini\_vending\_syn.sdf**

**ProbC**

**CONV.v**

**PRelu.v**

**top.v**

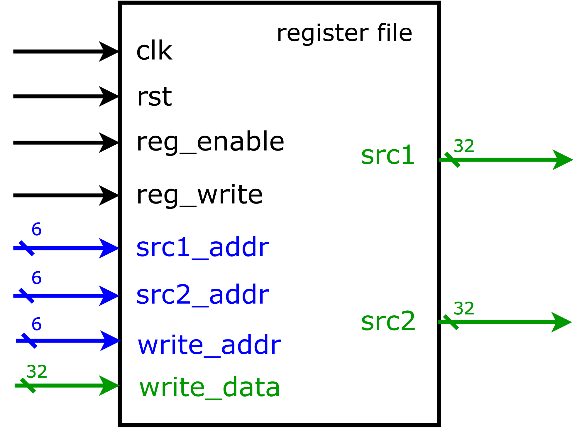
**top\_tb.v**

**top\_syn.v**

**top\_syn.sdf**

Fig.1 File hierarchy for Homework submission

Prob A: Register File



1. Based on the register file structure in LabA, please design a 64 x 32 register file by yourself. *We do not provide any reference code in the file package, but you can refer the code in Lab4 tutorial PDF.*
2. Port list



1. You should follow the file name rules as follow.

Register file

* + File name: **regfile.v**
  + Module name: **regfile**

Register file testbench

* + File name: **regfile\_tb.v**
  + Module name: **regfile\_tb**

1. You should verify your code by the following test patterns.

Write data into register file

|  |  |  |  |
| --- | --- | --- | --- |
| $time | reg\_write | write\_addr | write\_data |
| 16 | 1 | 6’d5 | 32’hffff\_0000 |
| 26 | 1 | 6’d15 | 32’hffff\_0001 |
| 36 | 1 | 6’d24 | 32’hffff\_0002 |
| 46 | 0 | 6’d30 | 32’hffff\_0003 |
| 56 | 0 | 6’d33 | 32’hffff\_0004 |
| 66 | 1 | 6’d44 | 32’hffff\_0005 |
| 76 | 1 | 6’d63 | 32’hffff\_0006 |
| 86 | 1 | 6’d63 | 32’hffff\_ffff |

Read data from register file

|  |  |  |  |
| --- | --- | --- | --- |
| $time | reg\_write | src1\_addr | src2\_addr |
| 96 | 0 | 6’d1 | 6’d2 |
| 106 | 0 | 6’d5 | 6’d33 |
| 116 | 0 | 6’d15 | 6’d44 |
| 126 | 0 | 6’d24 | 6’d63 |

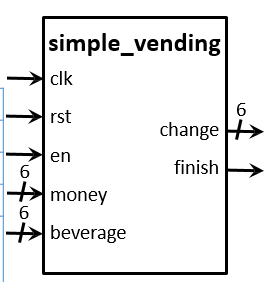
1. Show waveforms to explain that your register work correctly when read and write.

|  |
| --- |
| Simulation result on the terminal |
| 一張含有 桌 的圖片  自動產生的描述一張含有 桌 的圖片  自動產生的描述 |
| Waveform |
|  |
| Explanation of your waveform |
| 上面的波形僅有input和output的訊號，即為testbench給的值，因此還需用指令去抓reg內部讀取狀況至terminal上去觀察。  第一部分測試寫入能力在reg 0, 15, 24, 44, 63皆寫入data到reg；但當reg\_write拉低訊號至0時，就無法把data寫進reg(而是讀取src1, sr2值)，所以30, 33還是維持初始化的值，也就是0。  第二部分測試讀出能力，此時因此都可讀出數值。  另外由於always觸發條件為，也就是說只有在這兩種情況下，值才會改變。 |

1. Show SuperLint coverage

|  |
| --- |
|  |
|  |

Prob B: Revise a circuit “vending machine”



1. Revise a simple vending machine. The following is simple vending module’s specification. (Do NOT add or delete any I/O ports, internal wires and registers, but you can change their behavior.)
2. Port list



1. Please describe what the faults in module and what the ways to revise them.

Faults

|  |
| --- |
| 直接跑Superlint結果如下，查reference一個個去debug |
|  |

Solutions

|  |
| --- |
| 1.Sequential電路寫在同一個always block  2.用if else或case語句時所有條件都要包括到，所以最後還要加else結構  3.Blocking語句(=)一開始給所有用到的值初始化，就不須擔心因為有些情況沒有assign到所有用到值，而電路會為了鎖住上次的值而產生latch，這是我們不樂見的 |

1. **After synthesizing your design, you may have some information about the circuit. Please fill in the following form**.

|  |
| --- |
|  |

Timing (slack)

|  |
| --- |
|  |

Area (total cell area)

|  |
| --- |
|  |

Power (total)

|  |
| --- |
|  |

1. Your simulation result on the terminal. (You should $monitor the I/O port like the example TAs’ code does! )

|  |
| --- |
|  |

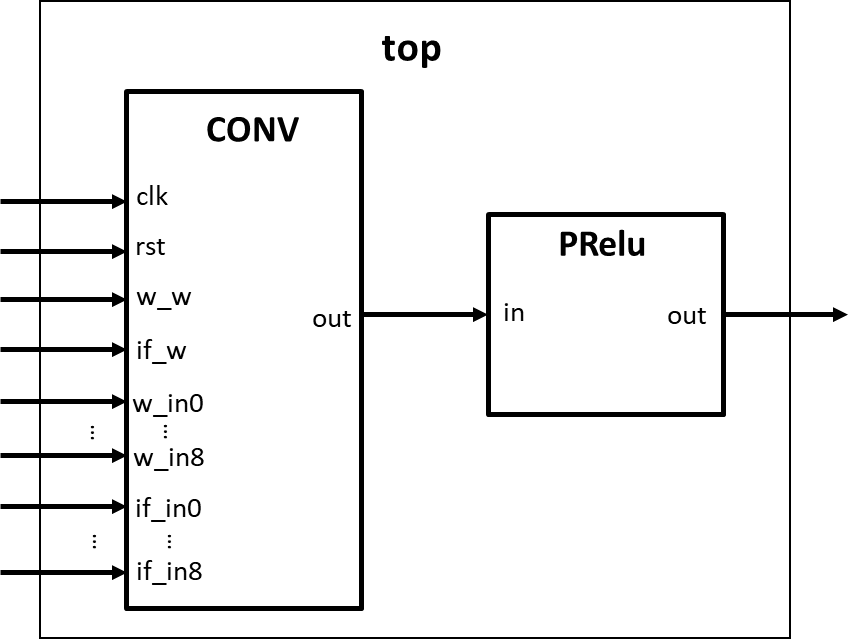
1. Your waveform and explanation of your waveform

|  |
| --- |
| Waveform |
|  |
| Explanation of your waveform |
| vending machine分為三個階段   |  |  |  | | --- | --- | --- | | Phase0 | Phase1 | Phase3 | | wait for inserting money | choose beverage | get change & finish |   以上波形可看出   |  |  | | --- | --- | | Phase0 | 使用者投錢，機器並把錢先存在money\_temp | | Phase1 | 選擇飲料並把money\_temp減去beverage的商品價格 | | Phase3 | 找錢，並把finish拉高，讓使用者知道交易已完成/\*此部分用conbinatioal寫，要與sequential電路分開寫\*/ | |

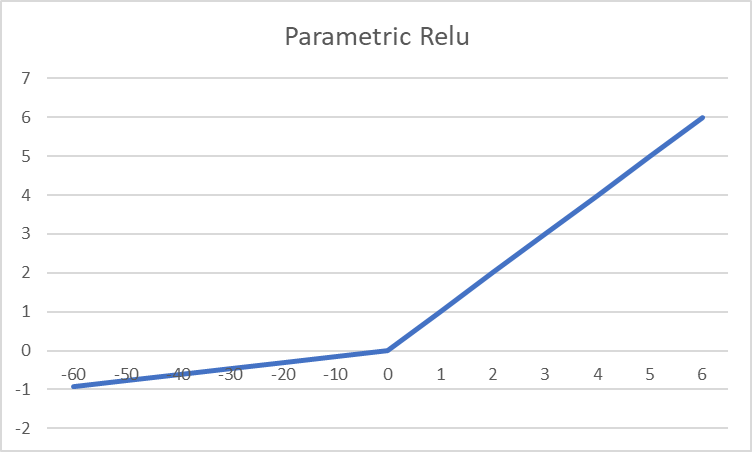
1. Superlint Coverage

|  |
| --- |
|  |
|  |

Prob C: Convolution and activation function

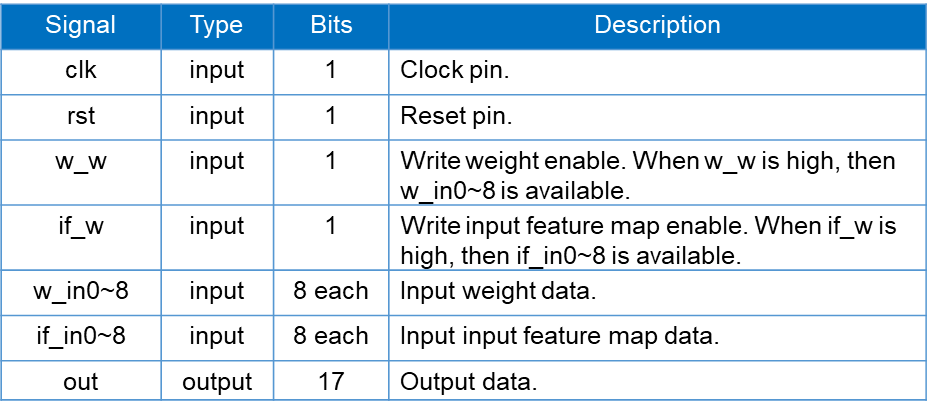


1. Based on Lab B , please design a Convolution unit and a PRelu unit by yourself.

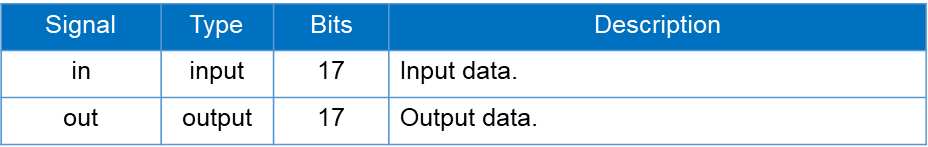


1. Port list

CONV



Parametric Relu



1. You should follow the file name rules as follow.

Convolution

* + File name: **CONV.v**
  + Module name: **CONV**

Relu

* + File name:**PRelu.v**
  + Module name: **PRelu**

Top

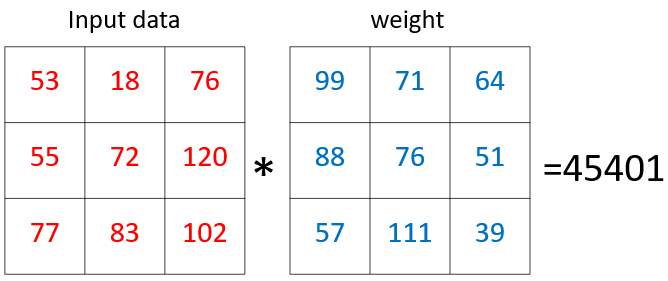
* + File name: **top.v**
  + Module name: **top**

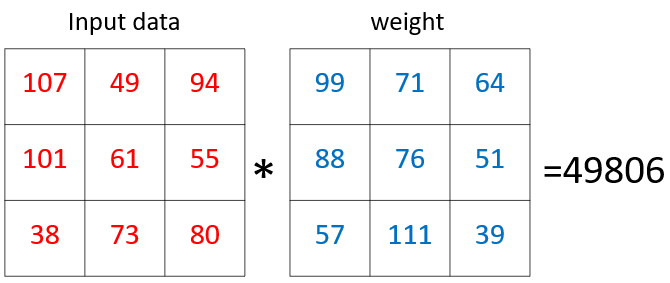
Testbench

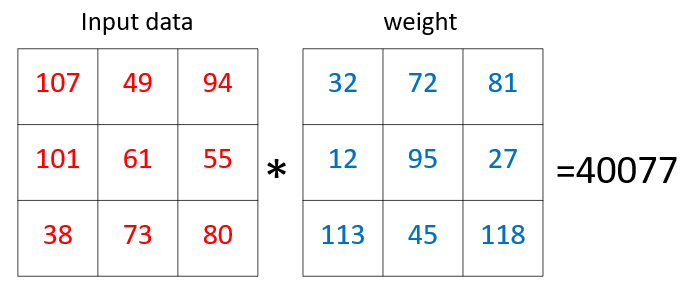
* + File name: **top\_tb.v**
  + Module name: **top\_tb**

1. Data and result

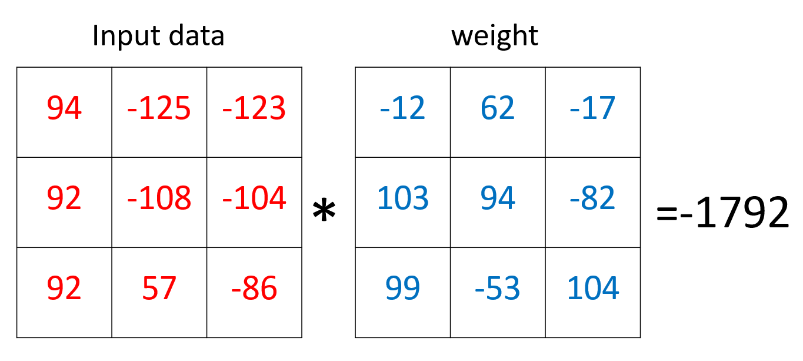
Below is the data of testbench1 and the results after convolution.

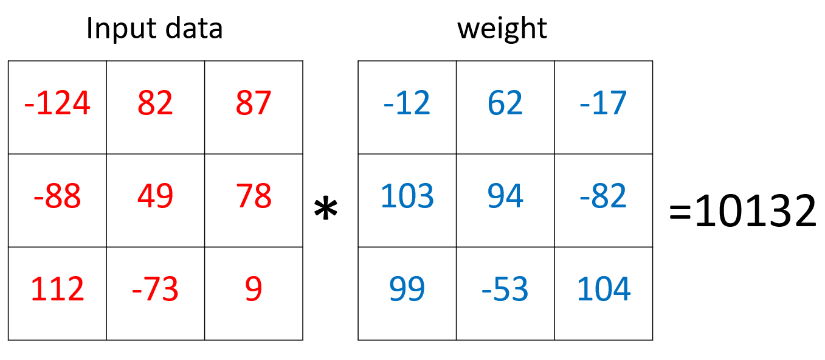


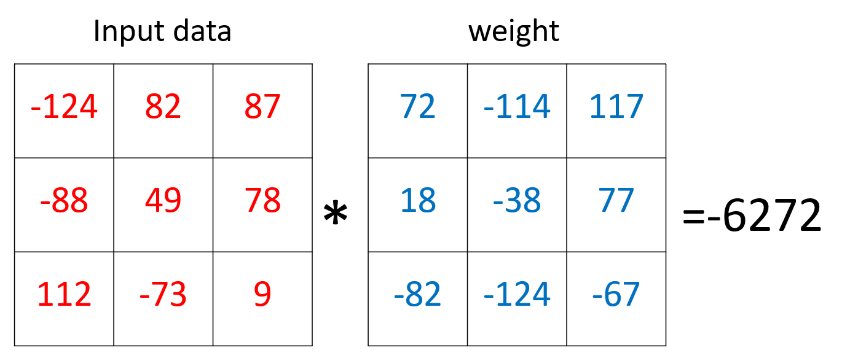




Below is the data of testbench2 and the results after convolution.







1. Show SuperLint coverage

|  |
| --- |
| Simulation result on the terminal |
| 一張含有 桌 的圖片  自動產生的描述一張含有 桌 的圖片  自動產生的描述 |
| Your waveform |
| Testbench1    Testbench2 |
| Explanation of your waveform |
| 沒修相關課程，大概有去看[神經網路科普影片](https://www.youtube.com/watch?v=aircAruvnKk)。但這題講白了這題就是把兩個矩陣的個別元素相乘，而對我來說難點在負數相乘要先做sign extension，而我的解題思路為  1.個別輸入連到array上方便一次用for loop處理，有4種輸入的情況w\_w和if\_w皆為1，個別為1與都為0  2.用for loop把array每一項個別處理  3.把結果跟0位元cascade到17位，再做sign extension  4.最後再乘得結果  5. **R**ectified **L**inear **U**nit函數映射(線性整流函數，活化函數主要目的是用來增加類神經網路模型的非線性) |
| SuperLint Coverage |
| 在COV.v檔中， |
| Synthesis |
| 一張含有 文字 的圖片  自動產生的描述 |

1. At last, please write the lesson you learned from Lab4

實驗1為模擬64 X 32 register file寫入、存取、讀出的狀況；實驗2為模擬自動販賣機的流程；實驗三為convolution的基本運算，從這幾次實驗慢慢發現這些實作內容十分貼近日常生活與上學期修的計算機組織的理論，希望我能跟上大家的腳步，這學期把實驗課所學所遇到的問體與解決方法銘記在心，提升coding與邏輯思考的能力。

Appendix A : Commands we will use to check your homework

