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!
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

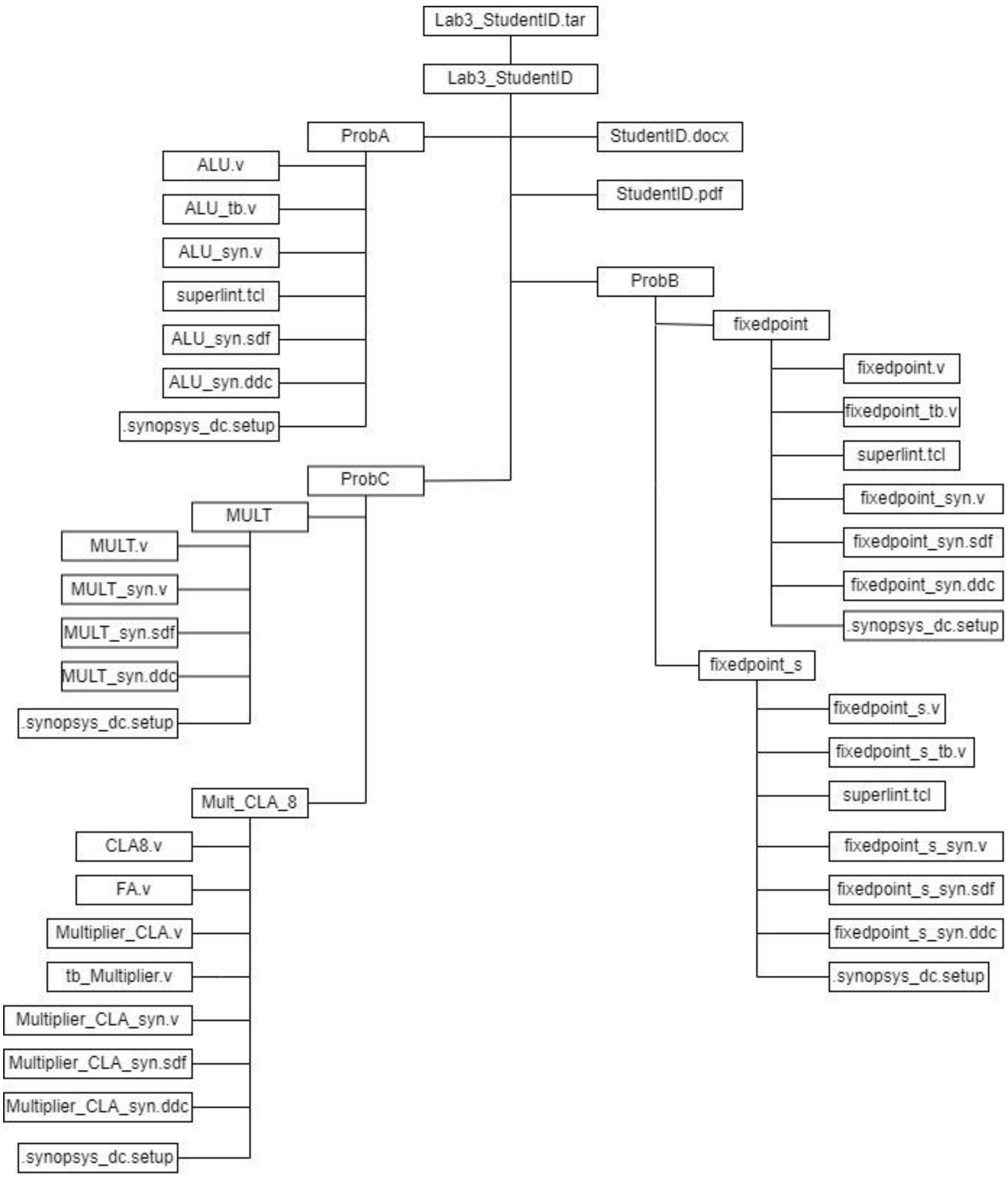
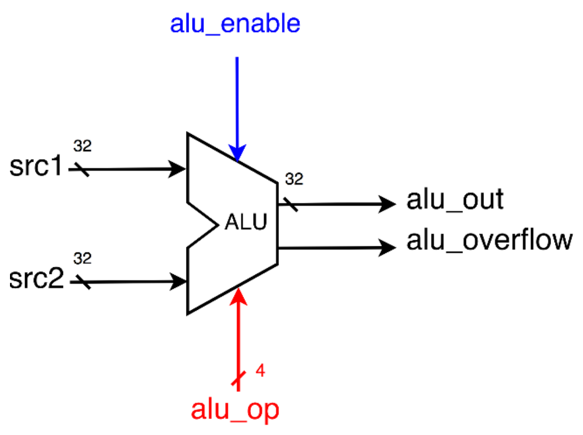


Fig.1 File hierarchy for Homework submission

Design your Verilog code with the following specifications:



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

alu_op	Operation	Description
00000	ADD	$src1_{signed}$ + $src2_{signed}$
00001	SUB	$src1_{signed}$ - $src2_{signed}$
00010	OR	src1 or src2
00011	AND	src1 and src2
00100	XOR	src1 xor src2
00101	NOT	Invertion of src1
00110	NAND	src1 nand src2
00111	NOR	src1 nor src2

alu_op	Operation	Description
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	MUL	alu_out = lower 32 bits of (src1 * src2)
10100	MULH	alu_out = upper 32 bits of (src1 $_{\text{signed}}$ * src2 $_{\text{signed}}$)
10101	MULHSU	alu_out = upper 32 bits of (src1 $_{\text{signed}}$ * src2 $_{\text{unsigned}}$)
10110	MULHU	alu_out = upper 32 bits of (src1 $_{\text{unsigned}}$ * src2 $_{\text{unsigned}}$)

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

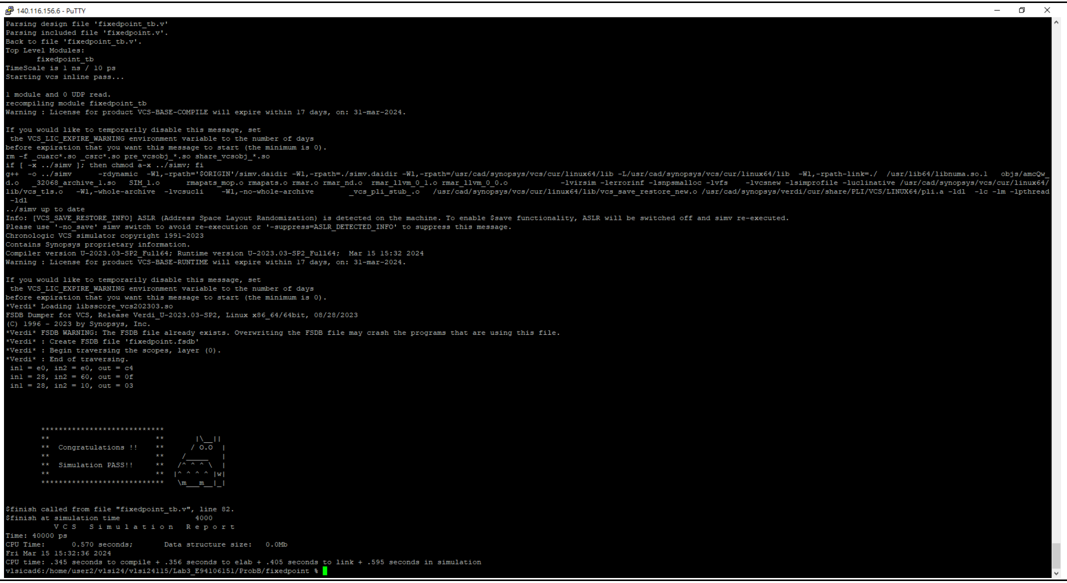
b. Follow the PPT file to synthesize your code.

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)**
0.00	**3966.122973**	**3.7416**

Please attach your design waveforms.

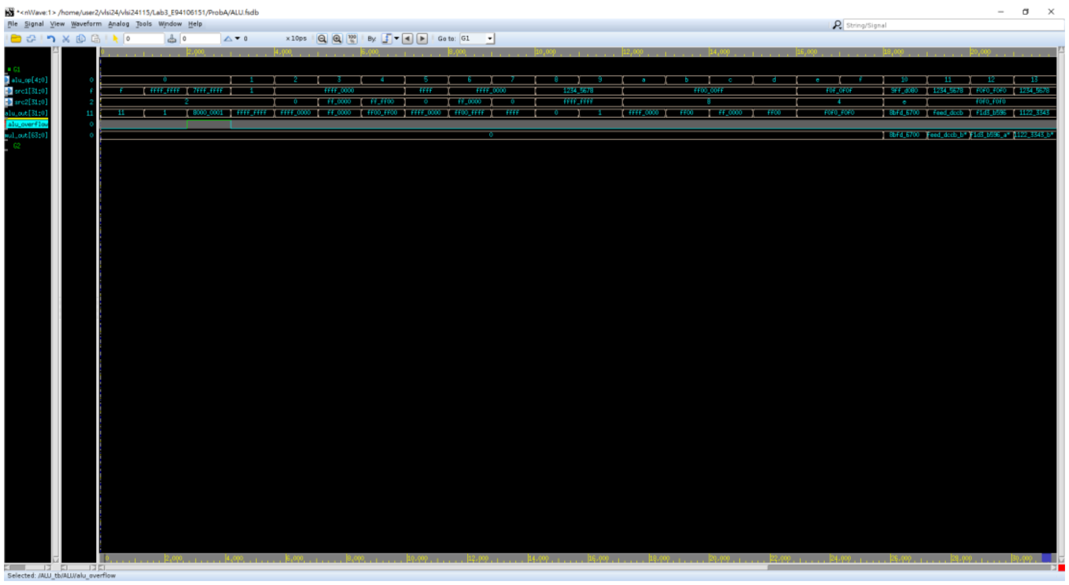
Your simulation result on the terminal.



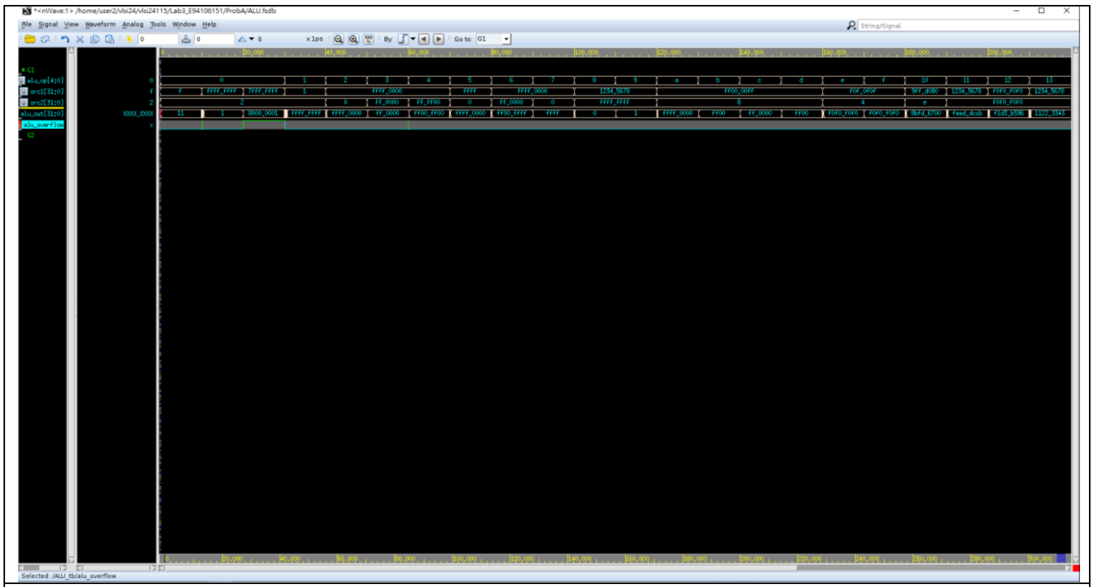
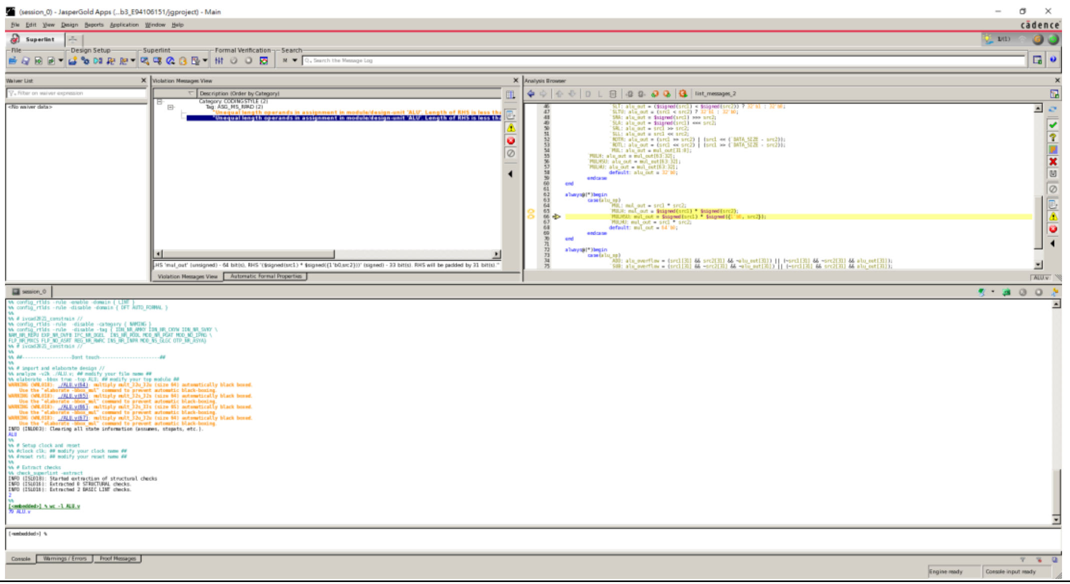
```
Parsing design file 'fixedpoint_tb.v'
Parsing included file 'fixedpoint.v'.
Back to file 'fixedpoint_tb.v'.
Top Level Modules:
       fixedpoint_tb
TimeScale is 1 ns / 10 ps
Starting vcs inline pass...

1 module and 0 UDP read.
recompiling module fixedpoint_tb
Warning : License for product VCS-BASE-COMPILE will expire within 17 days, on: 31-mar-2024.

If you would like to temporarily disable this message, set
 the VCS_LIC_EXPIRE_WARNING environment variable to the number of days
before expiration that you want this message to start (the minimum is 0).
rm -f _cuarc*.so _csrc*.so pre_vcsobj_*.so share_vcsobj_*.so
if [ -x ../simv ]; then chmod a-x ../simv; fi
g++  -o ../simv      -rdynamic  -Wl,-rpath='$ORIGIN'/simv.daidir -Wl,-rpath=./simv.daidir -Wl,-rpath=/usr/cad/synopsys/vcs/cur/linux64/lib -L/usr/cad/synopsys/vcs/cur/linux64/lib  -Wl,-rpath-link=./  /usr/lib64/libnuma.so.1   objs/amcQw_
d.o   _32068_archive_1.so   SIM_l.o      rmapats_mop.o rmapats.o rmar.o rmar_nd.o  rmar_llvm_0_1.o rmar_llvm_0_0.o           -lvirsim -lerrorinf -lsnpsmalloc -lvfs    -lvcsnew -lsimprofile -luclinative /usr/cad/synopsys/vcs/cur/linux64/
lib/vcs_tls.o   -Wl,-whole-archive  -lvcsucli    -Wl,-no-whole-archive        _vcs_pli_stub_.o   /usr/cad/synopsys/vcs/cur/linux64/lib/vcs_save_restore_new.o /usr/cad/synopsys/verdi/cur/share/PLI/VCS/LINUX64/pli.a -ldl  -lc -lm -lpthread
 -ldl
../simv up to date
Info: [VCS_SAVE_RESTORE_INFO] ASLR (Address Space Layout Randomization) is detected on the machine. To enable $save functionality, ASLR will be switched off and simv re-executed.
Please use '-no_save' simv switch to avoid re-execution or '-suppress=ASLR_DETECTED_INFO' to suppress this message.
Chronologic VCS simulator copyright 1991-2023
Contains Synopsys proprietary information.
Compiler version U-2023.03-SP2_Full64; Runtime version U-2023.03-SP2_Full64;  Mar 15 15:32 2024
Warning : License for product VCS-BASE-RUNTIME will expire within 17 days, on: 31-mar-2024.

If you would like to temporarily disable this message, set
 the VCS_LIC_EXPIRE_WARNING environment variable to the number of days
before expiration that you want this message to start (the minimum is 0).
*Verdi* Loading libsscore_vcs202303.so
FSDB Dumper for VCS, Release Verdi_U-2023.03-SP2, Linux x86_64/64bit, 08/28/2023
(C) 1996 - 2023 by Synopsys, Inc.
*Verdi* FSDB WARNING: The FSDB file already exists. Overwriting the FSDB file may crash the programs that are using this file.
*Verdi* : Create FSDB file 'fixedpoint.fsdb'
*Verdi* : Begin traversing the scopes, layer (0).
*Verdi* : End of traversing.
 in1 = e0, in2 = e0, out = c4
 in1 = 28, in2 = 60, out = 0f
 in1 = 28, in2 = 10, out = 03




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


$finish called from file "fixedpoint_tb.v", line 82.
$finish at simulation time                 4000
           V C S   S i m u l a t i o n   R e p o r t
Time: 40000 ps
CPU Time:      0.570 seconds;       Data structure size:   0.0Mb
Fri Mar 15 15:32:36 2024
CPU time: .345 seconds to compile + .356 seconds to elab + .405 seconds to link + .595 seconds in simulation
vlsicad6:/home/user2/vlsi24/vlsi24115/Lab3_E94106151/ProbB/fixedpoint %
```

Your waveform (RTL & Synthesis) :

RTL


*<nWave:1> /home/user2/vlsi24/vlsi24115/Lab3_E94106151/ProbA/ALU.fsdb
File Signal View Waveform Analog Tools Window Help
alu_op[4:0]
src1[31:0]
src2[31:0]
alu_out[31:0]
alu_overflow
mul_out[63:0]
Selected: /ALU_tb/ALU/alu_overflow

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.

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

d. Follow the PPT file to synthesize your code.

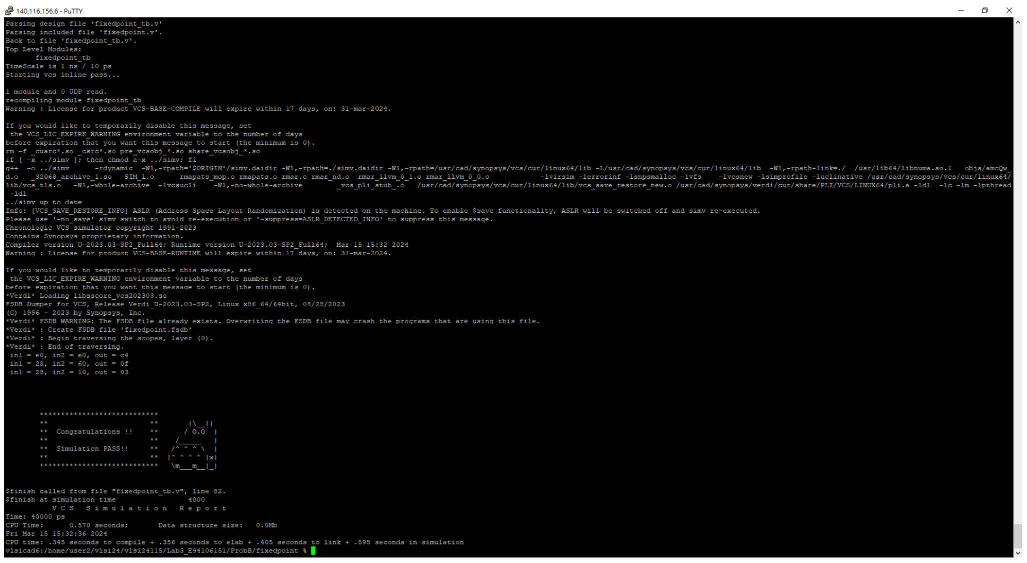
After you synthesize your design, you may have some information about the circuit. Fill in the following form.

Timing (slack)	**Area (total cell area)**	**Power (total)**

0.47	**79.418881**	**5.7195e-02**

Please attach your design waveforms.

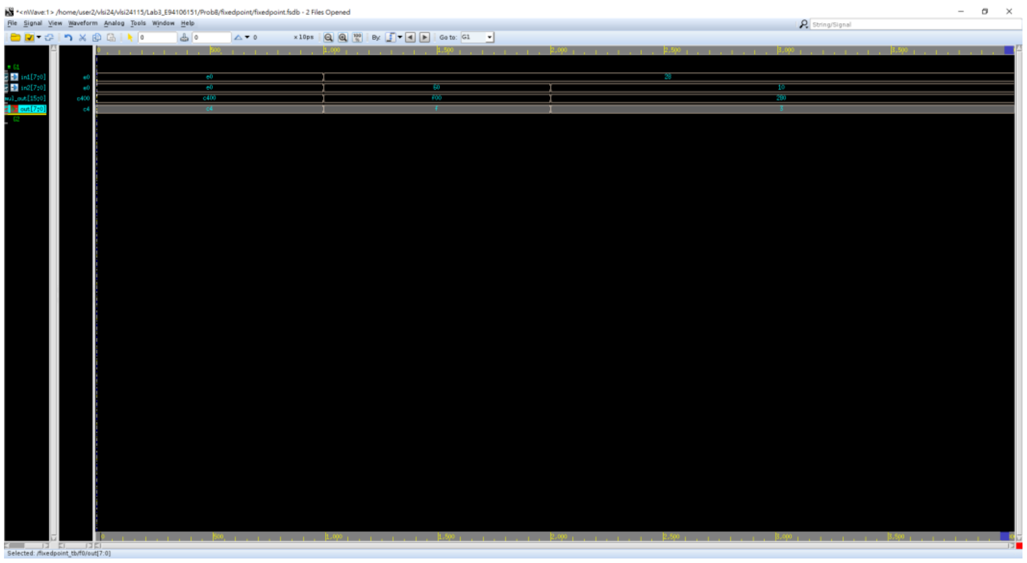
Your simulation result on the terminal.



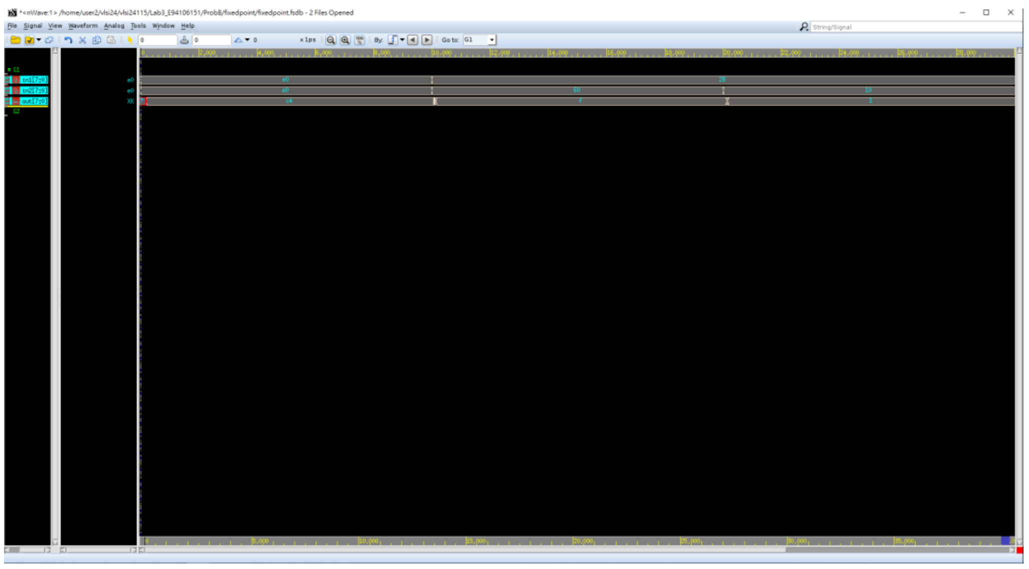
```
Parsing design file 'fixedpoint_tb.v'
Parsing included file 'fixedpoint.v'.
Back to file 'fixedpoint_tb.v'.
Top Level Modules:
       fixedpoint_tb
TimeScale is 1 ns / 10 ps
Starting vcs inline pass...

1 module and 0 UDP read.
recompiling module fixedpoint_tb
Warning : License for product VCS-BASE-COMPILE will expire within 17 days, on: 31-mar-2024.

If you would like to temporarily disable this message, set
 the VCS_LIC_EXPIRE_WARNING environment variable to the number of days
before expiration that you want this message to start (the minimum is 0).
rm -f _cuarc*.so _csrc*.so pre_vcsobj_*.so share_vcsobj_*.so
if [ -x ../simv ]; then chmod a-x ../simv; fi
g++  -o ../simv      -rdynamic  -Wl,-rpath='$ORIGIN'/simv.daidir -Wl,-rpath=./simv.daidir -Wl,-rpath=/usr/cad/synopsys/vcs/cur/linux64/lib -L/usr/cad/synopsys/vcs/cur/linux64/lib  -Wl,-rpath-link=./  /usr/lib64/libnuma.so.1   objs/amcQw_
d.o   _32068_archive_1.so   SIM_l.o      rmapats_mop.o rmapats.o rmar.o rmar_nd.o  rmar_llvm_0_1.o rmar_llvm_0_0.o            -lvirsim -lerrorinf -lsnpsmalloc -lvfs    -lvcsnew -lsimprofile -luclinative /usr/cad/synopsys/vcs/cur/linux64/
lib/vcs_tls.o   -Wl,-whole-archive  -lvcsucli    -Wl,-no-whole-archive        _vcs_pli_stub_.o   /usr/cad/synopsys/vcs/cur/linux64/lib/vcs_save_restore_new.o /usr/cad/synopsys/verdi/cur/share/PLI/VCS/LINUX64/pli.a -ldl  -lc -lm -lpthread
 -ldl
../simv up to date
Info: [VCS_SAVE_RESTORE_INFO] ASLR (Address Space Layout Randomization) is detected on the machine. To enable $save functionality, ASLR will be switched off and simv re-executed.
Please use '-no_save' simv switch to avoid re-execution or '-suppress=ASLR_DETECTED_INFO' to suppress this message.
Chronologic VCS simulator copyright 1991-2023
Contains Synopsys proprietary information.
Compiler version U-2023.03-SP2_Full64; Runtime version U-2023.03-SP2_Full64;  Mar 15 15:32 2024
Warning : License for product VCS-BASE-RUNTIME will expire within 17 days, on: 31-mar-2024.

If you would like to temporarily disable this message, set
 the VCS_LIC_EXPIRE_WARNING environment variable to the number of days
before expiration that you want this message to start (the minimum is 0).
*Verdi* Loading libsscore_vcs202303.so
FSDB Dumper for VCS, Release Verdi_U-2023.03-SP2, Linux x86_64/64bit, 08/28/2023
(C) 1996 - 2023 by Synopsys, Inc.
*Verdi* FSDB WARNING: The FSDB file already exists. Overwriting the FSDB file may crash the programs that are using this file.
*Verdi* : Create FSDB file 'fixedpoint.fsdb'
*Verdi* : Begin traversing the scopes, layer (0).
*Verdi* : End of traversing.
 in1 = e0, in2 = e0, out = c4
 in1 = 28, in2 = 60, out = 0f
 in1 = 28, in2 = 10, out = 03




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


$finish called from file "fixedpoint_tb.v", line 82.
$finish at simulation time                4000
           V C S   S i m u l a t i o n   R e p o r t
Time: 40000 ps
CPU Time:      0.570 seconds;       Data structure size:   0.0Mb
Fri Mar 15 15:32:36 2024
CPU time: .345 seconds to compile + .356 seconds to elab + .405 seconds to link + .595 seconds in simulation
vlsicad6:/home/user2/vlsi24/vlsi24115/Lab3_E94106151/ProbB/fixedpoint %
```

Your waveform (RTL & Synthesis) :

RTL

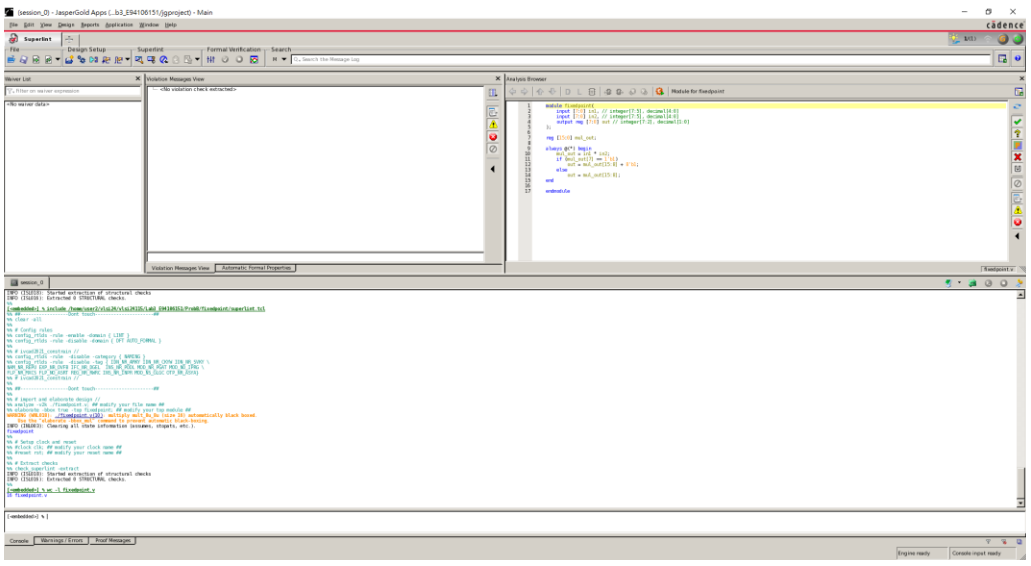

*<nWave:1> /home/user2/vlsi24/vlsi24115/Lab3_E94106151/ProbB/fixedpoint/fixedpoint.fsdb - 2 Files Opened
File Signal View Waveform Analog Tools Window Help
String/Signal
x 10ps
Go to: G1
500 1,000 1,500 2,000 2,500 3,000 3,500
G1
in1[7:0] e0 e0 28
in2[7:0] e0 e0 60 10
mul_out[15:0] c400 c400 f00 280
out[7:0] c4 c4 f 3
G2
Selected: /fixedpoint_tb/f0/out[7:0]

Synthesis



SuperLint Coverage

Coverage = (1-(0/16))*100% = 100%



```
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]
);

reg [15:0] mul_out;

always @(*) begin
    mul_out = in1 * in2;
    if (mul_out[7] == 1'b1)
        out = mul_out[15:8] + 8'b1;
    else
        out = mul_out[15:8];
end

endmodule
```

```
INFO (ISL018): Started extraction of structural checks
INFO (ISL016): Extracted 0 STRUCTURAL checks.
%%
[<embedded>] % include /home/user2/vlsi24/vlsi24115/Lab3_E94106151/ProbB/fixedpoint/superlint.tcl
%% ##-------------------Dont touch---------------------##
%% clear -all
%%
%% # Config rules
%% config_rtlds -rule -enable -domain { LINT }
%% config_rtlds -rule -disable -domain { DFT AUTO_FORMAL }
%%
%% # ivcad2021_constrain //
%% 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}
%% # ivcad2021_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(10): 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.
%%
[<embedded>] % wc -l fixedpoint.v
16 fixedpoint.v
```

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

Prob B-2: 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.

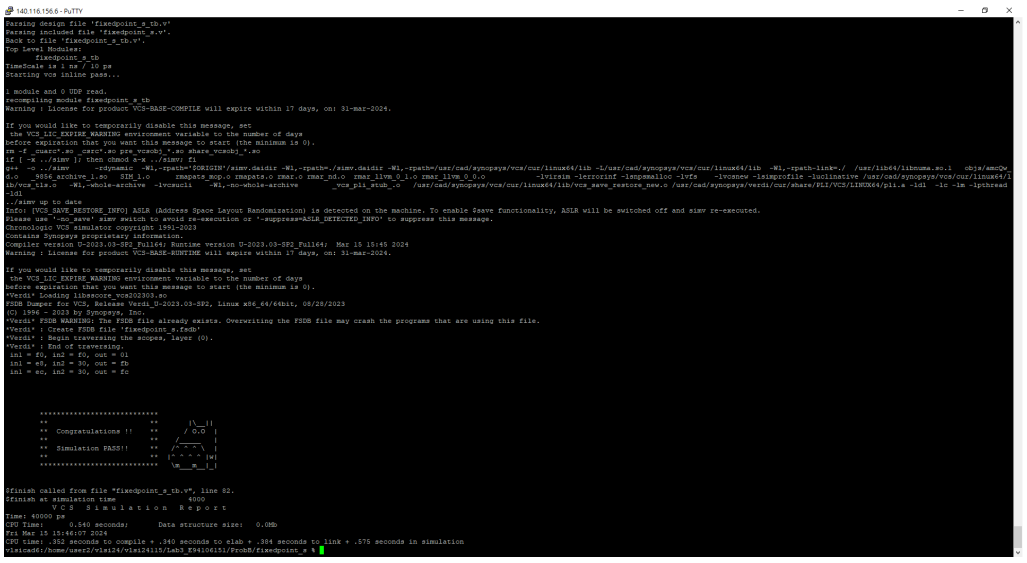
b. Follow the PPT file to synthesize your code.

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)**
0.3	**392.739848**	**0.2848**

Please attach your design waveforms.

Your simulation result on the terminal.

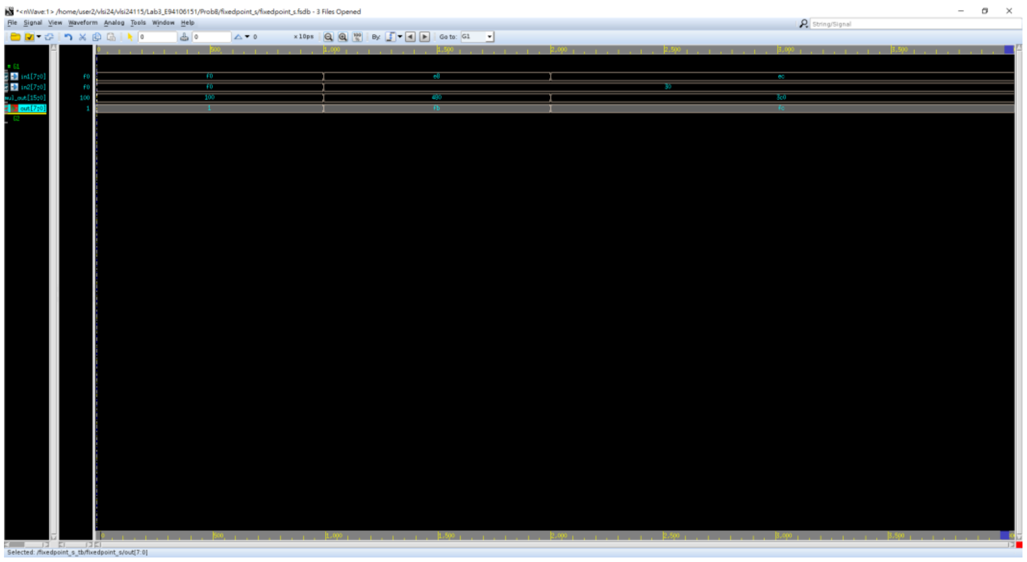


140.116.156.6 - PuTTY

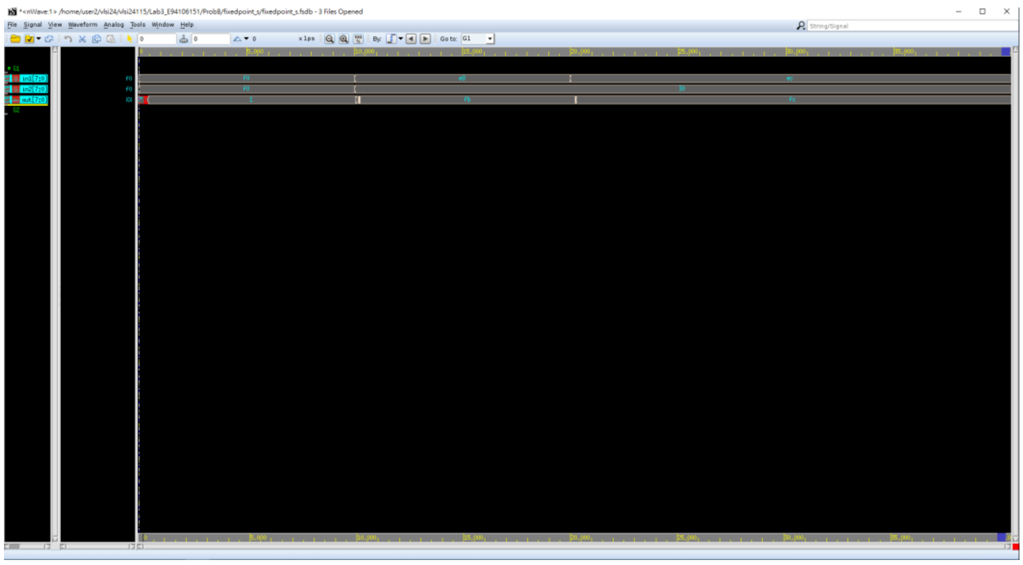
```
Parsing design file 'fixedpoint_s_tb.v'
Parsing included file 'fixedpoint_s.v'.
Back to file 'fixedpoint_s_tb.v'.
Top Level Modules:
       fixedpoint_s_tb
TimeScale is 1 ns / 10 ps
Starting vcs inline pass...

1 module and 0 UDP read.
recompiling module fixedpoint_s_tb
Warning : License for product VCS-BASE-COMPILE will expire within 17 days, on: 31-mar-2024.

If you would like to temporarily disable this message, set
 the VCS_LIC_EXPIRE_WARNING environment variable to the number of days
before expiration that you want this message to start (the minimum is 0).
rm -f _cuarc*.so _csrc*.so pre_vcsobj_*.so share_vcsobj_*.so
if [ -x ../simv ]; then chmod a-x ../simv; fi
g++  -o ../simv      -rdynamic  -Wl,-rpath='$ORIGIN'/simv.daidir -Wl,-rpath=./simv.daidir -Wl,-rpath=/usr/cad/synopsys/vcs/cur/linux64/lib -L/usr/cad/synopsys/vcs/cur/linux64/lib  -Wl,-rpath-link=./  /usr/lib64/libnuma.so.1   objs/amcQw_
d.o   _9856_archive_1.so   SIM_l.o      rmapats_mop.o rmapats.o rmar.o rmar_nd.o  rmar_llvm_0_1.o rmar_llvm_0_0.o           -lvirsim -lerrorinf -lsnpsmalloc -lvfs    -lvcsnew -lsimprofile -luclinative /usr/cad/synopsys/vcs/cur/linux64/l
ib/vcs_tls.o   -Wl,-whole-archive  -lvcsucli    -Wl,-no-whole-archive        _vcs_pli_stub_.o   /usr/cad/synopsys/vcs/cur/linux64/lib/vcs_save_restore_new.o /usr/cad/synopsys/verdi/cur/share/PLI/VCS/LINUX64/pli.a -ldl  -lc -lm -lpthread
-ldl
../simv up to date
Info: [VCS_SAVE_RESTORE_INFO] ASLR (Address Space Layout Randomization) is detected on the machine. To enable $save functionality, ASLR will be switched off and simv re-executed.
Please use '-no_save' simv switch to avoid re-execution or '-suppress=ASLR_DETECTED_INFO' to suppress this message.
Chronologic VCS simulator copyright 1991-2023
Contains Synopsys proprietary information.
Compiler version U-2023.03-SP2_Full64; Runtime version U-2023.03-SP2_Full64;  Mar 15 15:45 2024
Warning : License for product VCS-BASE-RUNTIME will expire within 17 days, on: 31-mar-2024.

If you would like to temporarily disable this message, set
 the VCS_LIC_EXPIRE_WARNING environment variable to the number of days
before expiration that you want this message to start (the minimum is 0).
*Verdi* Loading libsscore_vcs202303.so
FSDB Dumper for VCS, Release Verdi_U-2023.03-SP2, Linux x86_64/64bit, 08/28/2023
(C) 1996 - 2023 by Synopsys, Inc.
*Verdi* FSDB WARNING: The FSDB file already exists. Overwriting the FSDB file may crash the programs that are using this file.
*Verdi* : Create FSDB file 'fixedpoint_s.fsdb'
*Verdi* : Begin traversing the scopes, layer (0).
*Verdi* : End of traversing.
 in1 = f0, in2 = f0, out = 01
 in1 = e8, in2 = 30, out = fb
 in1 = ec, in2 = 30, out = fc



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


$finish called from file "fixedpoint_s_tb.v", line 82.
$finish at simulation time                 4000
           V C S   S i m u l a t i o n   R e p o r t
Time: 40000 ps
CPU Time:      0.540 seconds;       Data structure size:   0.0Mb
Fri Mar 15 15:46:07 2024
CPU time: .352 seconds to compile + .340 seconds to elab + .384 seconds to link + .575 seconds in simulation
vlsicad6:/home/user2/vlsi24/vlsi24115/Lab3_E94106151/ProbB/fixedpoint_s %
```

Your waveform (RTL & Synthesis) :

RTL

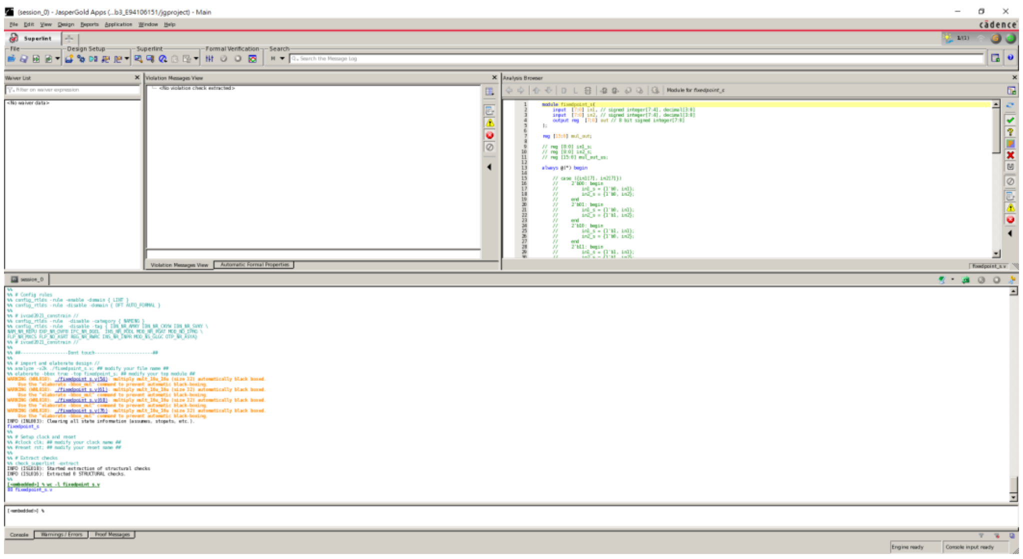

*<nWave:1> /home/user2/vlsi24/vlsi24115/Lab3_E94106151/ProbB/fixedpoint_s/fixedpoint_s.fsdb - 3 Files Opened
File Signal View Waveform Analog Tools Window Help
String/Signal
x 10ps
Go to: G1
G1
in1[7:0]
in2[7:0]
mul_out[15:0]
out[7:0]
G2
f0
f0
100
1
e8
30
480
fb
ec
3c0
fc
500
1,000
1,500
2,000
2,500
3,000
3,500
Selected: /fixedpoint_s_tb/fixedpoint_s/out[7:0]

Synthesis


*<nWave:1> /home/user2/vlsi24/vlsi24115/Lab3_E94106151/ProbB/fixedpoint_s/fixedpoint_s.fsdb - 3 Files Opened
in1[7:0]
in2[7:0]
out[7:0]
F0
e0
ec
30
1
Fb
Fc

SuperLint Coverage

Coverage = (1-(0/88))*100% = 100%



```verilog
module fixedpoint_s(
    input  [7:0] in1, // signed integer[7:4], decimal[3:0]
    input  [7:0] in2, // signed integer[7:4], decimal[3:0]
    output reg  [7:0] out // 8 bit signed integer[7:0]
);

reg [15:0] mul_out;

// reg [8:0] in1_s;
// reg [8:0] in2_s;
// reg [15:0] mul_out_us;

always @(*) begin

    // case ({in1[7], in2[7]})
    //     2'b00: begin
    //         in1_s = {1'b0, in1};
    //         in2_s = {1'b0, in2};
    //     end
    //     2'b01: begin
    //         in1_s = {1'b0, in1};
    //         in2_s = {1'b1, in2};
    //     end
    //     2'b10: begin
    //         in1_s = {1'b1, in1};
    //         in2_s = {1'b0, in2};
    //     end
    //     2'b11: begin
    //         in1_s = {1'b1, in1};
```

```
%%
%% # Config rules
%% config_rtlds -rule -enable -domain { LINT }
%% config_rtlds -rule -disable -domain { DFT AUTO_FORMAL }
%%
%% # ivcad2021_constrain //
%% 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}
%% # ivcad2021_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(58): multiply mult_16u_16u (size 32) automatically black boxed.
    Use the "elaborate -bbox_mul" command to prevent automatic black-boxing.
WARNING (WNL018): ./fixedpoint_s.v(61): multiply mult_16u_16u (size 32) automatically black boxed.
    Use the "elaborate -bbox_mul" command to prevent automatic black-boxing.
WARNING (WNL018): ./fixedpoint_s.v(68): multiply mult_16u_16u (size 32) automatically black boxed.
    Use the "elaborate -bbox_mul" command to prevent automatic black-boxing.
WARNING (WNL018): ./fixedpoint_s.v(76): multiply mult_16u_16u (size 32) automatically black boxed.
    Use the "elaborate -bbox_mul" command to prevent automatic black-boxing.
INFO (INL003): Clearing all state information (assumes, stopats, etc.).
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 (ISL036): Extracted 0 STRUCTURAL checks.
%%
[<embedded>] % wc -l fixedpoint_s.v
88 fixedpoint_s.v
[<embedded>] %
```

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。

Problem		Command
ProbA	Compile	% vcs -R ALU.v -full64
	Simulate	% vcs -R ALU_tb.v -debug_access+all -full64 +define+FSDB
	Synthesis	% vcs -R ALU_tb.v -debug_access+all -full64 +define+FSDB+syn
ProbB-1	Compile	% vcs -R fixedpoint.v -full64
	Simulate	% vcs -R fixedpoint_tb.v -debug_access+all -full64 +define+FSDB
	Synthesis	% vcs -R fixedpoint_tb.v -debug_access+all -full64 +define+FSDB+syn
ProbB-2	Compile	% vcs -R fixedpoint_s.v -full64
	Simulate	% vcs -R fixedpoint_s_tb.v -debug_access+all -full64 +define+FSDB
	Synthesis	% vcs -R fixedpoint_s_tb.v -debug_access+all -full64 +define+FSDB+syn

Appendix A : Commands we will use to check your homework