7.2.3 Mô tả thiết kế bằng Verilog

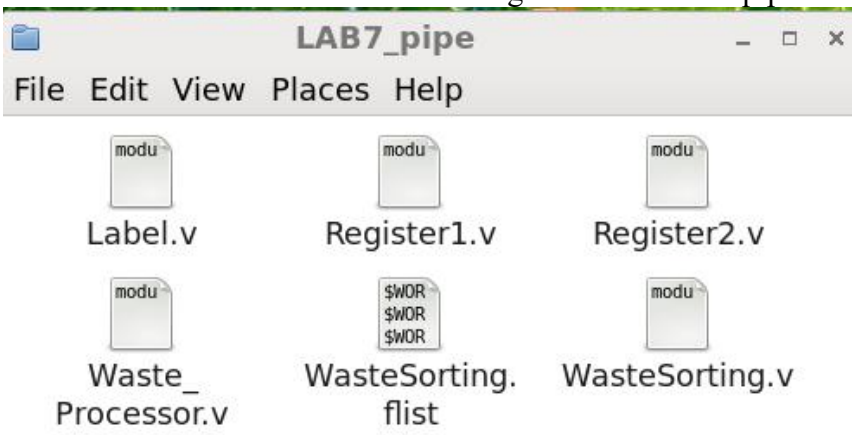
Đầu vào	Đầu ra	Các công đoạn
Specification	Các file mã lập trình đuôi .v, trong đó mô tả các khối trong thiết kế cùng kết nối giữa chúng thông qua ngôn ngữ Verilog	- Dùng Verilog mô tả lần lượt các khối trong thiết kế - Kết nối các khối con và instance chúng trong các khối lớn hơn - Kết nối các khối lớn hơn cho đến khi đến cấp độ cao nhất

Bảng 7.2 Mô tả bước thiết kế cấp độ RTL.

Ở mục này, ta sẽ thực hiện thiết kế 1 file Verilog:

- **WasteSorting.v** : Khối xử lí lớn.

Hoặc ta có thể tách thiết kế ra nhiều file Verilog để mô tả rõ tính pipeline hơn :



- 

7.2.3.1. Mô tả thiết kế bằng Verilog.

nội dung cũng như giải thích ngắn gọn các file trên được thể hiện trong hình 7.3. Các file này sẽ được thêm đường dẫn vào file lab_rtl.flist như bài thực hành-thí nghiệm trước (ta sẽ thực hiện thiết kế 1 file verilog trong trường hợp này)

WasteSorting.v

```
module WasteSorting (
    input wire clk,
    input wire rst,
    input wire [7:0] in_waste,

    output reg [7:0] plastic_counter,
    output reg [7:0] glass_counter,
    output reg [7:0] paper_counter,
    output reg [7:0] metal_counter,
    output reg [7:0] textile_counter,
    output reg [7:0] compost_counter,
    output reg [7:0] landfill_counter,
    output reg [7:0] counter,

    output reg [7:0] out_plastic,
    output reg [7:0] out_glass,
    output reg [7:0] out_paper,
    output reg [7:0] out_metal,
    output reg [7:0] out_textile,
    output reg [7:0] out_compost,
    output reg [7:0] out_landfill,
    output reg [7:0] weight
);


reg [7:0] weight_weight;

always @(posedge clk ) begin
    if (rst) begin

    out_plastic         <= 8'd0;
    out_glass           <= 8'd0;
    out_paper           <= 8'd0;
    out_metal           <= 8'd0;
    out_textile         <= 8'd0;
    out_compost         <= 8'd0;
    out_landfill        <= 8'd0;
    weight              <= 8'd0;

    plastic_counter     <= 8'd0;
    glass_counter       <= 8'd0;
    paper_counter       <= 8'd0;
    metal_counter       <= 8'd0;
    textile_counter     <= 8'd0;
    compost_counter     <= 8'd0;
    landfill_counter    <= 8'd0;
    counter             <= 8'd0;
```

```
    end else begin
        weight_weight = {4'b0, in_waste [3:0]};
        counter = counter+ 8'd1;                                    // đếm số lần của in_waste
        weight = weight + weight_weight;                            // tổng size
        case (in_waste[7:6])
            2'b00: begin
                case (in_waste[5:4])
                    2'b00: begin
                        out_plastic = out_plastic + weight_weight;      //size từng cái
                        plastic_counter = plastic_counter + 8'd1;       // thứ tự từng cái

                    end
                    2'b01: begin
                        out_glass = out_glass + weight_weight;
                        glass_counter = glass_counter + 8'd1;

                    end
                    2'b10: begin
                        out_paper = out_paper + weight_weight;
                        paper_counter = paper_counter + 8'd1;

                    end
                    2'b11: begin
                        out_metal = out_metal + weight_weight;
                        metal_counter = metal_counter + 8'd1;
                    end
                endcase
            end
            2'b01:
                case (in_waste[5:4])
                2'b00: begin
                        out_textile = out_textile + weight_weight;
                        textile_counter = textile_counter + 8'd1;

                        end
                default: begin
                out_compost = out_compost + weight_weight;
                compost_counter = compost_counter + 8'd1;

            end
                endcase
            default: begin
                        out_landfill = out_landfill + weight_weight;
                        landfill_counter = landfill_counter + 8'd1;

                    end

        endcase
    end
id

idmodule
```

Hình 7.3 Code Verilog của RTL bài lab 7

7.2.3.2. Mô tả thiết kế bằng Verilog nêu rõ tính Pipeline

Label.v

```
module Label (
    input wire clk,
    input wire rst,
    input wire endi,
    input [3:0] label,
    input [7:0] weight,
    output reg [3:0] address,
    output reg [7:0] out1,
output reg q
);

always @(posedge clk or posedge rst) begin
    if (rst) begin
        address <= 4'd0;
        out1 <= 8'd0;
    end else if (endi) begin
        case (label)
            4'b0000: address <= 4'd0;  // plastic
            4'b0001: address <= 4'd1;  // glass
            4'b0010: address <= 4'd2;  // paper
            4'b0011: address <= 4'd3;  // metal
            4'b0100: address <= 4'd4;  // textile
            4'b0110,
            4'b0111: address <= 4'd5;  // compost
            4'b1000,
            4'b1001,
            4'b1111,
            4'b1100: address <= 4'd6;  // landfill
            default: address <= 4'd7;  // enr
        endcase
        out1 <= weight;
        q<=endi;
    end else begin
        address <= address; // giữ nguyên giá trị
        out1 <= out1;       // giữ nguyên giá trị
    end
end

endmodule
```

Hình 7.4 file label.v

Register1.v

```
module Register1 (
    input wire clk,
    input wire rst,
    input wire [7:0] d,
    output reg [7:0] q,
    output reg q1
);

always @(posedge clk or posedge rst) begin
    if (rst) begin
        q <= 8'b0;
        q1 <= 1'b0;
    end else begin
        q <= d;
        q1 <= 1'b1;
    end
end

endmodule
```

Hình 7.5 file Register1.v

Register2.v

```
module Register2 (
    input wire clk,
    input wire rst,
    input wire endi,
    input wire [3:0] d,
    input wire [7:0] d1,
    output reg [3:0] q,
    output reg [7:0] q1,
    output reg q2
);

always @(posedge clk or posedge rst) begin
    if (rst) begin
        q <= 4'b0;
        q1 <= 8'b0;
    end else if (endi) begin
        q <= d;
        q1 <= d1;
        q2 <=endi;
    end
end

endmodule
```

Hình 7.6 file Register2.v

Waste_Processor.v

```
module Waste_Processor (
    input wire clk,
    input wire rst,
input wire endi,
    input [7:0] in_waste,
    output wire [3:0] label,
    output wire [7:0] weight,
   output wire q
);

assign label = endi ? in_waste[7:4] : 4'b0000;
assign weight = endi ? {4'b0000, in_waste[3:0]} : 8'b00000000;
assign q=endi;


endmodule
```

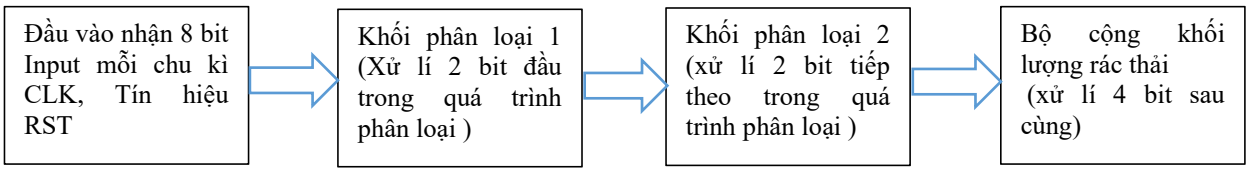
Hình 7.7 file Waster_Processor.v

WasteSorting.v

```
module WasteSorting (
    input wire clk,
    input wire rst,
    input wire [7:0] in_waste,
    output reg [7:0] out_plastic,
    output reg [7:0] out_glass,
    output reg [7:0] out_paper,
    output reg [7:0] out_metal,
    output reg [7:0] out_textile,
    output reg [7:0] out_compost,
    output reg [7:0] out_landfill,
    output reg [7:0] out_enr,
    output reg [7:0] weight

);
    reg [7:0] mem_weight [7:0];

    wire endi;
    wire [7:0] reg_in_waste1;
Register1 reg1 (
        .clk(clk),
        .rst(rst),
        .d(in_waste),
        .q(reg_in_waste1),
        .q1(endi)
);
    wire [3:0] label;
    wire [7:0] weight1;
    wire endi1;
Waste_Processor process(
```

Hình 7.8 file WasteSorting.v

7.2.3.3 Phân tích tính Pipeline trong thiết kế



Thời gian giữa các qui trình sẽ được setup dựa trên thời gian lớn nhất có thể xảy ra của toàn bộ khắp các khối xử lí. (VD : trong phân tích STA ta có thể thấy để xử lí đầu vào 00000000 sau đó 11111111 cần thời gian x, thời gian x này là lớn nhất, ta sẽ setup thời gian delay giữa mỗi khối là x để đảm bảo không có sự chồng lấn thông tin xảy ra.) Sự delay này có thể thực hiện được bằng hàm **delay #x** trong code verilog và systemverilog, trong design mạch thực tế, ta sẽ sử dụng Phase Lock Loop hoặc Resistor với các giá trị khác nhau để tạo ra delay mong muốn.

7.2.4 Thực hiện viết testbench

7.2.4.1. Xây dựng testbench có khả năng thu thập input từ file định dạng sẵn.

Đầu vào	Đầu ra	Các công đoạn
- Các file mã lập trình đuôi .v - Các file testbench cũng đuôi .v hay .sv (SystemVerilog)	- Các báo cáo (report) về quá trình mô phỏng - Các file dạng sóng (waveform) nhằm kiểm tra mô phỏng thiết kế	- Xây dựng môi trường cho việc kiểm tra - Kiểm tra lỗi cú pháp của RTL (các file .v) và testbench - Chạy mô phỏng và xuất ra dạng sóng - Kiểm tra các file báo cáo (report) và dạng sóng nhằm đảm bảo mô phỏng RTL sạch lỗi

Bảng 7.9: Mô tả bước kiểm tra thiết kế cấp độ RTL (Verification)

Bảng 7.4 là mô tả khái quát của bước kiểm tra thiết kế cấp độ RTL (Verification)

Vì tính chất phức tạp của đầu vào hệ thống , ta sẽ sử dụng hàm **$fwrite,$fscanf** và **$fopen, $fclose** để đọc và tạo thư mục test: **input.txt** và **weight.txt, counter.txt** từ file testtop.v trong bài thực hành-thí nghiệm trước để áp dụng cho bài thực hành-thí nghiệm này. (Sau đó ta có thể xây dựng lại thuật toán trên bằng các môi trường ngoài như python, C hay các ngôn ngữ bậc cao hơn, thậm chí ta có thể xây dựng một file riêng biệt để kiểm tra output của testbench với môi trường ngoài có khớp với nhau chưa).

nội dung của file trên được thể hiện trong hình 4.10.

Sau khi thêm đường dẫn của testtop.v vào file lab_ben.flist, ta thực hiện chạy giống hệt bài thực hành-thí nghiệm trước.

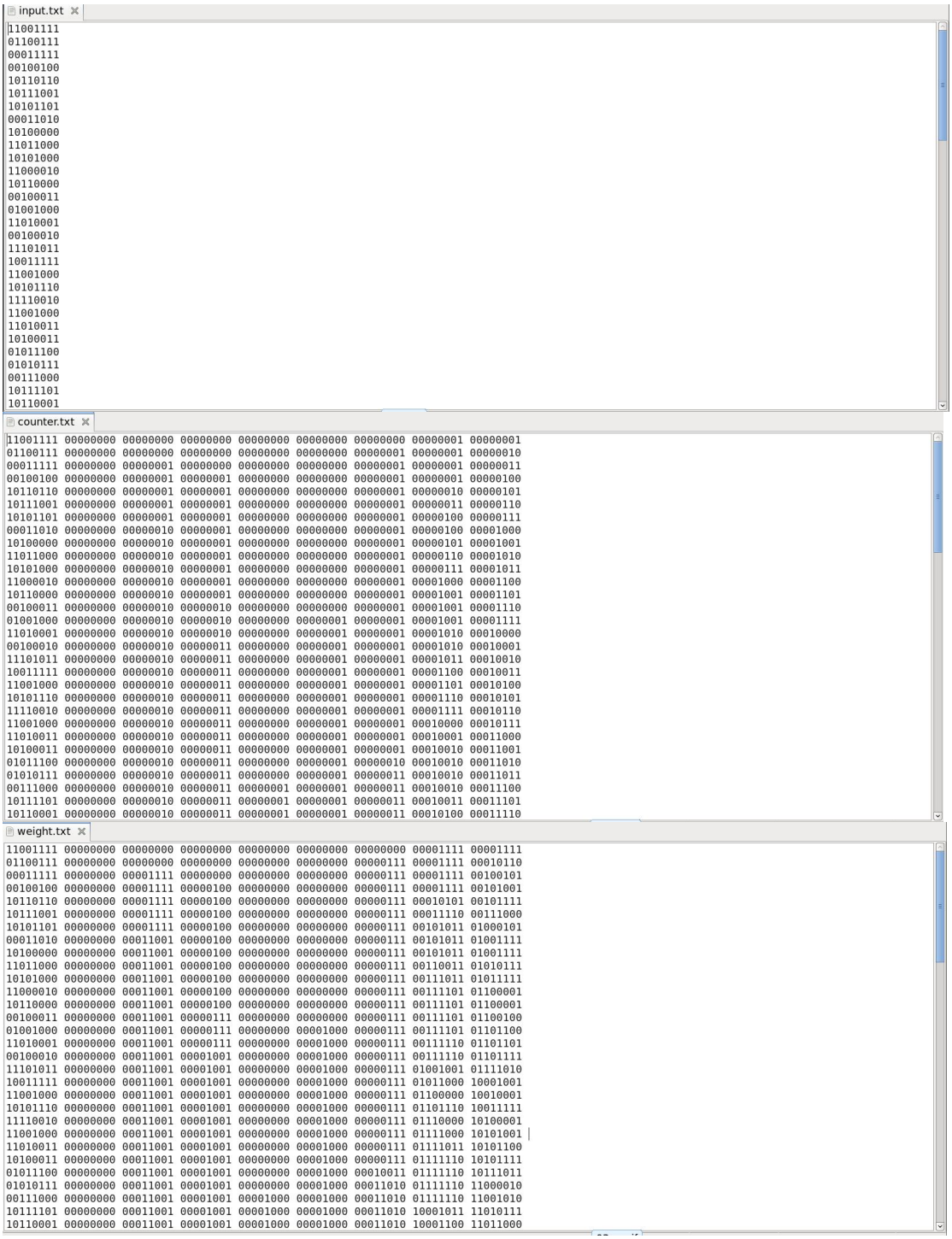
Lưu ý là cần sửa lại đường dẫn trong file set_env.bash để trỏ tới thư mục của bài thực hành-thí nghiệm hiện tại. Kết quả sau khi chạy mô phỏng được thể hiện trong hình 7.5. Tương tự, kết quả dạng sóng được thể hiện trong hình 7.6

*testtop.v

```
module testtop;

    reg clk;
    reg rst;
    reg [7:0] in_waste;
    integer in_file, out_file,out_file1;
    integer scan_status;

    wire [7:0] plastic_counter;
    wire [7:0] glass_counter;
    wire [7:0] paper_counter;
    wire [7:0] metal_counter;
    wire [7:0] textile_counter;
    wire [7:0] compost_counter;
    wire [7:0] landfill_counter;
    wire [7:0] counter;

    wire [7:0] out_plastic;
    wire [7:0] out_glass;
    wire [7:0] out_paper;
    wire [7:0] out_metal;
    wire [7:0] out_textile;
    wire [7:0] out_compost;
    wire [7:0] out_landfill;
    wire [7:0] weight;

    WasteSorting uut (
        .clk(clk),
        .rst(rst),
        .in_waste(in_waste),
        .plastic_counter(plastic_counter),
        .glass_counter(glass_counter),
        .paper_counter(paper_counter),
        .metal_counter(metal_counter),
        .textile_counter(textile_counter),
        .compost_counter(compost_counter),
        .landfill_counter(landfill_counter),
        .counter(counter),
        .out_plastic(out_plastic),
        .out_glass(out_glass),
        .out_paper(out_paper),
        .out_metal(out_metal),
        .out_textile(out_textile),
        .out_compost(out_compost),
        .out_landfill(out_landfill),
        .weight(weight)
    );

    initial begin
        clk = 0;
        forever #5 clk = ~clk;
    end
```

```
    initial begin
        #10;
        rst = 1;
        #10;
        rst = 0;

        // Open the input file
        in_file = $fopen("/home/albert/Desktop/LAB7/03_verif/verif/sv/input.txt", "r");
        if (in_file == 0) begin
            $display("Error: Could not open input file.");
            $finish;
        end

        // Open the output file
        out_file = $fopen("/home/albert/Desktop/LAB7/03_verif/verif/sv/weight.txt", "w");
        if (out_file == 0) begin
            $display("Error: Could not open output file.");
            $finish;
        end

        out_file1 = $fopen("/home/albert/Desktop/LAB7/03_verif/verif/sv/counter.txt", "w");
        if (out_file1 == 0) begin
            $display("Error: Could not open output file.");
            $finish;
        end


       while (!$feof(in_file)) begin
           scan_status = $fscanf(in_file, "%b\n", in_waste);
           if (scan_status != 1) begin
               $display("Error: Failed to read input values.");
               $finish;
           end

           #10; // wait for a few clock cycles for the output to settle

           $fwrite(out_file, "%b %b %b %b %b %b %b %b %b \n",in_waste, out_plastic, out_glass, out_paper, out_metal, out_textile,
out_compost, out_landfill, weight);
           $fwrite(out_file1, "%b %b %b %b %b %b %b %b %b %b \n",in_waste, plastic_counter, glass_counter, paper_counter, metal_counter,
textile_counter, compost_counter, landfill_counter, counter);

       end


       // Close files
       $fclose(in_file);
       $fclose(out_file);
       $fclose(out_file1);
       $display("Test completed. Outputs written to output.txt");
       $finish;
   end

endmodule
```

Hình 7.10 Code Verilog của testbench bài lab 7



Hình 7.11 file input.txt và counter.txt và weight.txt

7.2.4.2. Xây dựng môi trường test bằng ngôn ngữ logic để đảm bảo testbench hoạt động như đã định.

- Chọn python làm môi trường test viết (lưu ý sử dụng python3.9)
- Tạo folder tên DFT
- Tạo 3 file 3 pyhton trong folder DFT với tên "**check.py**", "**testcase.py**" , "**in_wastev1.py**"

Với nội dung :

- File **in_wastev1.py** là code python dùng để lấy data in_waste từ file input ở địa chỉ **/home/albert/Desktop/LAB7/input/input.txt** để xử lý và xuất ra file output ở **/home/albert/Desktop/LAB7/output/weightpy.txt** và **/home/albert/Desktop/LAB7/output/counterpy.txt**

```
def waste_sorting(input_file, counter_file, weight_file):
    # Initialize counters and outputs
    plastic_counter = 0
    glass_counter = 0
    paper_counter = 0
    metal_counter = 0
    textile_counter = 0
    compost_counter = 0
    landfill_counter = 0
    counter = 0

    out_plastic = 0
    out_glass = 0
    out_paper = 0
    out_metal = 0
    out_textile = 0
    out_compost = 0
    out_landfill = 0
    weight = 0

    # Read input file and process data
    with open(input_file, 'r') as infile, open(counter_file, 'w') as counter_outfile, open(weight_file, 'w') as weight_outfile:
        for line in infile:
            in_waste = int(line.strip(), 2)
            weight_weight = in_waste & 0x0F  # Extract the 4 least significant bits as weight
            counter += 1
            weight += weight_weight

            # Determine waste type and update counters
            waste_type = (in_waste >> 6) & 0x03
            sub_type = (in_waste >> 4) & 0x03

            if waste_type == 0b00:
                if sub_type == 0b00:
                    out_plastic += weight_weight
                    plastic_counter += 1
                elif sub_type == 0b01:
                    out_glass += weight_weight
                    glass_counter += 1
                elif sub_type == 0b10:
                    out_paper += weight_weight
                    paper_counter += 1
                elif sub_type == 0b11:
                    out_metal += weight_weight
                    metal_counter += 1
            elif waste_type == 0b01:
                if sub_type == 0b00:
                    out_textile += weight_weight
                    textile_counter += 1
                else:
                    out_compost += weight_weight
                    compost_counter += 1
            else:
                out_landfill += weight_weight
                landfill_counter += 1

            # Write current results to output files
            weight_outfile.write(f"{in_waste:08b} {out_plastic:08b} {out_glass:08b} {out_paper:08b} {out_metal:08b} {out_textile:08b}
{out_compost:08b} {out_landfill:08b} {weight:08b}\n")
            counter_outfile.write(f"{in_waste:08b} {plastic_counter:08b} {glass_counter:08b} {paper_counter:08b} {metal_counter:08b}
{textile_counter:08b} {compost_counter:08b} {landfill_counter:08b} {counter:08b}\n")

# Call the function with updated file paths
waste_sorting("/home/albert/Desktop/LAB7/input/input.txt",
              "/home/albert/Desktop/LAB7/output/counterpy.txt",
              "/home/albert/Desktop/LAB7/output/weightpy.txt")
```

Hình 7.12. File in_wastev1.py

- File **testcase.py** là file tạo ra test case bằng cách random 2 giá trị 1 và 0 và đưa data đến file **/home/albert/Desktop/LAB7/input/input.txt**

```
import random

def generate_test_cases(output_file, num_cases=20):
    with open(output_file, 'w') as f:
        for _ in range(num_cases):
            # Sinh giá trị ngẫu nhiên từ 0 đến 255 và chuyển thành chuỗi nhị phân 8-bit
            in_waste = random.randint(0, 255)
            in_waste_bin = format(in_waste, '08b')
            f.write(f"{in_waste_bin}\n")


# Gọi hàm để sinh các test cases và lưu vào file
generate_test_cases("/home/albert/Desktop/LAB7/input/input.txt")
```

Hình 7.13. File testcase.py

File **check.py** dùng để check xem 2 file output của 2 môi trường khác nhau có giống nhau không bằng cách so sánh từng dòng của 2 file output của 2 code

```
                    else:
                        diff.append(f"{RED}{char1}{RESET}" if char1 else f"{RED}<{char2}>{RESET}")

                diff_line = ''.join(diff)
                results.append(f"Line {i + 1}: fail\nFile1: {line1}\nFile2: {line2}\nDiff: {diff_line}")

        # Ghi kết quả vào file result.txt
        with open(result_file, 'w') as f:
            for result in results:
                f.write(result + '\n')

def main():
    # Đặt tên các tệp cần so sánh
    file_pairs = [
        ("/home/albert/Desktop/LAB7/output/counter.txt", "/home/albert/Desktop/LAB7/output/counterpy.txt", "/home/albert/Desktop/LAB7/output/
result_counter.txt"),
        ("/home/albert/Desktop/LAB7/output/weight.txt", "/home/albert/Desktop/LAB7/output/weightpy.txt", "/home/albert/Desktop/LAB7/output/
result_weight.txt")
    ]

    for file1, file2, result_file in file_pairs:
        compare_files_line_by_line(file1, file2, result_file)

if __name__ == "__main__":
    main()

def compare_files_line_by_line(file1, file2, result_file):
    RED = '\033[91m'  # Mã màu đỏ cho terminal
    RESET = '\033[0m'  # Mã reset màu

    with open(file1, 'r') as f1, open(file2, 'r') as f2:
        lines1 = f1.readlines()
        lines2 = f2.readlines()

        max_lines = max(len(lines1), len(lines2))
        results = []

        for i in range(max_lines):
            # Lấy dòng từ mỗi tệp, nếu không có dòng thì gán là rỗng
            line1 = lines1[i].strip() if i < len(lines1) else ""
            line2 = lines2[i].strip() if i < len(lines2) else ""

            if line1 == line2:
                results.append(f"Line {i + 1}: pass")
            else:
                # Tạo màu đỏ cho các điểm khác biệt
                diff = []
                max_length = max(len(line1), len(line2))
                for j in range(max_length):
                    char1 = line1[j] if j < len(line1) else ''
                    char2 = line2[j] if j < len(line2) else ''
                    if char1 == char2:
                        diff.append(char1)
```

Hình 7.14. File check.py

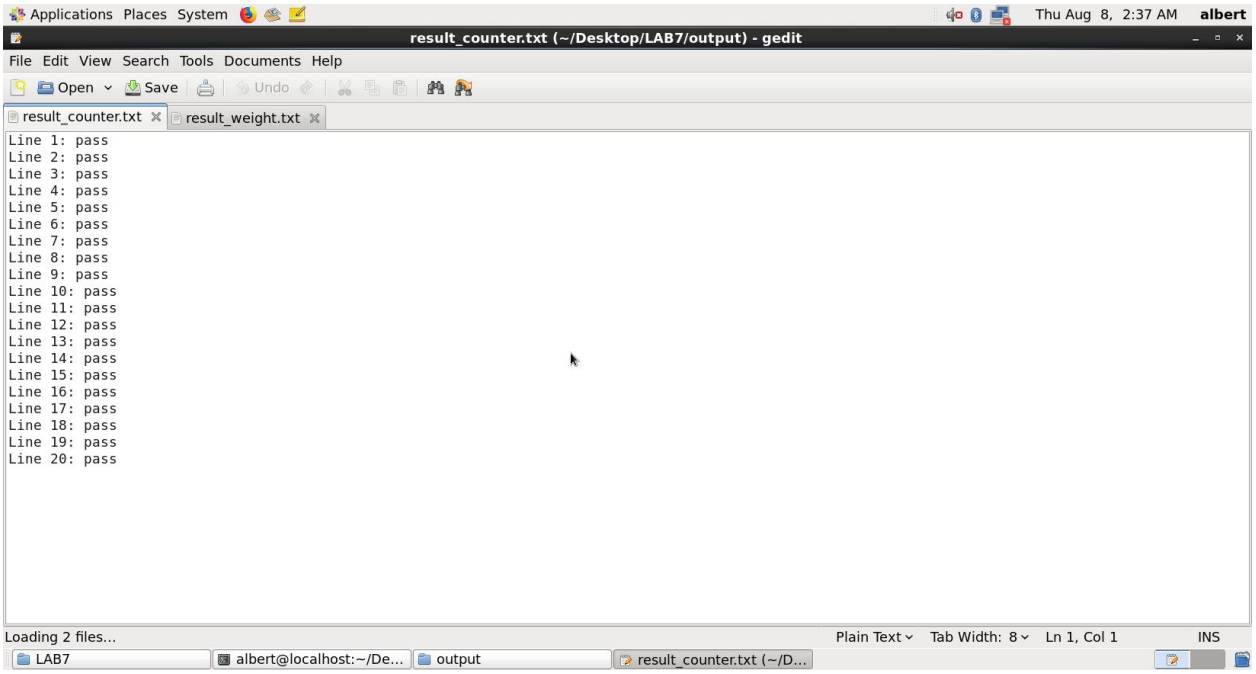
Tạo file run_all.sh với nội dung

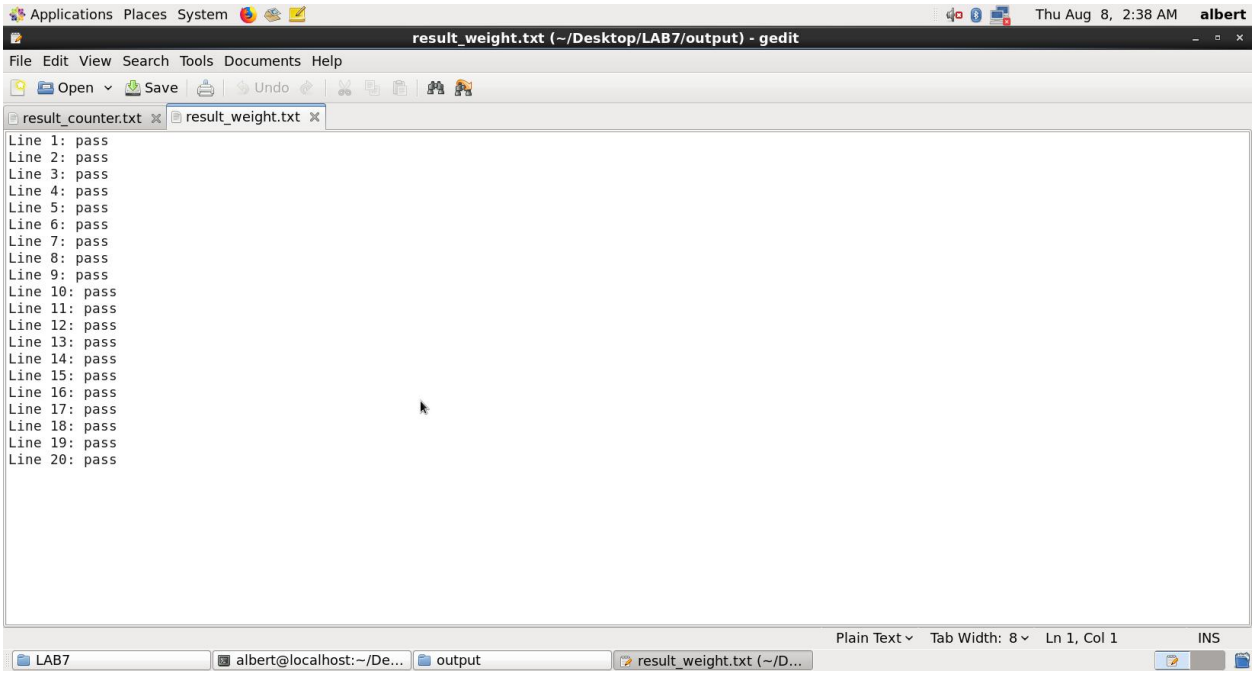
run_all.sh

```
#!/bin/bash

python3.6 /home/albert/Desktop/LAB7/DFT/testcase.py
cd /home/albert/Desktop/LAB7/03_verif/
source env_scripts/set_env.bash
cd verif/scripts/
make
cd /home/albert/Desktop/LAB7/DFT
python3.6 in_wastev1.py
python3.6 check.py
```

Hình 7.15 file run_all.sh

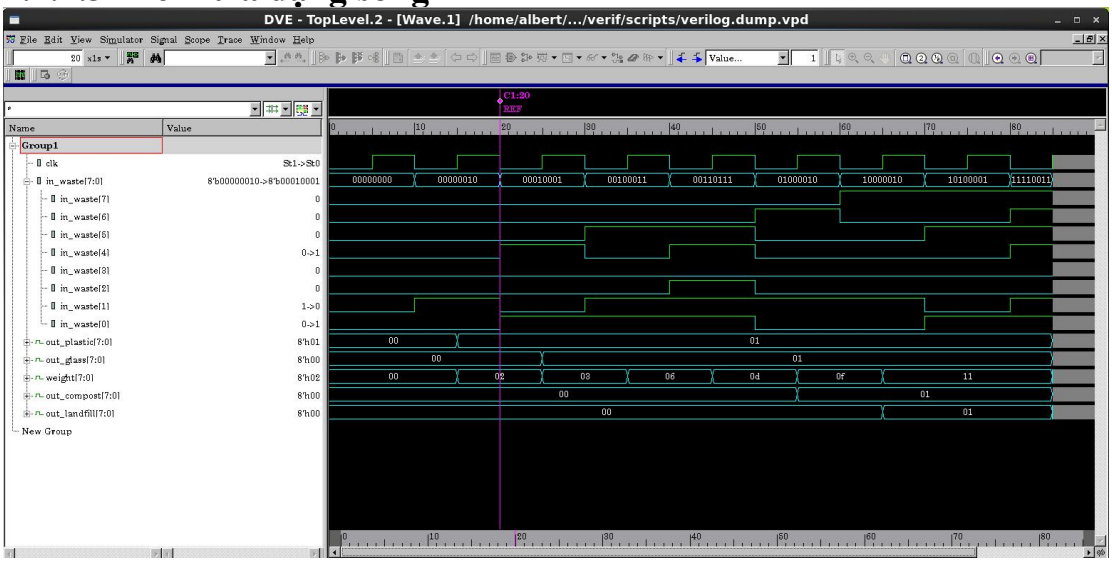
Trước tiên tạo input bằng cách chạy file testcase.py . Sau đó thực hiện các thao tác giống như các chương trước , thực hiện xong sẽ chạy file in_wastev1.py để tạo ra output sau đó chạy file check.py kiểm tra xem kết quả có đúng không





Hình 7.16 file input và counter.txt và weight.txt được thực hiện ở môi trường ngoài.

7.2.4.3 Kiểm tra dạng sóng



Hình 7.17 mô phỏng dạng sóng khi t setup ngỏ vào giới hạn

7.2.5 Tổng hợp (Synthesis)

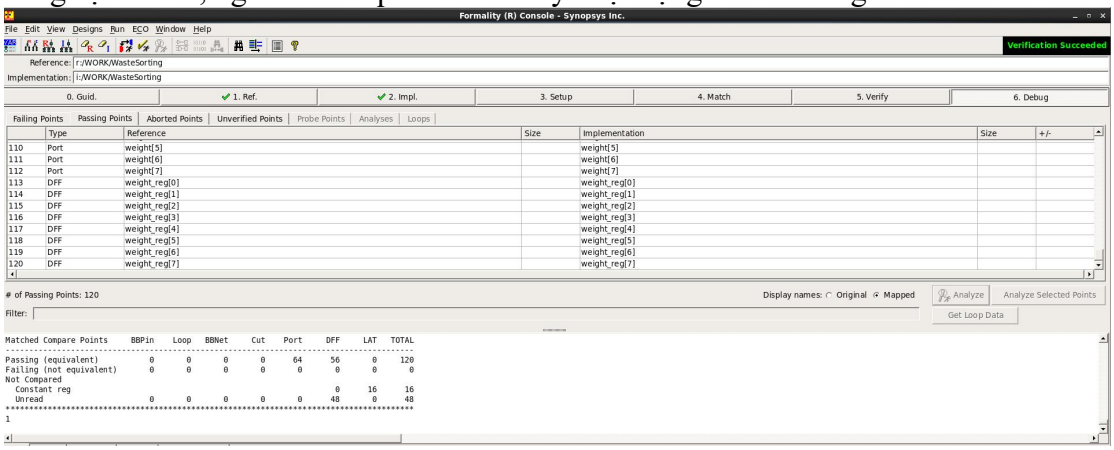
Trong phần tổng hợp synthesis này ta sẽ thêm một số ràng buộc thời gian đơn giản trong hình 7.8

```
#============= ANALYSE DESIGN ==================
#analyze -format verilog -vcs "-f ./../../03_verif/verif/scripts/rtl.f"
analyze -format verilog "./../../02_rtl/WasteSorting.v"
elaborate WasteSorting
current_design WasteSorting

#============= CONSTRAINT FOR DESIGN ============
create_clock -name clk -period 40 {clk}
set_input_delay -max 10 -clock clk  [all_inputs]
set_input_delay -min 1 -clock clk  [all_inputs]
set_output_delay -max 10 -clock clk [all_outputs]
set_output_delay -min 1 -clock clk [all_outputs]
set_fanout_load 8 [all_outputs]
```

Hình 7.18 : phần thay đổi trong file dc_command.src

7.2.6 Kiểm tra netlist

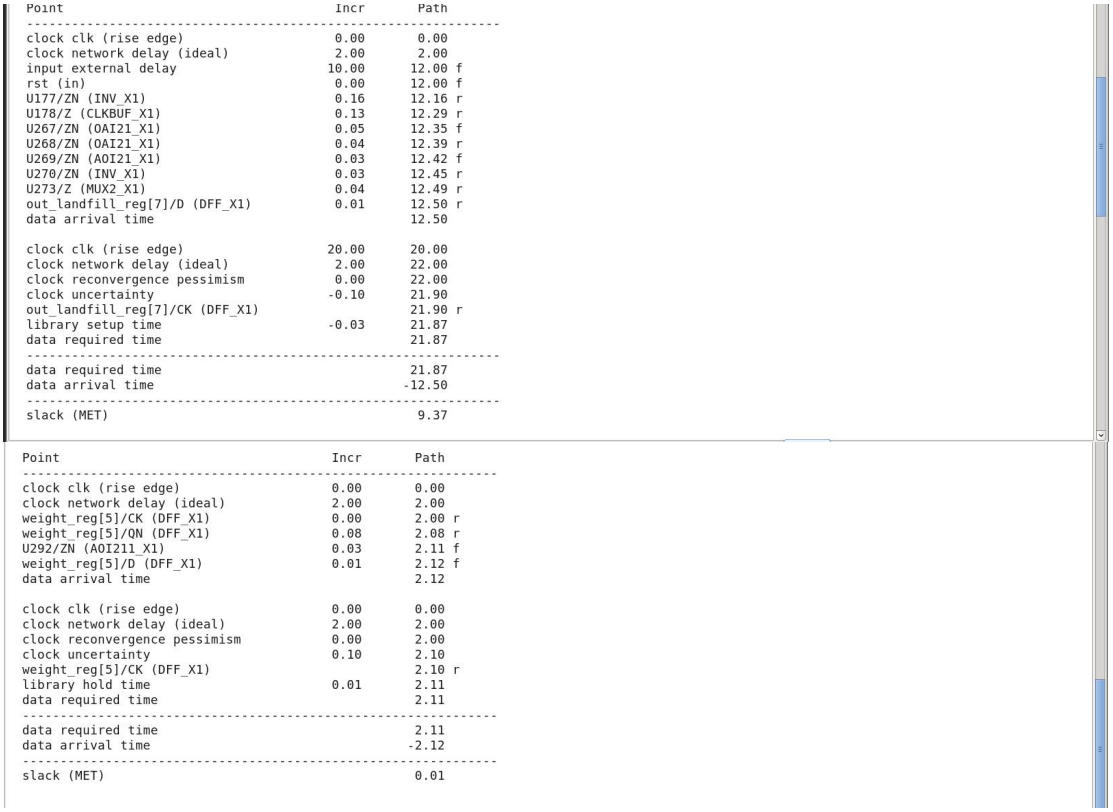
Do trong quá trình thiết kế bằng verilog, ta đã thêm vào các bit 0 trôi trong quá trình thực hiện cộng biến 4 bit cho biến 8 bit, nên tool formality sẽ cảnh báo : 4 bit 0 đang bị thả trôi, ngoài ra thì quá trình verify hoạt động bình thường.



Hình 7.19 verification successful

7.2.7 Phân tích thời gian tĩnh (STA)

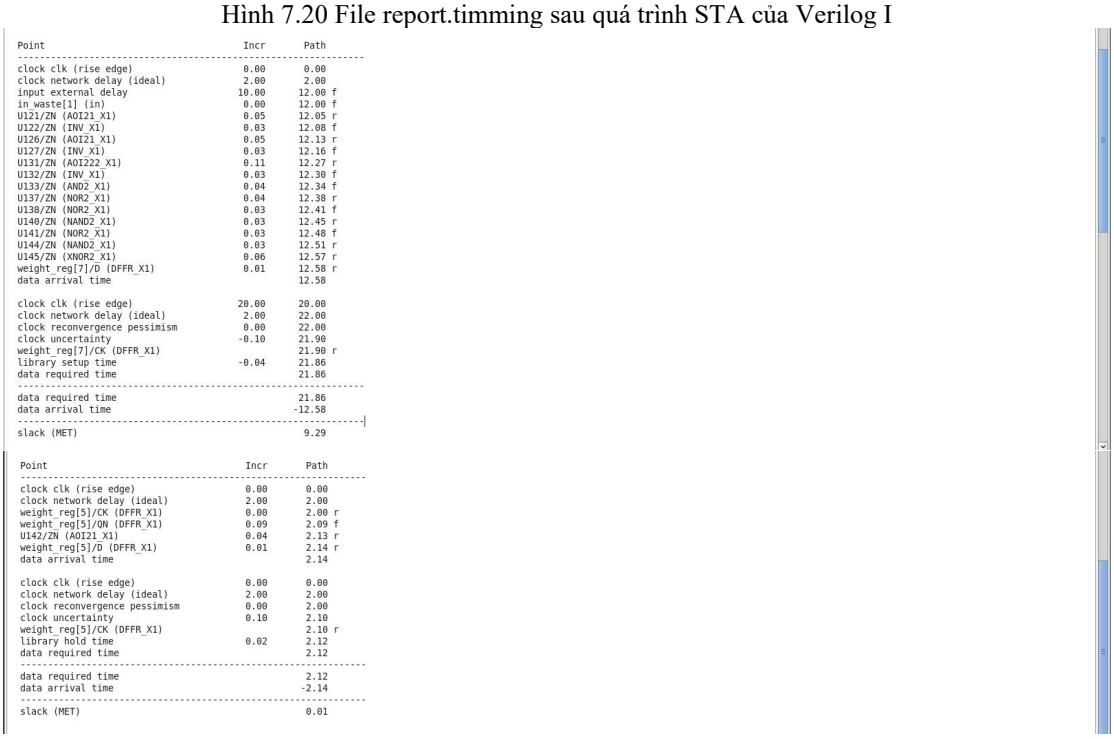
Đây là quá trình quan trọng trong thiết kế các model sử dụng thuật toán pipeline, ta không chỉ cần kiểm tra các model có phù hợp với constraint mà hệ thống yêu cầu, mà còn phải đáp ứng các specifications của khách hàng, nhà sản xuất.



Point	Incr	Path
clock clk (rise edge)	0.00	0.00
clock network delay (ideal)	2.00	2.00
input external delay	10.00	12.00 f
rst (in)	0.00	12.00 f
U177/ZN (INV_X1)	0.16	12.16 r
U178/Z (CLKBUF_X1)	0.13	12.29 r
U267/ZN (OAI21_X1)	0.05	12.35 f
U268/ZN (OAI21_X1)	0.04	12.39 r
U269/ZN (AOI21_X1)	0.03	12.42 f
U270/ZN (INV_X1)	0.03	12.45 r
U273/Z (MUX2_X1)	0.04	12.49 r
out_landfill_reg[7]/D (DFF_X1)	0.01	12.50 r
data arrival time		12.50
clock clk (rise edge)	20.00	20.00
clock network delay (ideal)	2.00	22.00
clock reconvergence pessimism	0.00	22.00
clock uncertainty	-0.10	21.90
out_landfill_reg[7]/CK (DFF_X1)		21.90 r
library setup time	-0.03	21.87
data required time		21.87
data required time		21.87
data arrival time		-12.50
slack (MET)		9.37

Point	Incr	Path
clock clk (rise edge)	0.00	0.00
clock network delay (ideal)	2.00	2.00
weight_reg[5]/CK (DFF_X1)	0.00	2.00 r
weight_reg[5]/QN (DFF_X1)	0.08	2.08 r
U292/ZN (AOI211_X1)	0.03	2.11 f
weight_reg[5]/D (DFF_X1)	0.01	2.12 f
data arrival time		2.12
clock clk (rise edge)	0.00	0.00
clock network delay (ideal)	2.00	2.00
clock reconvergence pessimism	0.00	2.00
clock uncertainty	0.10	2.10
weight_reg[5]/CK (DFF_X1)		2.10 r
library hold time	0.01	2.11
data required time		2.11
data required time		2.11
data arrival time		-2.12
slack (MET)		0.01

Hình 7.20 File report.timming sau quá trình STA của Verilog I



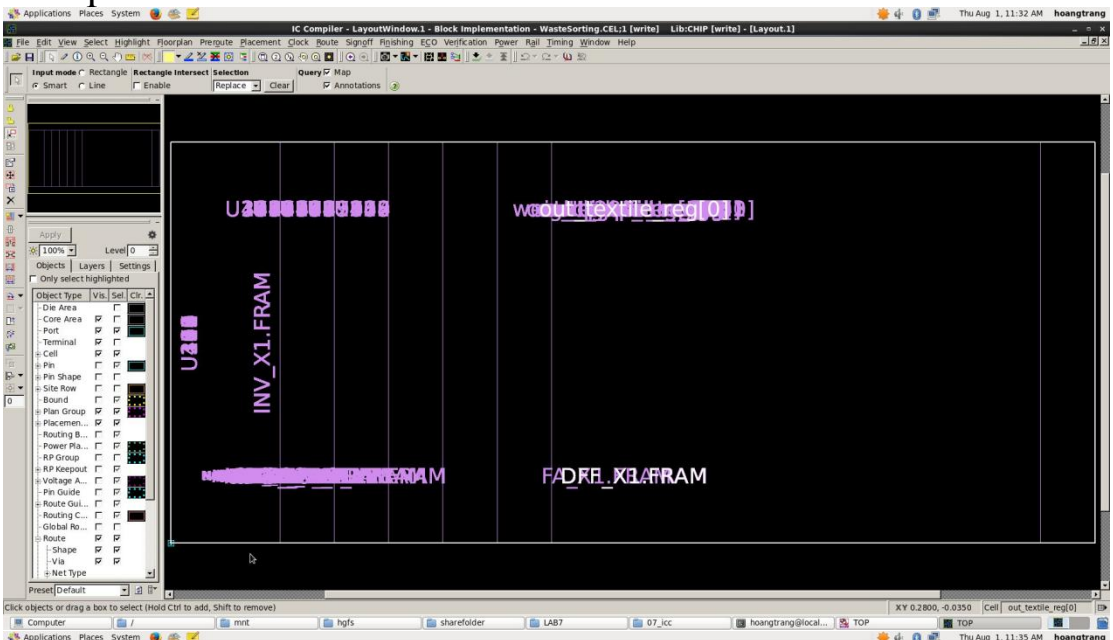
Point	Incr	Path
clock clk (rise edge)	0.00	0.00
clock network delay (ideal)	2.00	2.00
input external delay	10.00	12.00 f
in_waste[1] (in)	0.00	12.00 f
U121/ZN (AOI21_X1)	0.05	12.05 r
U122/ZN (INV_X1)	0.03	12.08 f
U126/ZN (AOI21_X1)	0.05	12.13 r
U127/ZN (INV_X1)	0.03	12.16 f
U131/ZN (AOI222_X1)	0.11	12.27 r
U132/ZN (INV_X1)	0.03	12.30 f
U133/ZN (AND2_X1)	0.04	12.34 f
U137/ZN (NOR2_X1)	0.04	12.38 r
U138/ZN (NOR2_X1)	0.03	12.41 f
U140/ZN (NAND2_X1)	0.03	12.45 r
U141/ZN (NOR2_X1)	0.03	12.48 f
U144/ZN (NAND2_X1)	0.03	12.51 r
U145/ZN (XNOR2_X1)	0.06	12.57 r
weight_reg[7]/D (DFFR_X1)	0.01	12.58 r
data arrival time		12.58
clock clk (rise edge)	20.00	20.00
clock network delay (ideal)	2.00	22.00
clock reconvergence pessimism	0.00	22.00
clock uncertainty	-0.10	21.90
weight_reg[7]/CK (DFFR_X1)		21.90 r
library setup time	-0.04	21.86
data required time		21.86
data required time		21.86
data arrival time		-12.58
slack (MET)		9.29

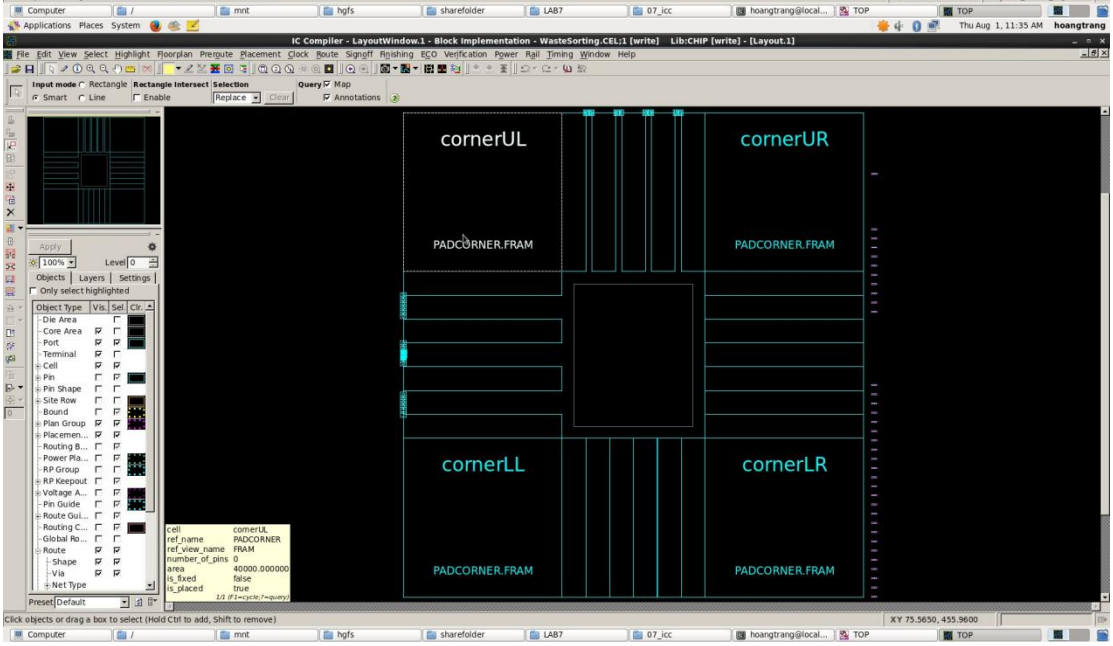
Point	Incr	Path
clock clk (rise edge)	0.00	0.00
clock network delay (ideal)	2.00	2.00
weight_reg[5]/CK (DFFR_X1)	0.00	2.00 r
weight_reg[5]/QN (DFFR_X1)	0.09	2.09 f
U142/ZN (AOI21_X1)	0.04	2.13 r
weight_reg[5]/D (DFFR_X1)	0.01	2.14 r
data arrival time		2.14
clock clk (rise edge)	0.00	0.00
clock network delay (ideal)	2.00	2.00
clock reconvergence pessimism	0.00	2.00
clock uncertainty	0.10	2.10
weight_reg[5]/CK (DFFR_X1)		2.10 r
library hold time	0.02	2.12
data required time		2.12
data required time		2.12
data arrival time		-2.14
slack (MET)		0.01

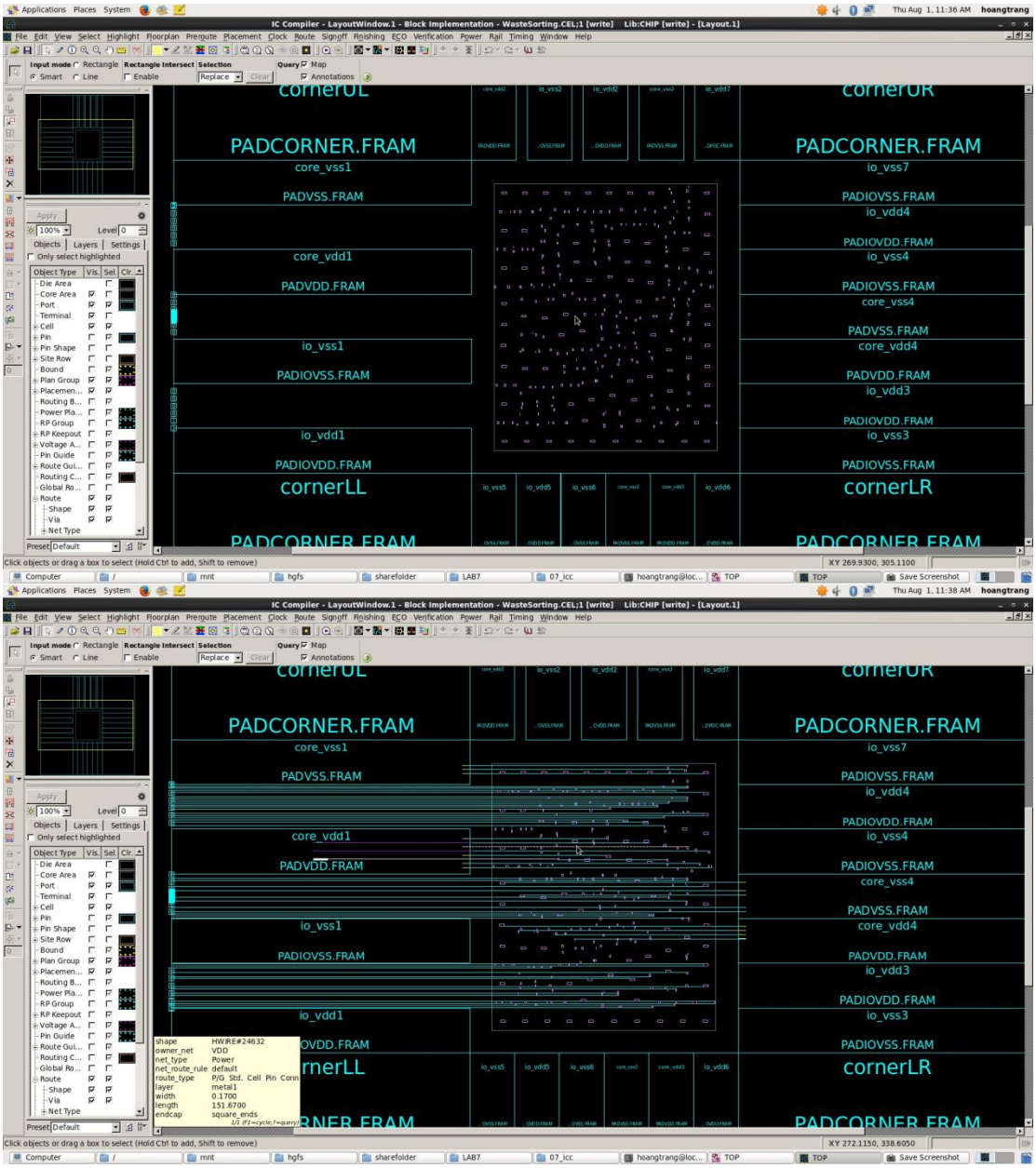
Hình 7.20 File report.timming sau quá trình STA của Verilog II

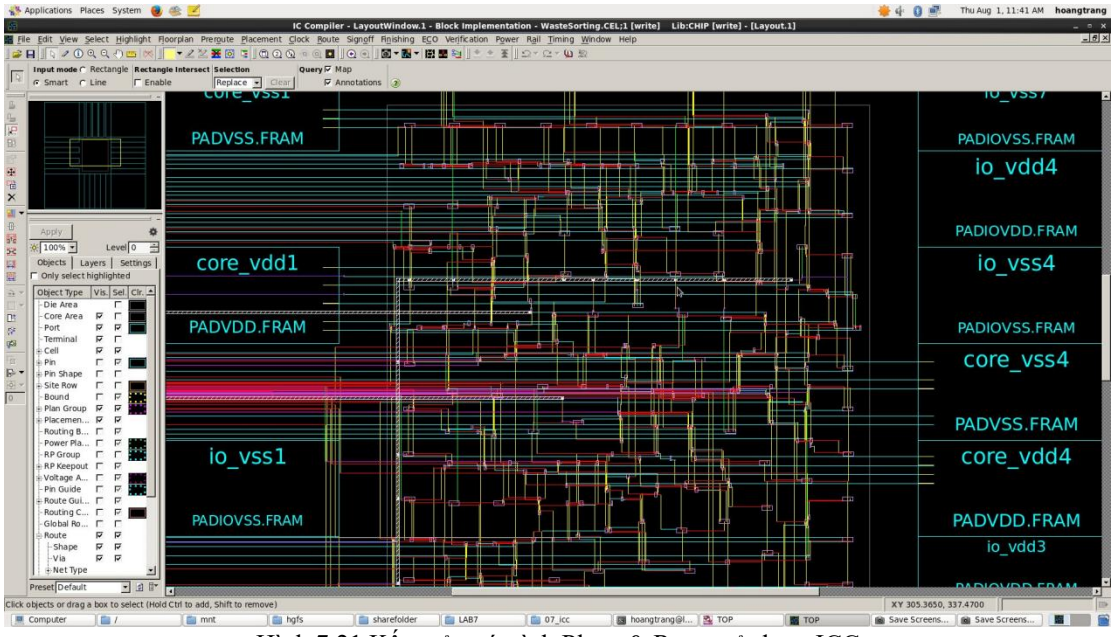
7.2.8 Place & Route

Kết quả Place & Route:






Applications Places System
Thu Aug 1, 11:36 AM hoangtrang
IC Compiler - LayoutWindow.1 - Block Implementation - WasteSorting.CEL;1 [write] Lib:CHIP [write] - [Layout.1]
File Edit View Select Highlight Floorplan Preroute Placement Clock Route Signoff Finishing ECO Verification Power Rail Timing Window Help
cornerUL
PADCORNER.FRAM
core_vss1
PADVSS.FRAM
core_vdd1
PADVDD.FRAM
io_vss1
PADIOVSS.FRAM
io_vdd1
PADIOVDD.FRAM
cornerLL
cornerUR
PADCORNER.FRAM
io_vss7
PADIOVSS.FRAM
io_vdd4
PADIOVDD.FRAM
io_vss4
PADIOVSS.FRAM
core_vss4
PADVSS.FRAM
core_vdd4
PADVDD.FRAM
io_vdd3
PADIOVDD.FRAM
io_vss3
PADIOVSS.FRAM
cornerLR
Click objects or drag a box to select (Hold Ctrl to add, Shift to remove)
XY 269.9300, 305.1100
Applications Places System
Thu Aug 1, 11:38 AM hoangtrang
IC Compiler - LayoutWindow.1 - Block Implementation - WasteSorting.CEL;1 [write] Lib:CHIP [write] - [Layout.1]
shape HWIRE#24632
owner_net VDD
net_type Power
net_route_rule default
route_type P/G Std. Cell Pin Conn
layer metal1
width 0.1700
length 151.6700
endcap square_ends
Click objects or drag a box to select (Hold Ctrl to add, Shift to remove)
XY 272.1150, 338.6050



Hình 7.21 Kết quả quá trình Place & Route sử dụng ICC