National Cheng Kung University

Department of Electrical Engineering

Introduction to VLSI CAD (Spring 2022)

Lab Session 3

Design of ALU and Fixed point Multiplication

Name	Student ID	
鄭智宇	E94096110	
Practical Sections:	Points	Marks
Prob A	20	
Prob B	30	
Prob C	30	
Report	15	
File hierarchy, naming…etc.	5	
Notes		

Due Date: 15:00, March 08, 2023 @ 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!
2) All homework requirements should be uploaded in this file hierarchy or you will not get the full credit.
   NOTE: Please **DO NOT** upload waveforms!
3) 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.
4) If you upload a dead body which we can't even compile you will get **NO** credit!
5) All Verilog file should get at least 90% superLint Coverage.
6) File hierarchy should not be changed; it may cause your code can not be recompiled by TA successfully using the autograding commands

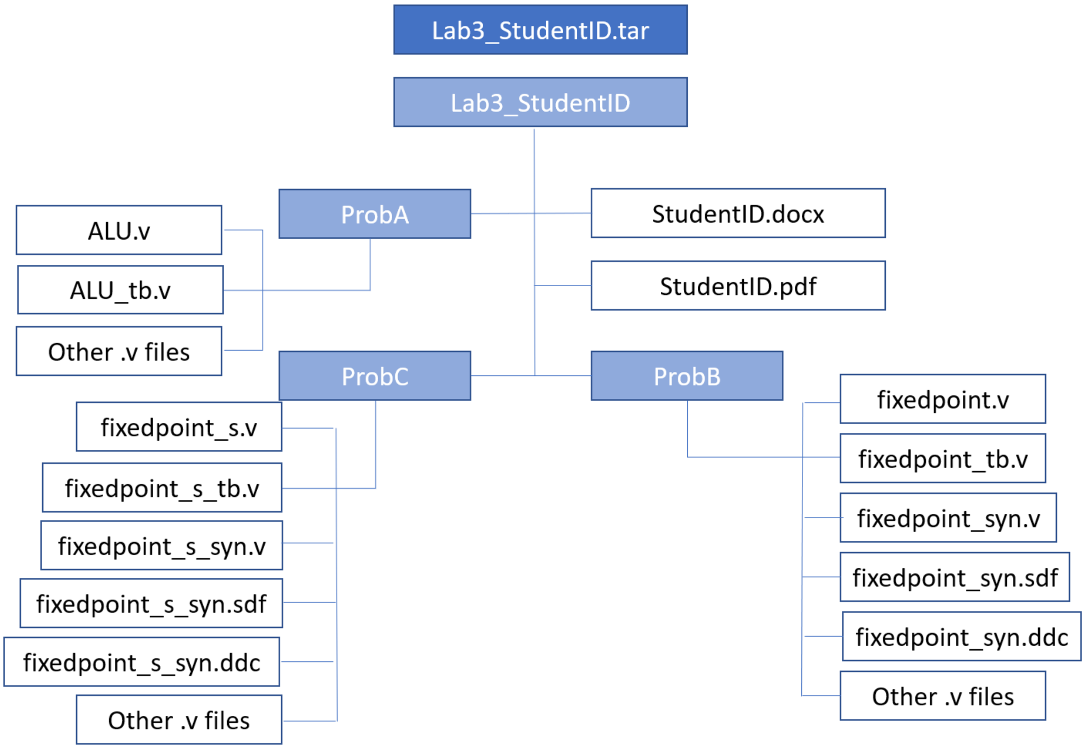
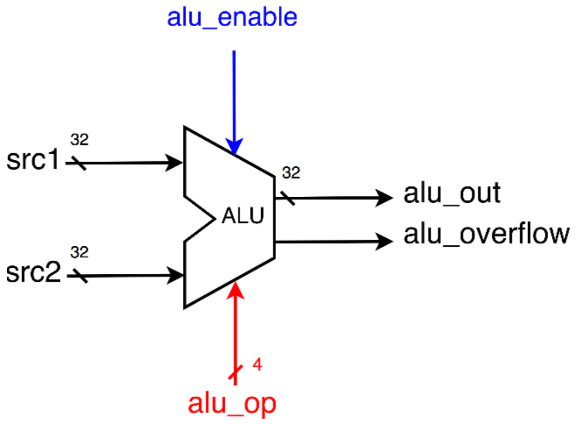


Fig.1 File hierarchy for Homework submission

Objectives:

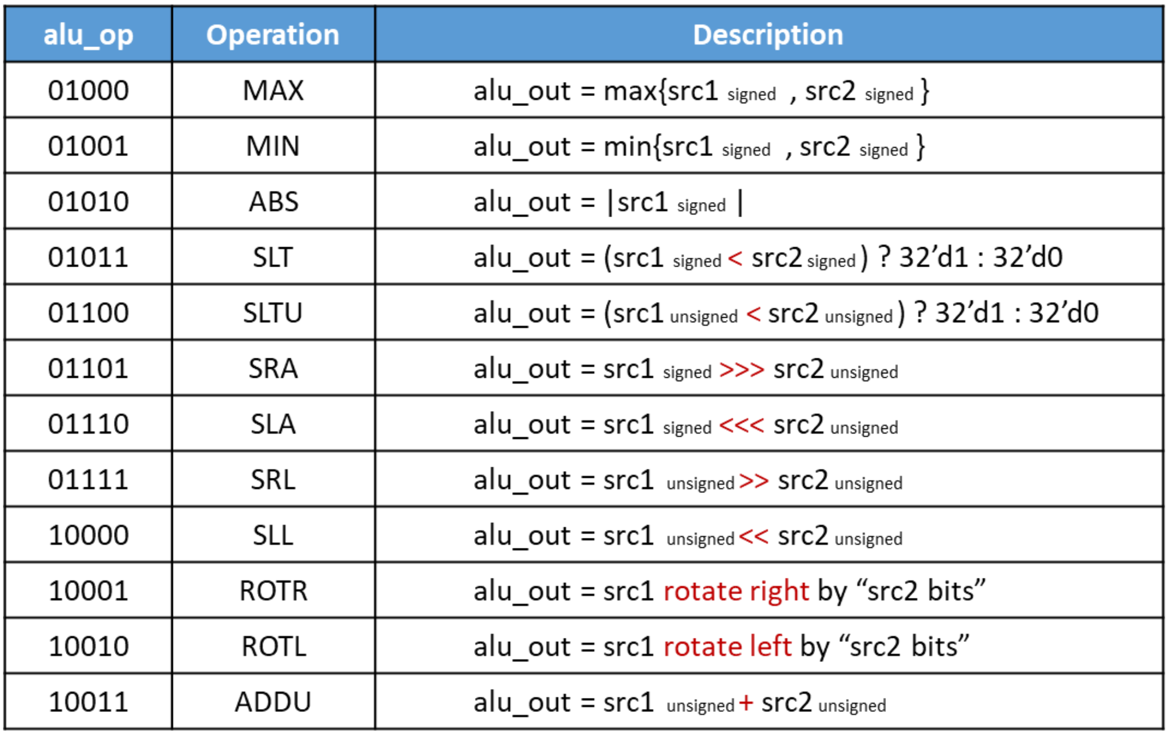
Learn how to design an ALU.

Prob A: Arithmetic Logic Unit



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

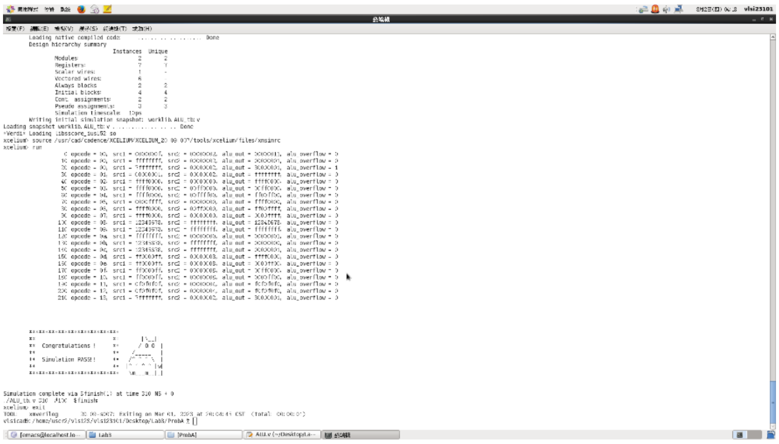
alu_op	Operation	Description
00000	ADD	alu_out = src1 $_{signed}$ + src2 $_{signed}$
00001	SUB	alu_out = src1 $_{signed}$ - src2 $_{signed}$
00010	OR	alu_out = src1 | src2
00011	AND	alu_out = src1 & src2
00100	XOR	alu_out = src1 ^ src2
00101	NOT	alu_out = ~src1
00110	NAND	alu_out = ~(src1 & src2)
00111	NOR	alu_out = ~(src1 | src2)



alu_op	Operation	Description
01000	MAX	alu_out = max{$src1_{\text{signed}}$, $src2_{\text{signed}}$ }
01001	MIN	alu_out = min{$src1_{\text{signed}}$, $src2_{\text{signed}}$ }
01010	ABS	alu_out = $\lvert src1_{\text{signed}} \rvert$
01011	SLT	alu_out = ($src1_{\text{signed}} < src2_{\text{signed}}$) ? 32’d1 : 32’d0
01100	SLTU	alu_out = ($src1_{\text{unsigned}} < src2_{\text{unsigned}}$) ? 32’d1 : 32’d0
01101	SRA	alu_out = $src1_{\text{signed}}$ >>> $src2_{\text{unsigned}}$
01110	SLA	alu_out = $src1_{\text{signed}}$ <<< $src2_{\text{unsigned}}$
01111	SRL	alu_out = $src1_{\text{unsigned}}$ >> $src2_{\text{unsigned}}$
10000	SLL	alu_out = $src1_{\text{unsigned}}$ << $src2_{\text{unsigned}}$
10001	ROTR	alu_out = src1 rotate right by “src2 bits”
10010	ROTL	alu_out = src1 rotate left by “src2 bits”
10011	ADDU	alu_out = $src1_{\text{unsigned}} + src2_{\text{unsigned}}$

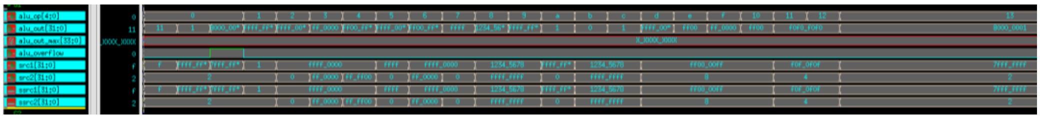
- **Please attach your design waveforms.**

Your simulation result on the terminal.



```
        Loading native compiled code:     .................. Done
        Design hierarchy summary
                                Instances  Unique
                Modules:                2       2
                Registers:              7       7
                Scalar wires:           1       -
                Vectored wires:         6       -
                Always blocks:          2       2
                Initial blocks:         4       4
                Cont. assignments:      2       2
                Pseudo assignments:     3       3
                Simulation timescale:  10ps
        Writing initial simulation snapshot: worklib.ALU_tb:v
Loading snapshot worklib.ALU_tb:v .................... Done
*Verdi* Loading libsscore_ius152.so
xcelium> source /usr/cad/cadence/XCELIUM/XCELIUM_20.09.007/tools/xcelium/files/xmsimrc
xcelium> run
                  0 opcode = 00, src1 = 0000000f, src2 = 00000002, alu_out = 00000011, alu_overflow = 0
                 10 opcode = 00, src1 = ffffffff, src2 = 00000002, alu_out = 00000001, alu_overflow = 0
                 20 opcode = 00, src1 = 7fffffff, src2 = 00000002, alu_out = 80000001, alu_overflow = 1
                 30 opcode = 01, src1 = 00000001, src2 = 00000002, alu_out = ffffffff, alu_overflow = 0
                 40 opcode = 02, src1 = ffff0000, src2 = 00000000, alu_out = ffff0000, alu_overflow = 0
                 50 opcode = 03, src1 = ffff0000, src2 = 00ff0000, alu_out = 00ff0000, alu_overflow = 0
                 60 opcode = 04, src1 = ffff0000, src2 = 00ffff00, alu_out = ff00ff00, alu_overflow = 0
                 70 opcode = 05, src1 = 0000ffff, src2 = 00000000, alu_out = ffff0000, alu_overflow = 0
                 80 opcode = 06, src1 = ffff0000, src2 = 00ff0000, alu_out = ff00ffff, alu_overflow = 0
                 90 opcode = 07, src1 = ffff0000, src2 = 00000000, alu_out = 0000ffff, alu_overflow = 0
                100 opcode = 08, src1 = 12345678, src2 = ffffffff, alu_out = 12345678, alu_overflow = 0
                110 opcode = 09, src1 = 12345678, src2 = ffffffff, alu_out = ffffffff, alu_overflow = 0
                120 opcode = 0a, src1 = ffffffff, src2 = 00000000, alu_out = 00000001, alu_overflow = 0
                130 opcode = 0b, src1 = 12345678, src2 = ffffffff, alu_out = 00000000, alu_overflow = 0
                140 opcode = 0c, src1 = 12345678, src2 = ffffffff, alu_out = 00000001, alu_overflow = 0
                150 opcode = 0d, src1 = ff0000ff, src2 = 00000008, alu_out = ffff0000, alu_overflow = 0
                160 opcode = 0e, src1 = ff0000ff, src2 = 00000008, alu_out = 00ff0000, alu_overflow = 0
                170 opcode = 0f, src1 = ff0000ff, src2 = 00000008, alu_out = 00ff0000, alu_overflow = 0
                180 opcode = 10, src1 = ff0000ff, src2 = 00000008, alu_out = 0000ff00, alu_overflow = 0
                190 opcode = 11, src1 = cf0f0f0f, src2 = 00000004, alu_out = fcf0f0fc, alu_overflow = 0
                200 opcode = 12, src1 = cf0f0f0f, src2 = 00000004, alu_out = fcf0f0fc, alu_overflow = 0
                210 opcode = 13, src1 = 7fffffff, src2 = 00000002, alu_out = 80000001, alu_overflow = 0



        **************************************
        **                          **        |\__|
        **  Congratulations !       **       / O O |
        **                          **      /_____ |
        **  Simulation PASS!!       **     /^ ^ ^ \ |
        **                          **    |^ ^ ^ ^ |w|
        **************************************    \m___m__|_|


Simulation complete via $finish(1) at time 310 NS + 0
./ALU_tb.v:310     #100  $finish;
xcelium> exit
TOOL:   xmverilog      20.09-s007: Exiting on Mar 01, 2023 at 20:04:45 CST  (total: 00:00:01)
```

Your waveform :

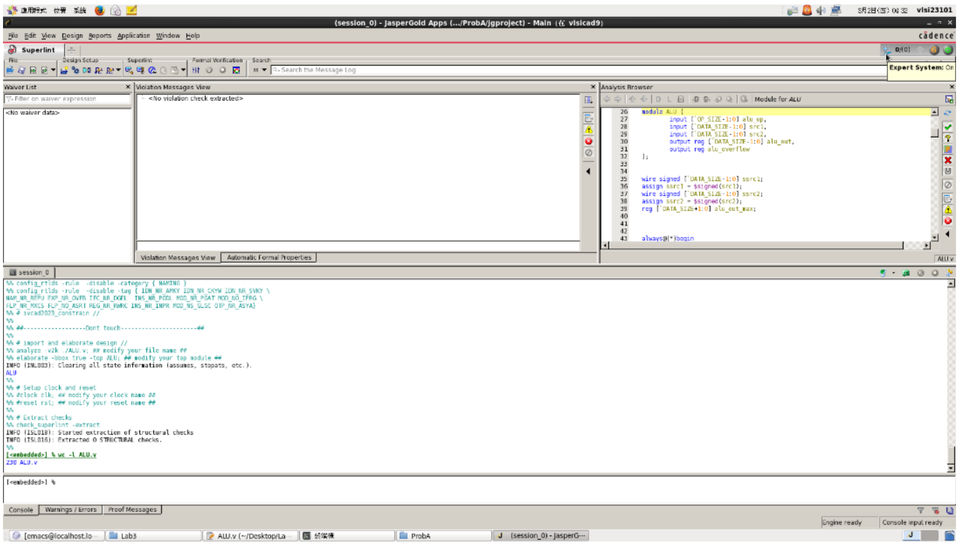


Step1 : ssrc1 = src1 的 signed 型態

ssrc2 = src2 的 signed 型態

Step2 : 利用 ssrc1, src1, ssrc2, src2, alu_op 判斷 alu_overflow, alu_out

SuperLint Coverage



(session_0) - JasperGold Apps (.../ProbA/jgproject) - Main (在 vlsicad9)

Expert System: On

Analysis Browser — Module for ALU

```
module ALU (
        input [`OP_SIZE-1:0] alu_op,
        input [`DATA_SIZE-1:0] src1,
        input [`DATA_SIZE-1:0] src2,
        output reg [`DATA_SIZE-1:0] alu_out,
        output reg alu_overflow
);


wire signed [`DATA_SIZE-1:0] ssrc1;
assign ssrc1 = $signed(src1);
wire signed [`DATA_SIZE-1:0] ssrc2;
assign ssrc2 = $signed(src2);
reg [`DATA_SIZE+1:0] alu_out_max;



always@(*)begin
```

Waiver List

<No waiver data>

Violation Messages View

<No violation check extracted>

```
%% config_rtlds -rule  -disable -category { NAMING }
%% config_rtlds -rule  -disable -tag { IDN_NR_AMKY IDN_NR_CKYW IDN_NR_SVKY \
NAM_NR_REPU EXP_NR_OVFB TFC_NR_DGEL  INS_NR_PODL MOD_NR_PGAT MOD_NO_IPRG \
FLP_NR_MXCS FLP_NO_ASRT REG_NR_RWRC INS_NR_INPR MOD_NS_GLGC OTP_NR_ASYA}
%% # ivcad2023_constrain //
%%
%% ##-----------------Dont touch---------------------##
%%
%% # import and elaborate design //
%% analyze -v2k ./ALU.v; ## modify your file name ##
%% elaborate -bbox true -top ALU; ## modify your top module ##
INFO (INL003): Clearing all state information (assumes, stopats, etc.).
ALU
%%
%% # Setup clock and reset
%% #clock clk; ## modify your clock name ##
%% #reset rst; ## modify your reset name ##
%%
%% # Extract checks
%% check_superlint -extract
INFO (ISL018): Started extraction of structural checks
INFO (ISL016): Extracted 0 STRUCTURAL checks.
%%
[<embedded>] % wc -l ALU.v
238 ALU.v
[<embedded>] %
```

0 warning

Coverage : 100%

Prob B: Practice fixed point

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

a. The frame code and testbench are given. Follow the frame code to finish this homework. The decimal part should be rounded.

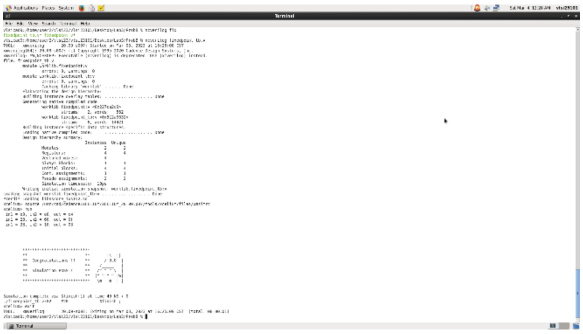
b. Follow the PPT file to synthesize your code.

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)**
19.16	**79.031998**	**23.4755 uW**

Please attach your design waveforms.

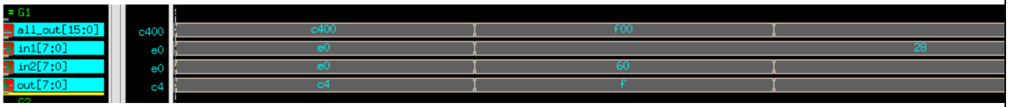
Your simulation result on the terminal.



```
ncsim> run
 in1 = e0, in2 = e0, out = c4
 in1 = 28, in2 = 60, out = [illegible]
 in1 = 28, in2 = 18, out = [illegible]

        **************************
        **                      **
        **  Congratulations !!  **
        **                      **
        **  Simulation PASS!!   **
        **                      **
        **************************
```

Your waveform (RTL & Synthesis) :

RTL:

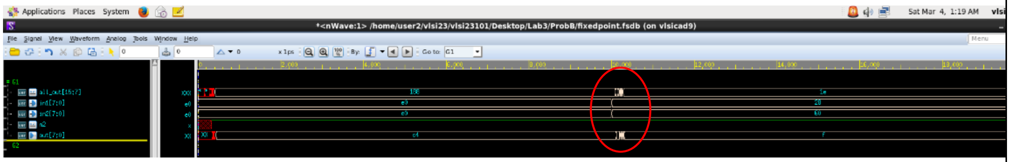


Step1 : all_out (16bits) = in1(8bits)*in2(8bits)

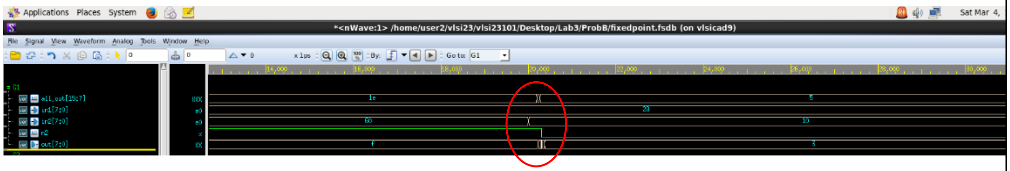
Step2 : out(8bits) = all_out[15:8] + 4'd1 (須進位)

= all_out[15:8] (不須進位)

Synthesis:



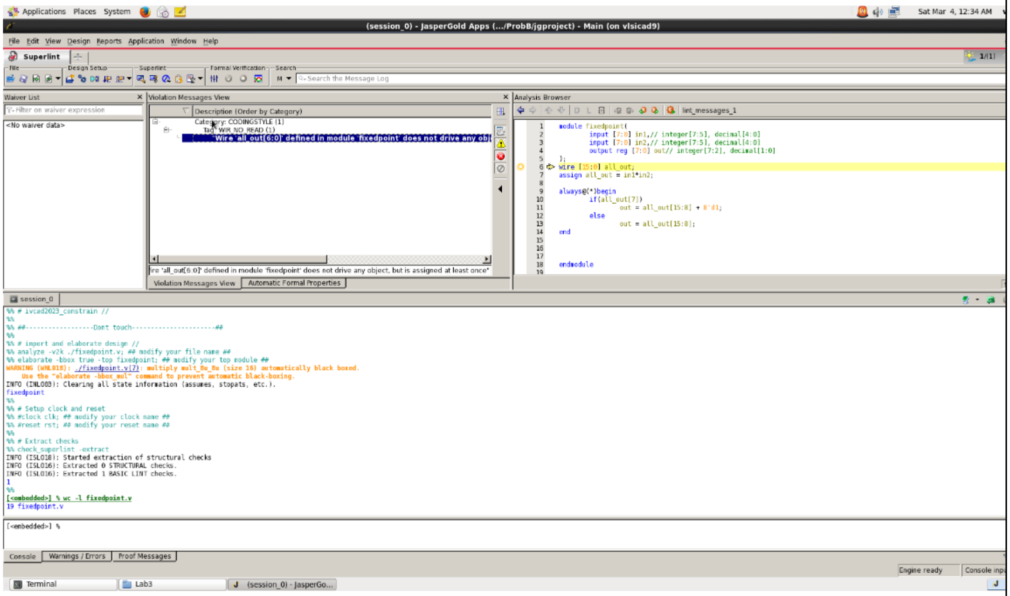
*<nWave:1> /home/user2/vlsi23/vlsi23101/Desktop/Lab3/ProbB/fixedpoint.fsdb (on vlsicad9)



*<nWave:1> /home/user2/vlsi23/vlsi23101/Desktop/Lab3/ProbB/fixedpoint.fsdb (on vlsicad9)

合成完後，在每個 step 間會出現延遲

SuperLint Coverage



```
Description (Order by Category)
Category: CODINGSTYLE (1)
  Tag: WIR_NO_READ (1)
    "Wire 'all_out[6:0]' defined in module 'fixedpoint' does not drive any obj
```

```
module fixedpoint(
        input [7:0] in1,// integer[7:5], decimal[4:0]
        input [7:0] in2,// integer[7:5], decimal[4:0]
        output reg [7:0] out// integer[7:2], decimal[1:0]
);
wire [15:0] all_out;
assign all_out = in1*in2;

always@(*)begin
        if(all_out[7])
                out = all_out[15:8] + 8'd1;
        else
                out = all_out[15:8];
end



endmodule
```

```
%% # ivcad2023_constrain //
%%
%% ##-----------------Dont touch--------------------##
%%
%% # import and elaborate design //
%% analyze -v2k ./fixedpoint.v; ## modify your file name ##
%% elaborate -bbox true -top fixedpoint; ## modify your top module ##
WARNING (WNL018): ./fixedpoint.v(7): multiply mult_8u_8u (size 16) automatically black boxed.
    Use the "elaborate -bbox_mul" command to prevent automatic black-boxing.
INFO (INL003): Clearing all state information (assumes, stopats, etc.).
fixedpoint
%%
%% # Setup clock and reset
%% #clock clk; ## modify your clock name ##
%% #reset rst; ## modify your reset name ##
%%
%% # Extract checks
%% check_superlint -extract
INFO (ISL018): Started extraction of structural checks
INFO (ISL016): Extracted 0 STRUCTURAL checks.
INFO (ISL016): Extracted 1 BASIC LINT checks.
1
%%
[<embedded>] % wc -l fixedpoint.v
19 fixedpoint.v

[<embedded>] %
```

1 warning

Coverage 約等於 95%

Prob C: Practice fixed point (signed)

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

a. The frame code and testbench are given. Follow the frame code to finish this homework. The decimal part should be rounded.

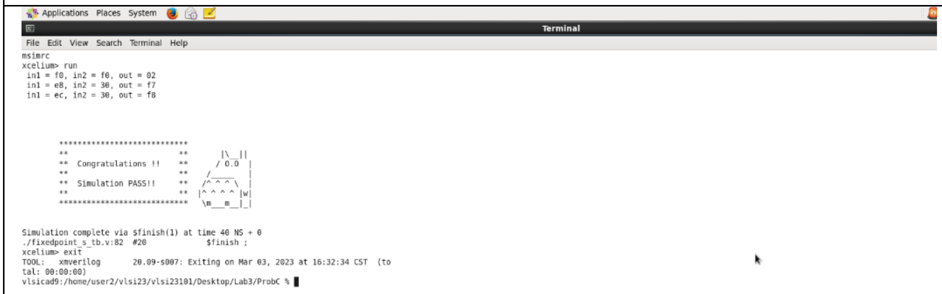
b. Follow the PPT file to synthesize your code.

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)**
18.96	**107.492398**	**32.1012 uW**

Please attach your design waveforms.

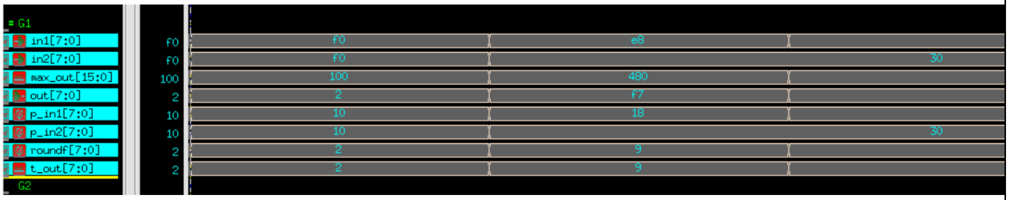
Your simulation result on the terminal.



```
msimrc
xcelium> run
 in1 = f0, in2 = f0, out = 02
 in1 = e8, in2 = 30, out = f7
 in1 = ec, in2 = 30, out = f8



        ****************************
        **                        **       |\__||
        **  Congratulations !!    **      / O.O  |
        **                        **    /_____   |
        **  Simulation PASS!!     **   /^ ^ ^ \  |
        **                        **  |^ ^ ^ ^ |w|
        ****************************   \m___m__|_|


Simulation complete via $finish(1) at time 40 NS + 0
./fixedpoint_s_tb.v:82  #20             $finish ;
xcelium> exit
TOOL:   xmverilog       20.09-s007: Exiting on Mar 03, 2023 at 16:32:34 CST  (to
tal: 00:00:00)
vlsicad9:/home/user2/vlsi23/vlsi23101/Desktop/Lab3/ProbC % 
```

Your waveform (RTL & Synthesis) :

RTL:



Step1： 統一將 in1 , in2 換成正數 p_in1 , p_in2

Step2 : max_out (16bits) = p_in1(8bits)*p_in2(8bits)

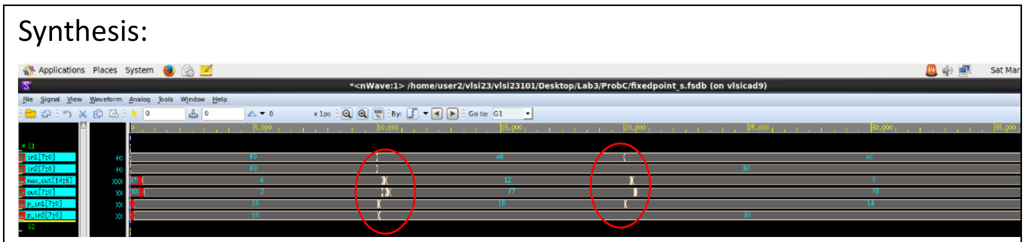
Step3 : t_out(8bit) = max_out[14:7] ,取出 sign[7], integer[6:1], decimal[0]

Step4 : roundf(8bits) = t_out(8bits) + 8'd1 (須進位)

= t_out(8bits) (不須進位)

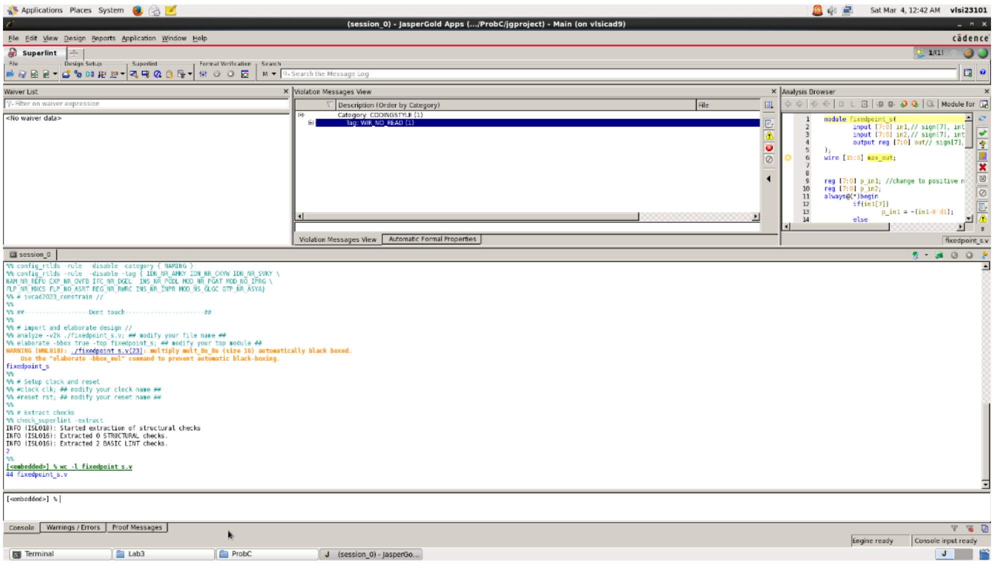
Step5 :out(8bits) = 將 t_out 轉為原本的 sign

Synthesis:


*<nWave:1> /home/user2/vlsi23/vlsi23101/Desktop/Lab3/ProbC/fixedpoint_s.fsdb (on vlsicad9)
G1
in1[7:0]
in2[7:0]
max_out[15:0]
out[7:0]
p_in1[7:0]
p_in2[7:0]

合成完後，在每個 step 間會出現延遲

SuperLint Coverage



```
module fixedpoint_s(
        input [7:0] in1,// sign[7], int
        input [7:0] in2,// sign[7], int
        output reg [7:0] out// sign[7],
);
wire [15:0] max_out;


reg [7:0] p_in1; //change to positive n
reg [7:0] p_in2;
always@(*)begin
        if(in1[7])
                p_in1 = -(in1-8'd1);
        else
```

```
%% config_rtlds -rule  -disable -category { NAMING }
%% config_rtlds -rule  -disable -tag { IDN_NR_AMKY IDN_NR_CKYW IDN_NR_SVKY \
NAM_NR_REPU EXP_NR_OVFB IFC_NR_DGEL  INS_NR_PODL MOD_NR_PGAT MOD_NO_IPRG \
FLP_NR_MXCS FLP_NO_ASRT REG_NR_RWRC INS_NR_INPR MOD_NS_GLGC OTP_NR_ASYA}
%% # ivcad2023_constrain //
%%
%% ##-------------------Dont touch---------------------##
%%
%% # import and elaborate design //
%% analyze -v2k ./fixedpoint_s.v; ## modify your file name ##
%% elaborate -bbox true -top fixedpoint_s; ## modify your top module ##
WARNING (WNL018): ./fixedpoint_s.v(23): multiply mult_8u_8u (size 16) automatically black boxed.
    Use the "elaborate -bbox_mul" command to prevent automatic black-boxing.
fixedpoint_s
%%
%% # Setup clock and reset
%% #clock clk; ## modify your clock name ##
%% #reset rst; ## modify your reset name ##
%%
%% # Extract checks
%% check_superlint -extract
INFO (ISL018): Started extraction of structural checks
INFO (ISL016): Extracted 0 STRUCTURAL checks.
INFO (ISL016): Extracted 2 BASIC LINT checks.
2
%%
[<embedded>] % wc -l fixedpoint_s.v
44 fixedpoint_s.v
```

```
[<embedded>] %
```

1 warning

Coverage 大約等於 98%

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

1. 學習到合成相關知識
2. **reg**、**wire** 的區別
3. 小數的 **binary** 表示方法

Appendix A : Commands we will use to check your homework

Problem		**Command**
ProbA	Compile	% ncverilog ALU.v
	Simulate	% ncverilog ALU_tb.v +define+FSDB +access+r
ProbB	Compile	% ncverilog fixedpoint.v
	Simulate	% ncverilog fixedpoint_tb.v +define+FSDB +access+r
	Synthesis	% ncverilog fixedpoint_tb.v +define+FSDB+syn +access+r
ProbC	Compile	% ncverilog fixedpoint_s.v
	Simulate	% ncverilog fixedpoint_s_tb.v +define+FSDB +access+r
	Synthesis	% ncverilog fixedpoint_s_tb.v +define+FSDB+syn +access+r