SOC LAB#4-2 Report

Group no: 4

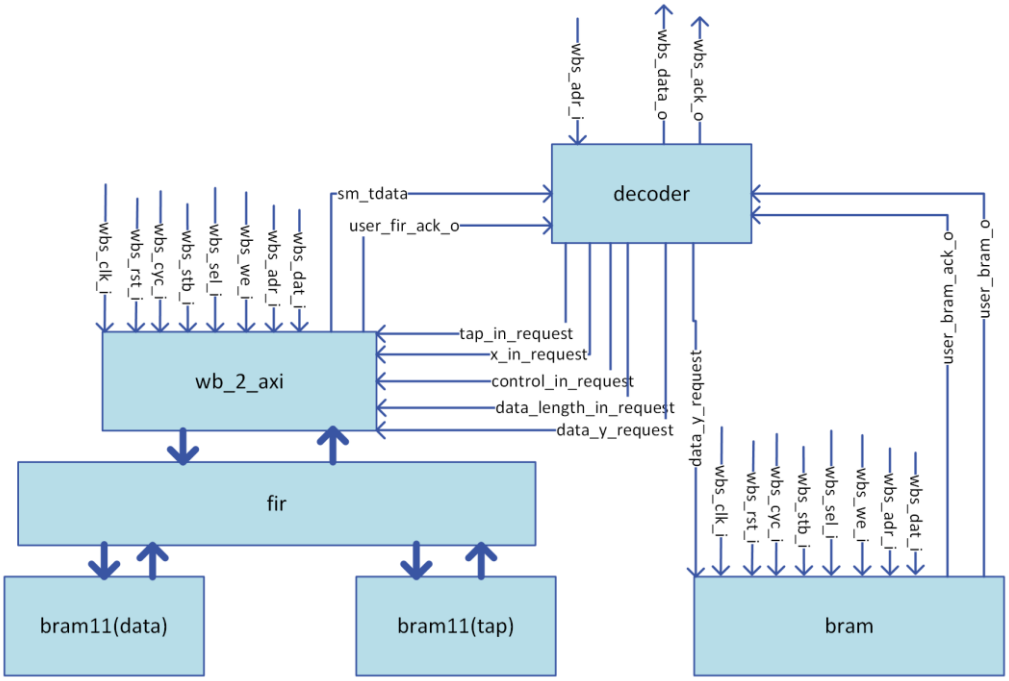
Members:

M11107410 羅善寬

M11107004 曹榮恩

M11107409 陳昱碩

1. Block Diagram:

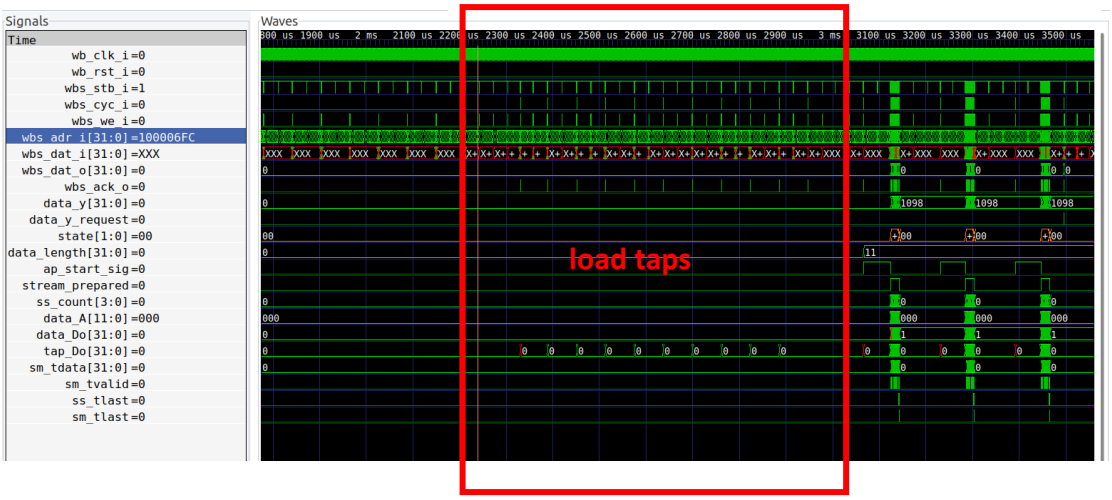


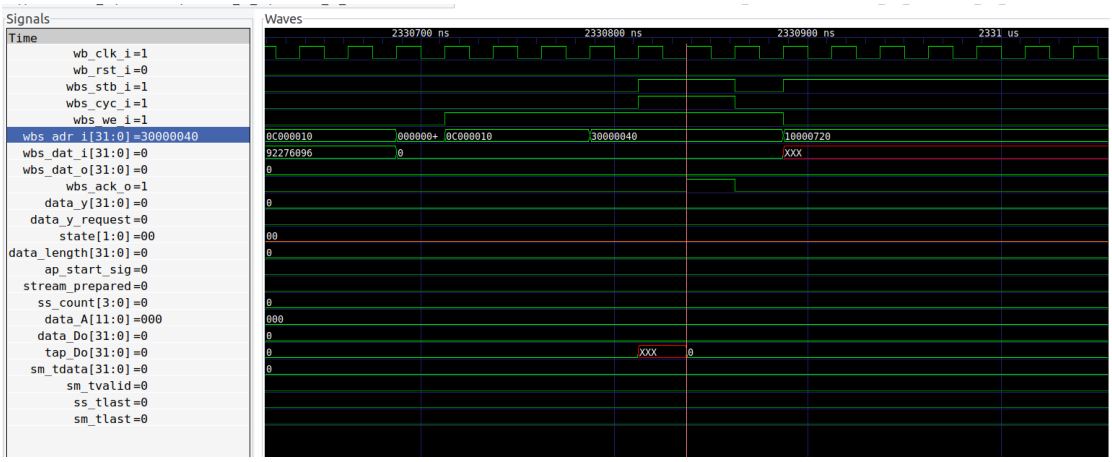
2. The interface protocol between firmware, user project and testbench

- Firmware 跟 user project 是 Wishbone
- Testbench 藉由 mprj 與 Firmware 和 user project 溝通

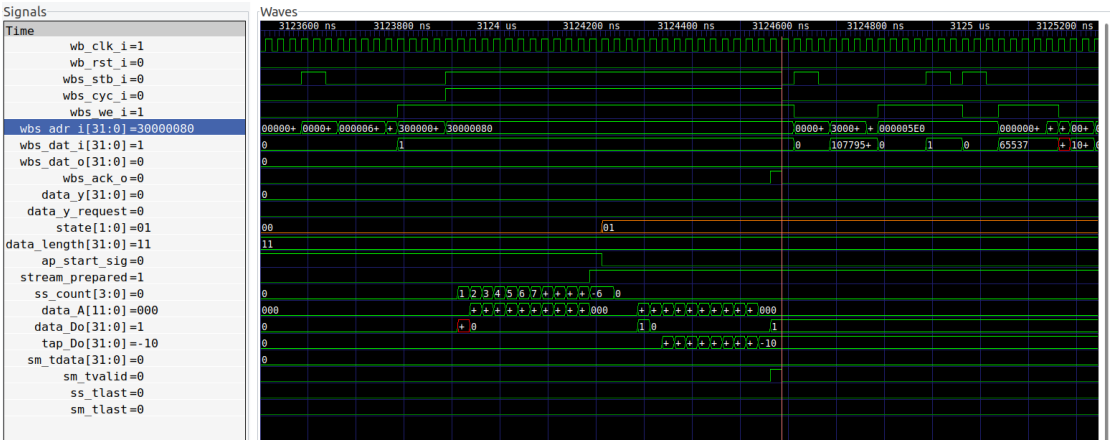
3. Waveform and analysis of the hardware/software behavior

- fir.v 前面先 load taps，tap 位置為 0x30000040

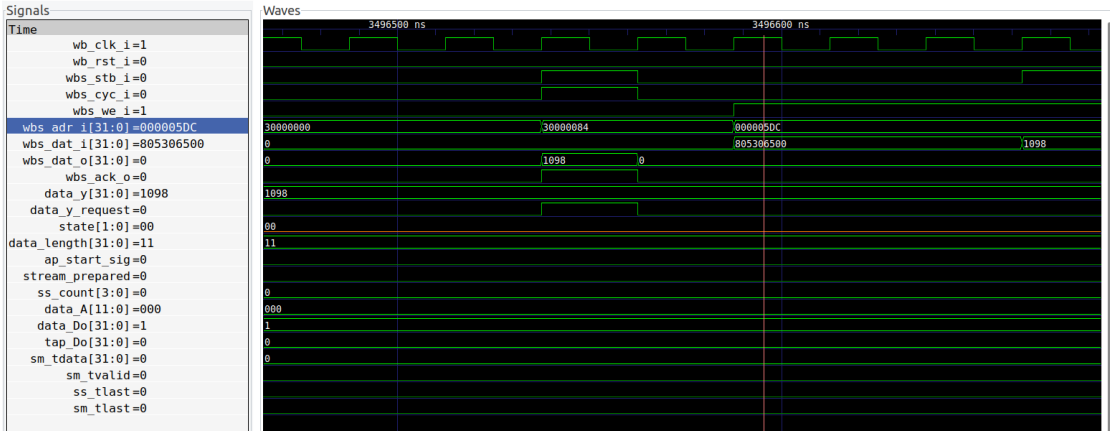




  - fir.v 執行 3 次，x 位置為 0x30000080



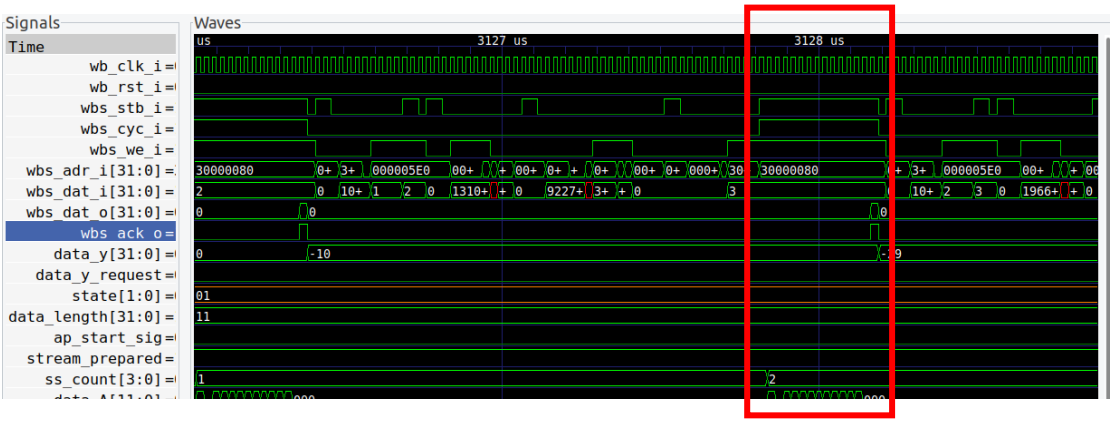
➢ fir.v 最後讀取輸出，y 位置為 0x30000084



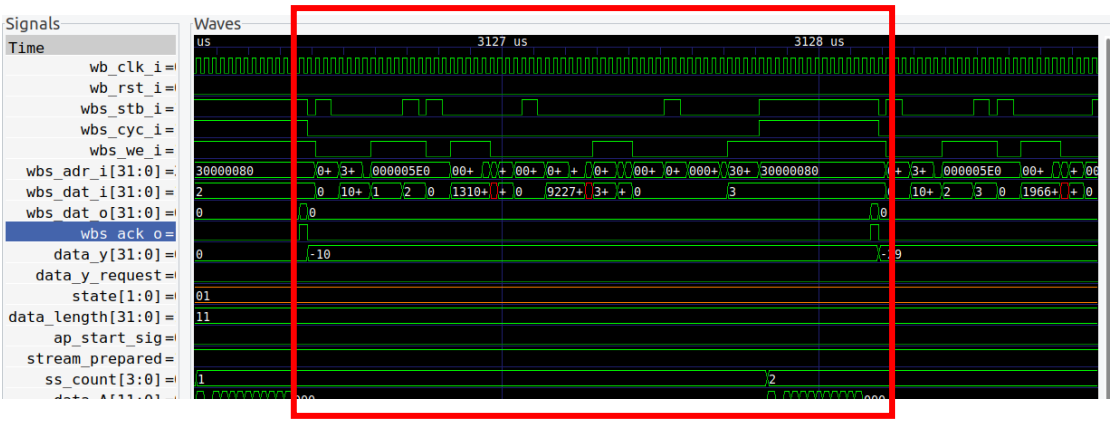
4. What is the FIR engine theoretical throughput, i.e. data rate? Actually measured throughput?

➢ 一個 clock 週期： 25ns

➢ theoretical throughput data rate： 13 clock

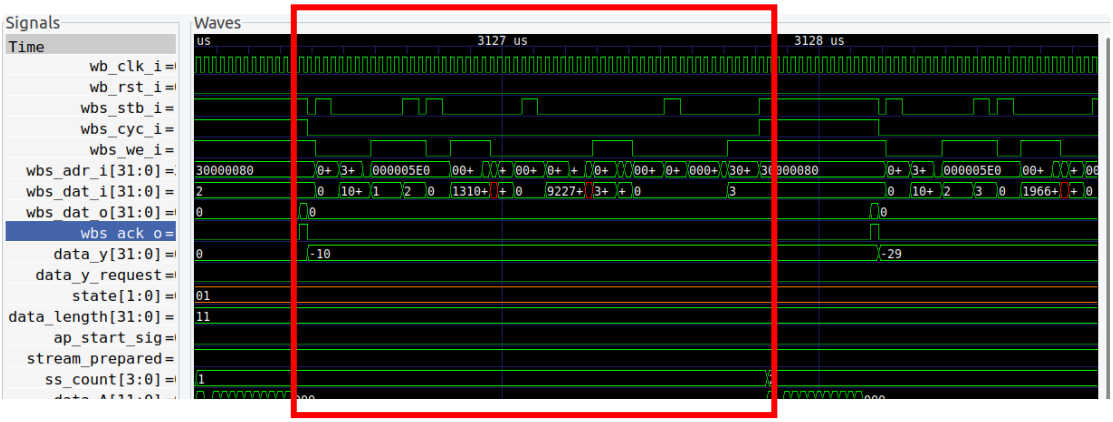


- Actually measured throughput： 72 clocks



5. What is latency for firmware to feed data?

- 59 clocks



6. What techniques used to improve the throughput?

6.1 Does bram12 give better performance, in what way?

- 對，因為 Bram 的位置多一個，所以可以多存取一筆資料，若把資料做 pre-fatch，可以加速後續的資料處理，因此可以有更好的效果。

6.2 Can you suggest other method to improve the performance?

- 增加資料輸入的頻寬。

7. Prepare firmware code & RTL:

```
module user_proj_example #(
    parameter BITS = 32,
    parameter DELAYS=10
)(
`ifdef USE_POWER_PINS
    inout vccd1,    // User area 1 1.8V supply
    inout vssd1,    // User area 1 digital ground
`endif

    // Wishbone Slave ports (WB MI A)
    input wb_clk_i,
    input wb_rst_i,
    input wbs_stb_i,
    input wbs_cyc_i,
    input wbs_we_i,
    input [3:0] wbs_sel_i,
    input [31:0] wbs_dat_i,
    input [31:0] wbs_adr_i,
    output wbs_ack_o,
    output [31:0] wbs_dat_o,

    // Logic Analyzer Signals
    input  [127:0] la_data_in,
    output [127:0] la_data_out,
    input  [127:0] la_oenb,

    // IOs
    input  [`MPRJ_IO_PADS-1:0] io_in,
    output [`MPRJ_IO_PADS-1:0] io_out,
    output [`MPRJ_IO_PADS-1:0] io_oeb,

    // IRQ
    output [2:0] irq
);
```

```
wire clk;
wire rst;
assign clk = wb_clk_i;
assign rst = wb_rst_i;

wire [1:0] decoded;
assign decoded = (wbs_adr_i[31:20] == 12'h380) ? 2'd2 :
       (wbs_adr_i[31:20] == 12'h300) ? 2'd1 : 0;

// request signal
wire user_bram_request;
assign user_bram_request = wbs_stb_i & wbs_cyc_i & (decoded == 2'd2);

wire user_fir_request;
assign user_fir_request = wbs_stb_i & wbs_cyc_i & (decoded == 2'd1);


wire user_fir_ack_o ;
assign user_fir_ack_o = (awready && wready) ? 1 :
            (control_in_request && awready) ?1 :
            (data_length_in_request && awready) ?1 :
            (data_y_request) ?1 :
            (rvalid) ? 1 :
            (sm_tvalid) ? 1 : 0;
            //(ss_tready) ? 1 : 0;
            //(sm_tvalid) ? 1 : 0;
```

```
assign wbs_ack_o = user_bram_ack_o | user_fir_ack_o ;

assign wbs_dat_o = (user_bram_ack_o)?user_bram_o:
        (data_y_request ) ? data_y:
        (user_fir_ack_o) ? sm_tdata : 0;

reg [(pDATA_WIDTH-1):0]data_y;
always@(posedge wb_clk_i)begin
    if(wb_rst_i) begin
        data_y <= 0;
    end
    else if(sm_tvalid) data_y <= sm_tdata;
end
```

```
wire tap_in_request, x_in_request, control_in_request,data_length_in_request, data_y_request;
assign tap_in_request = (user_fir_request && wbs_adr_i[19:0] >= 20'h40 && wbs_adr_i[19:0] <=20'h7f) ? 1: 0;
assign control_in_request = (user_fir_request && wbs_adr_i[19:0] == 20'h00 ) ? 1: 0;
assign data_length_in_request = (user_fir_request && wbs_adr_i[19:0] == 20'h10 ) ? 1: 0;
assign x_in_request = (user_fir_request && wbs_adr_i[19:0] >=20'h80) ? 1: 0;
assign data_y_request = (user_fir_request && wbs_adr_i[19:0] >=20'h84) ? 1: 0;


assign awvalid   = ((tap_in_request || control_in_request || data_length_in_request) & wbs_we_i) ? 1 : 0;
assign awaddr    = ((tap_in_request || control_in_request || data_length_in_request) & wbs_we_i) ? wbs_adr_i : 0;

assign wvalid    = ((tap_in_request || control_in_request || data_length_in_request)  & wbs_we_i)? 1 : 0;
assign wdata     = ((tap_in_request || control_in_request || data_length_in_request)  & wbs_we_i) ? wbs_dat_i : 0;

assign rready    = (tap_in_request & !wbs_we_i) ? 1 : 0;
assign arvalid   = (tap_in_request & !wbs_we_i) ? 1 : 0;
assign araddr    = (tap_in_request & !wbs_we_i) ? wbs_adr_i : 0;


assign ss_tlast  = (data_length_count==(data_length-1)) ? 1 : 0;

assign ss_tvalid = (x_in_request & wbs_we_i) ? 1 : 0;
assign ss_tdata  = (x_in_request & wbs_we_i) ? wbs_dat_i : 0;
```

```
reg [(pDATA_WIDTH-1):0]data_length,data_length_count;
always@(posedge wb_clk_i)begin
    if(wb_rst_i) begin
        data_length <= 0;
        data_length_count <=0;
    end
    else if(data_length_in_request) data_length <= wbs_dat_i;
    else if (sm_tlast) data_length_count <= 0;
    else if(sm_tvalid==1) data_length_count <= data_length_count+1;
end



assign sm_tready = (user_fir_request) ? 1 : 0;
```

8. Compilation

8.1 Run_clean

```
rm -rf ./gdb.debug ./gdbwave.debug
rm -f *.vcd *.hex
rm -f *.s *.o *.i *.out *.map
```

8.2 Run_sim

```
rm -f counter_la_fir.hex

riscv32-unknown-elf-gcc -Wl,--no-warn-rwx-segments -g \
        --save-temps \
        -Xlinker -Map=output.map \
        -I../../firmware \
        -march=rv32i -mabi=ilp32 -D__vexriscv__ \
        -Wl,-Bstatic,-T,../../firmware/sections.lds,--strip-discarded \
        -ffreestanding -nostartfiles -o counter_la_fir.elf ../../firmware/crt0_vex.S ../../-
  firmware/isr.c fir.c counter_la_fir.c
# -nostartfiles
riscv32-unknown-elf-objcopy -O verilog counter_la_fir.elf counter_la_fir.hex
riscv32-unknown-elf-objdump -D counter_la_fir.elf > counter_la_fir.out

# to fix flash base address
sed -ie 's/@10/@00/g' counter_la_fir.hex

iverilog -Ttyp -DFUNCTIONAL -DSIM -DUNIT_DELAY=#1 \
        -f./include.rtl.list -o counter_la_fir.vvp counter_la_fir_tb.v

vvp counter_la_fir.vvp
rm -f counter_la_fir.vvp counter_la_fir.elf counter_la_fir.hexe
```

8.3 Compilation

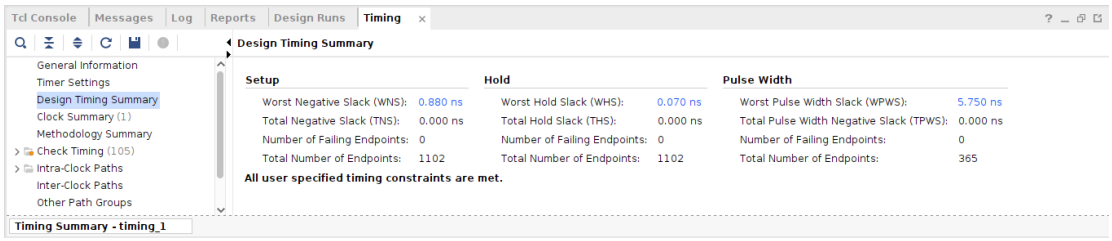
```
ubuntu@ubuntu2004:~/Desktop/lab4/Lab4-1/lab-exmem_fir/testbench/counter_la_fir$ source run_clean
ubuntu@ubuntu2004:~/Desktop/lab4/Lab4-1/lab-exmem_fir/testbench/counter_la_fir$ source run_sim
Reading counter_la_fir.hex
counter_la_fir.hex loaded into memory
Memory 5 bytes = 0x6f 0x00 0x00 0x0b 0x13
VCD info: dumpfile counter_la_fir.vcd opened for output.
LA Test 1 started
LA Test 2 passed
ubuntu@ubuntu2004:~/Desktop/lab4/Lab4-1/lab-exmem_fir/testbench/counter_la_fir$
```

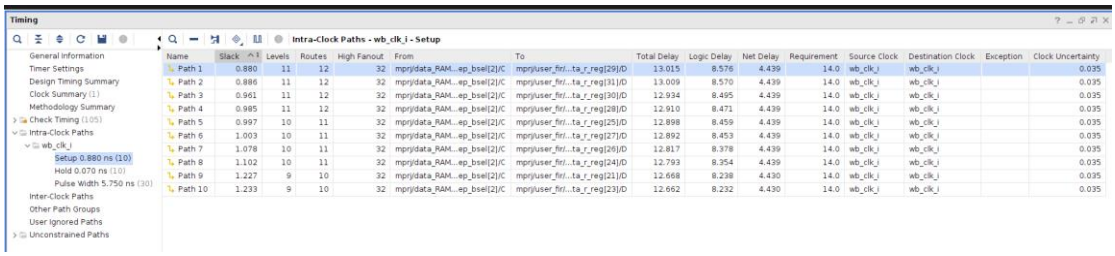
9. Synthesis & Verification

9.1 Waveform

9.2 Timing report

- Operation system：Linux
- Priod：14ns





9.3 Synthesis report

```
---------------------------------------------------------------------------------
Start RTL Component Statistics
---------------------------------------------------------------------------------
Detailed RTL Component Info :
+---Adders :
            2 Input   32 Bit       Adders := 1
            2 Input   12 Bit       Adders := 5
            2 Input    4 Bit       Adders := 3
+---Registers :
                      32 Bit     Registers := 6
                      12 Bit     Registers := 5
                       4 Bit     Registers := 4
                       1 Bit     Registers := 19
+---Multipliers :
                     32x32  Multipliers := 1
+---RAMs :
                    128K Bit   (4096 X 32 bit)          RAMs := 1
                     384 Bit   (12 X 32 bit)          RAMs := 2
+---Muxes :
            2 Input   32 Bit        Muxes := 31
            3 Input   32 Bit        Muxes := 1
            2 Input   12 Bit        Muxes := 11
            3 Input   12 Bit        Muxes := 1
            2 Input    8 Bit        Muxes := 3
            2 Input    4 Bit        Muxes := 7
            3 Input    4 Bit        Muxes := 2
            3 Input    2 Bit        Muxes := 1
            2 Input    2 Bit        Muxes := 1
            2 Input    1 Bit        Muxes := 33
            3 Input    1 Bit        Muxes := 15
            6 Input    1 Bit        Muxes := 1
---------------------------------------------------------------------------------
Finished RTL Component Statistics
---------------------------------------------------------------------------------
```

1. Slice Logic

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	547	0	0	53200	1.03
LUT as Logic	419	0	0	53200	0.79
LUT as Memory	128	0	0	17400	0.74
LUT as Distributed RAM	128	0			
LUT as Shift Register	0	0			
Slice Registers	265	0	0	106400	0.25
Register as Flip Flop	265	0	0	106400	0.25
Register as Latch	0	0	0	106400	0.00
F7 Muxes	0	0	0	26600	0.00
F8 Muxes	0	0	0	13300	0.00

9.4 Metrics :

#-of-clock (latency-timer) * clock_period * gate-resource

Metrics = 72 * 13.120 ns * (547+265) = 767,047.68