**National Cheng Kung University**

**Department of Electrical Engineering**

***Introduction to VLSI CAD (Spring 2024)***

**Lab Session 3**

**Design of ALU and Multiplication Using Verilog Coding**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Student ID | | |
| 游宗謀 | E94106151 | | |
| Practical Sections: | | Points | Marks |
| Prob A | | 30 |  |
| Prob B | | 30 |  |
| Prob C | | 20 |  |
| Report | | 15 |  |
| File hierarchy, naming…etc. | | 5 |  |
| Notes | | | |

**Due Date: 15:00, March 13, 2024 @ moodle**

**Deliverables**

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

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

1. All homework requirements should be uploaded in this file hierarchy or you will not get the full credit.

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

1. Important! TA will use the command in Appendix A to check your design under SoC Lab environment, if your code can not be recompiled by TA successfully using the commands, you will not get the 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. **File hierarchy should not be changed; it may cause** your code can not be recompiled by TA successfully using the autograding commands

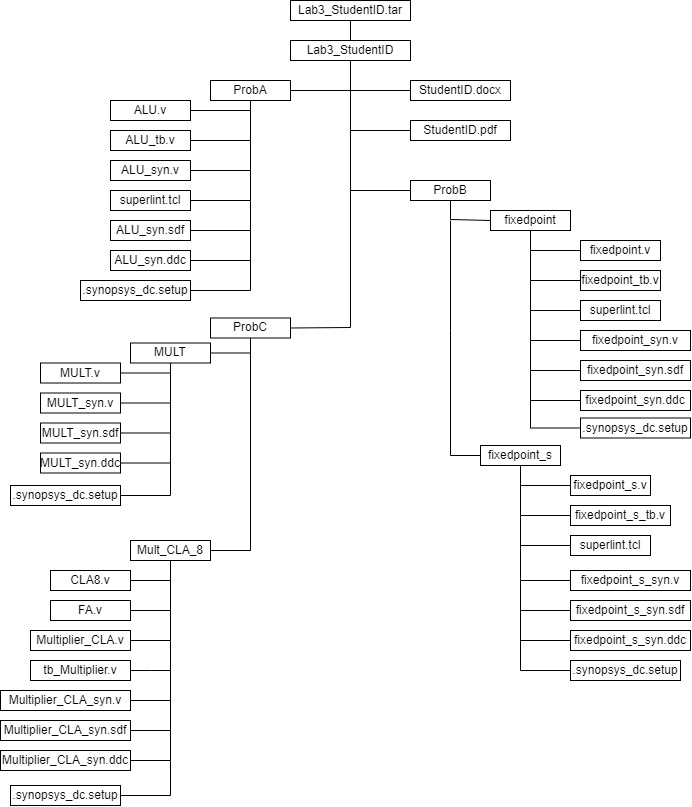
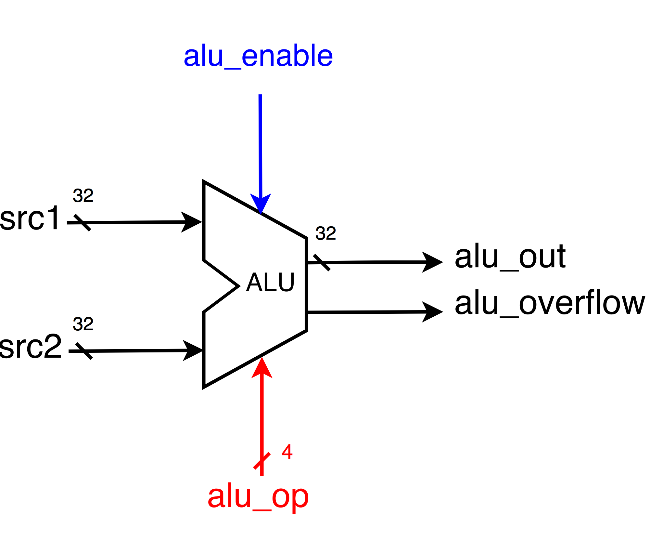


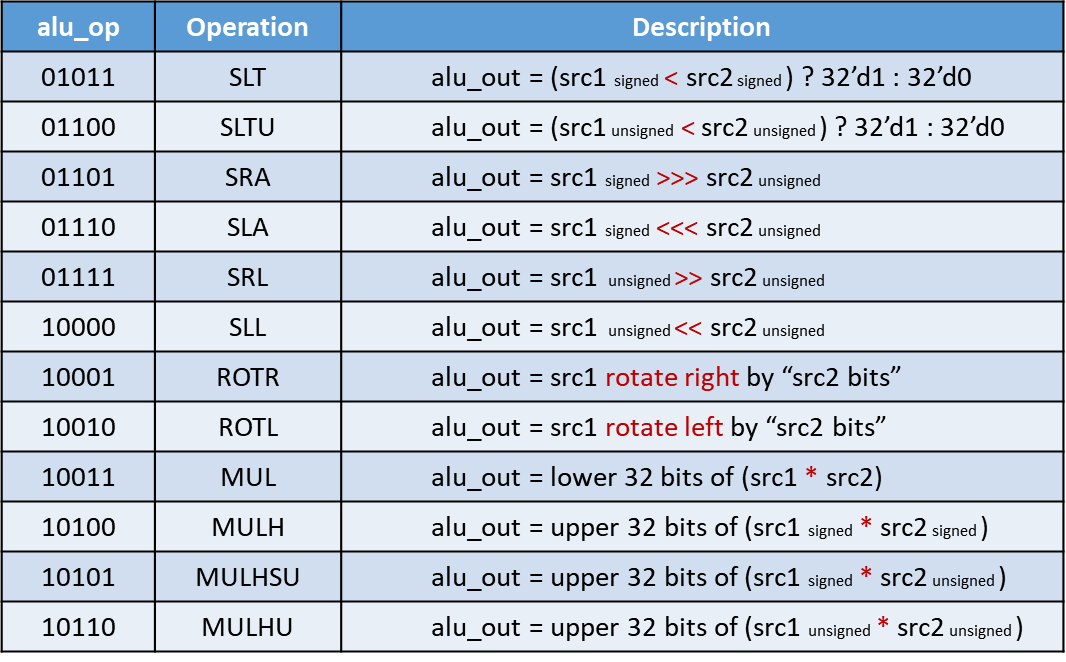
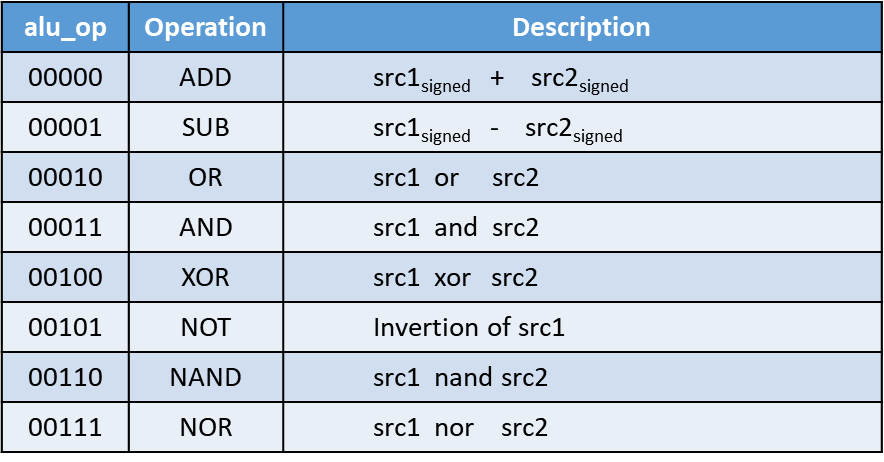
Fig.1 File hierarchy for Homework submission

Prob A: Arithmetic Logic Unit

**Design your Verilog code with the following specifications:**



1. Based on the reference code, please implement the following operations.



* 1. The frame code and testbench are given. Follow the frame code to finish this homework. The decimal part should be rounded.
  2. Follow the PPT file to synthesize your code.

**After you synthesize your design, you may have some information about the circuit. Fill in the following form.**

|  |  |  |
| --- | --- | --- |
| **Timing (slack)** | **Area (total cell area)** | **Power (total)** |
| **0.00** | **3966.122973** | **3.7416** |

**Please attach your design waveforms.**

|  |
| --- |
| Your simulation result on the terminal. |
|  |
| Your waveform (RTL & Synthesis) : |
| RTL  Synthesis |
| SuperLint Coverage |
| Coverage = (1-(2/79))\*100% = 97.47% |

Prob B-1: Practice fixed point

**Design your Verilog code with the following specifications:** Number format: unsigned numbers.

* 1. The frame code and testbench are given. Follow the frame code to finish this homework. The decimal part should be rounded.
  2. Follow the PPT file to synthesize your code.

**After you synthesize your design, you may have some information about the circuit. Fill in the following form.**

|  |  |  |
| --- | --- | --- |
| **Timing (slack)** | **Area (total cell area)** | **Power (total)** |
| **0.47** | **79.418881** | **5.7195e-02** |

**Please attach your design waveforms.**

|  |
| --- |
| Your simulation result on the terminal. |
|  |
| Your waveform (RTL & Synthesis) : |
| RTL  Synthesis |
| SuperLint Coverage |
| Coverage = (1-(0/16))\*100% = 100% |

Prob B-2: Practice fixed point (signed)

**Design your Verilog code with the following specifications:** Number format: signed numbers.

1. The frame code and testbench are given. Follow the frame code to finish this homework. The decimal part should be rounded.
2. Follow the PPT file to synthesize your code.

**After you synthesize your design, you may have some information about the circuit. Fill in the following form**

|  |  |  |
| --- | --- | --- |
| **Timing (slack)** | **Area (total cell area)** | **Power (total)** |
| **0.3** | **392.739848** | **0.2848** |

**Please attach your design waveforms.**

|  |
| --- |
| Your simulation result on the terminal. |
|  |
| Your waveform (RTL & Synthesis) : |
| RTL  Synthesis |
| SuperLint Coverage |
| Coverage = (1-(0/88))\*100% = 100% |

Prob C: Performance comparison

**Synthesize the 8\*8-bit CLA multiplier implemented in Lab2 and the given 8\*8-bit multiplier separately.**

You should answer the following questions:

**1. Determine the lowest achievable clock period for both, along with the corresponding area and power consumption.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Clock period** | **Timing (slack)** | **Area (total cell area)** | **Power (total)** |
| **CLA multiplier** | **0.43** | **0.00** | **245.566085** | **5.1816e-02** |
| **“\*”operator** | **0.17** | **0.00** | **151.735682** | **6.0849e-02** |

**2. Considering clock period and area, which structure has the better performance.**

**根據Clock Period和Area我們可以發現用“\*”operator 8\*8-bit multiplier的performance比較好。**

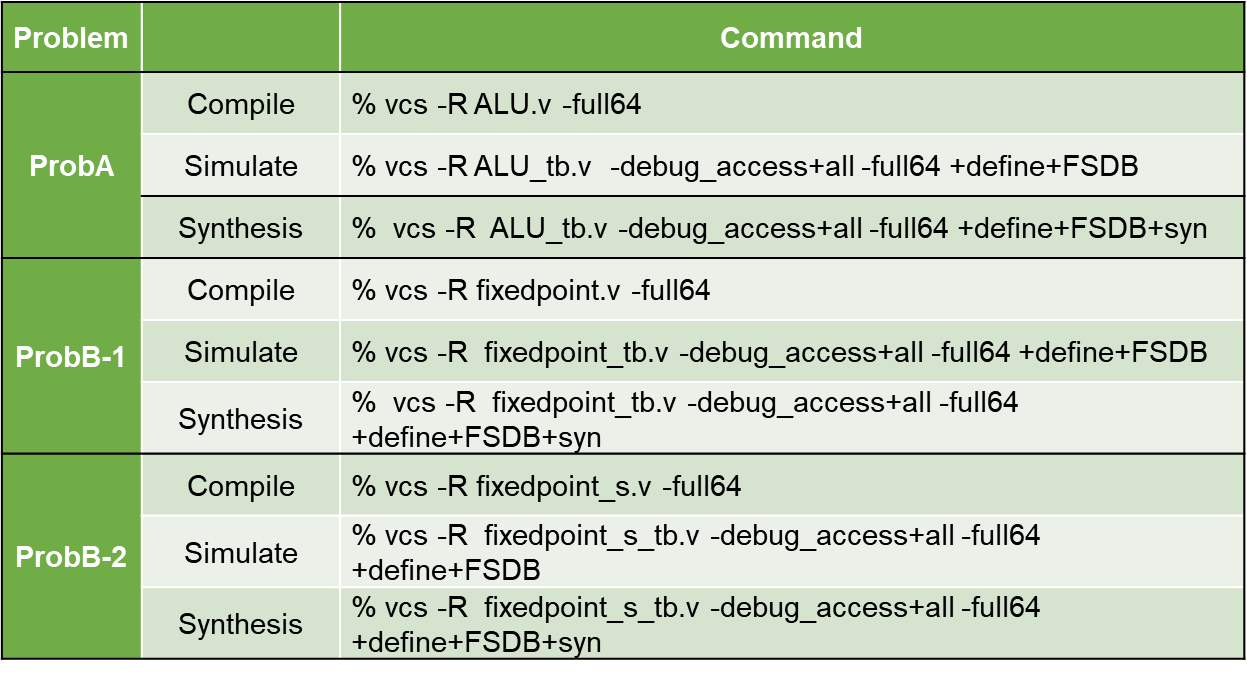
**因為在乘法器中加法器的使用頻率最高，會影響到整體的執行速度。普通的加法器中進位的延遲加總時間跟輸入的bit數成正比，所以用它組成的乘法器會有較大的延遲。而透過CLA Adder能將每一級的進位平行處理，藉此減少大量的延遲時間，進而提升整體的performance。但即便如此，在兩者都有用tool進行優化的情況下，“\*”operator 8\*8-bit multiplier的performance比較好，因為他的優化相比由我們先行優化的CLA更沒有限制。不過8\*8-bit CLA multiplier的Power卻比“\*”operator 8\*8-bit multiplier來的更小。**

**At last, please write the lessons learned from this lab session, or some suggestions for this lab session. Thank you.**

**在這次的lab中我學到$signed()的用法，這是我之前完全沒用過的寫法，因為之前寫要求的電路輸出都是以無號數為主。學會這個方法讓我可以不用寫一長串判斷式去檢查正負，減少很多要打的字數，但相對的在使用上也要格外小心，要注意邏輯和用法，才能不會出錯。**

**還有這次我也學到了如何使用合成的tool，我想這是非常難得的機會，想必可能連某些規模比較小的IC設計公司也沒機會用到這些工具吧，非常感謝這門課程讓我有機會使用這些工具。**

**此外很感謝其他同學在moodle上的提問、還有助教們總是及時的回答給與我們協助，讓我不用額外約時間打擾助教詢問作業問題。**



Appendix A : Commands we will use to check your homework