

**Laporan Tugas Besar Mata Kuliah Aplikasi Sistem Digital**  
**Implementasi Penjual Mesin Otomatis dengan Verilog**

Dosen Pengampu : Fairuz Azmi, S.T., M.T.



**Disusun oleh :**

- 1. Sahrul Ridho Firdaus (1103223009)**
- 2. Vinsensius Jonathan Fajarai (1103220179)**
- 3. Dewo Antonioly (1103223092)**
- 4. Ferensia Fransisca Agatha (1103223036)**
- 5. Grace Agustina Hutagalung (1103223089)**

**PROGRAM STUDI S1 TEKNIK KOMPUTER**  
**FAKULTAS TEKNIK ELEKTRO**  
**TELKOM UNIVERSITY**  
**BANDUNG**  
**2024/2025**

## 1. Pendahuluan

Desain IC Digital adalah proses yang bertujuan untuk merancang sirkuit terintegrasi yang merealisasikan logika digital pada perangkat elektronik. Proses ini mencakup berbagai tahapan, mulai dari deskripsi RTL (Register Transfer Level) hingga pembuatan tata letak fisik yang siap untuk fabrikasi chip. Implementasi sistem mesin penjual otomatis dilakukan menggunakan Verilog, sebuah bahasa pemrograman yang umum digunakan untuk merancang dan mensimulasikan sistem digital.

Vending machine dirancang menggunakan Verilog dengan logika finite state machine (FSM) yang memiliki tiga state utama: s0 (coin ), s1 (coin 5), dan s2 (coin 10). Sistem ini dirancang untuk menerima input berupa coin 5 atau 10, memproses transaksi, dan memberikan barang atau kembalian sesuai kebutuhan.

Proses desain ini memanfaatkan teknologi Process Design Kit (PDK) yang menyediakan komponen dasar untuk fabrikasi rangkaian. Verifikasi desain dilakukan melalui simulasi menggunakan ModelSim, sedangkan layout fisik dirancang menggunakan perangkat seperti Cadence Virtuoso.

Hasil akhir diharapkan mampu mencapai efisiensi area yang optimal, konsumsi daya yang rendah, serta arsitektur modular yang mempermudah pengembangan lebih lanjut. Desain ini menunjukkan bagaimana Verilog dapat digunakan untuk merealisasikan sistem digital yang efisien, fungsional, dan mudah diterapkan.

## 2. Teori Dasar

**Proses desain IC digital terdiri dari beberapa langkah utama:**

1. Spesifikasi: Mendefinisikan fungsi utama dari rangkaian digital.
2. Desain RTL (Register Transfer Level): Penulisan kode HDL (Verilog) berdasarkan spesifikasi fungsi.
3. Simulasi RTL: Memverifikasi fungsi desain menggunakan simulator digital (ModelSim).
4. Sintesis: Mengonversi desain RTL menjadi bentuk gerbang logika dengan memanfaatkan library dari PDK.
5. Desain Fisik: Membuat layout fisik dari hasil sintesis menggunakan perangkat lunak khusus.
6. Verifikasi Pasca-Layout: Mengevaluasi desain untuk memastikan sesuai dengan parameter fisik, seperti performa dan efisiensi daya.
7. Tape-Out dan Fabrikasi: Hasil akhir layout yang siap diproduksi.

## Deskripsi PDK yang Dipilih:

PDK (**Process Design Kit**) **OSU035** adalah salah satu PDK open-source yang disediakan oleh **Oklahoma State University (OSU)** untuk mendesain sirkuit digital dan analog menggunakan teknologi **CMOS 0.35 $\mu$ m**. PDK ini dirancang untuk keperluan edukasi dan penelitian di bidang desain sirkuit terpadu (IC).

### Fitur Utama OSU035

#### 1. Teknologi CMOS 0.35 $\mu$ m:

- Resolusi minimum transistor adalah **0.35 mikrometer**.
- Cocok untuk desain sirkuit analog dan digital.

#### 2. Library Standar:

- OSU035 menyediakan library sel standar (**standard cells**) untuk logika digital.
- Terdapat elemen-elemen seperti inverter, NAND, NOR, D flip-flop, dan lain-lain.

#### 3. Komponen dalam OSU035

- **Transistor NMOS dan PMOS.**
- **Resistor.**
- **Kapasitor.**
- **Dioda.**

## 3. Perancangan

### • 3.1 Spesifikasi

#### a. Fungsi Utama:

Desain vending machine yang menerima input berupa uang (5 atau 10 coin), mengeluarkan produk sesuai harga, dan memberikan kembalian jika diperlukan.

#### b. Input:

- **clk**: Sinyal clock.
- **rst**: Sinyal reset untuk menginisialisasi ulang mesin.
- **in**: Masukan uang (2-bit):
  - **01** = 5 coin.
  - **10** = 10 coin.

#### c. Output:

- **out**: Sinyal keluaran (1-bit), menunjukkan apakah produk telah dikeluarkan (**1**) atau tidak (**0**).
- **change**: Sinyal kembalian (2-bit):

- **00** = Tidak ada kembalian.
- **01** = Kembalian 5 coin.
- **10** = Kembalian 10 coin.

#### d. Fitur:

- Memproses input uang.
- Mengeluarkan produk jika total mencapai 10 coin.
- Memberikan kembalian yang sesuai jika input melebihi 10 coin.
- Menggunakan Finite State Machine (FSM) untuk mengelola kondisi.

### 3.2 Dataflow

#### a. Keadaan Awal (Reset):

Pada keadaan reset (**rst** = 1), mesin vending diinisialisasi ke keadaan awal (**s0**) dengan kembalian **00**.

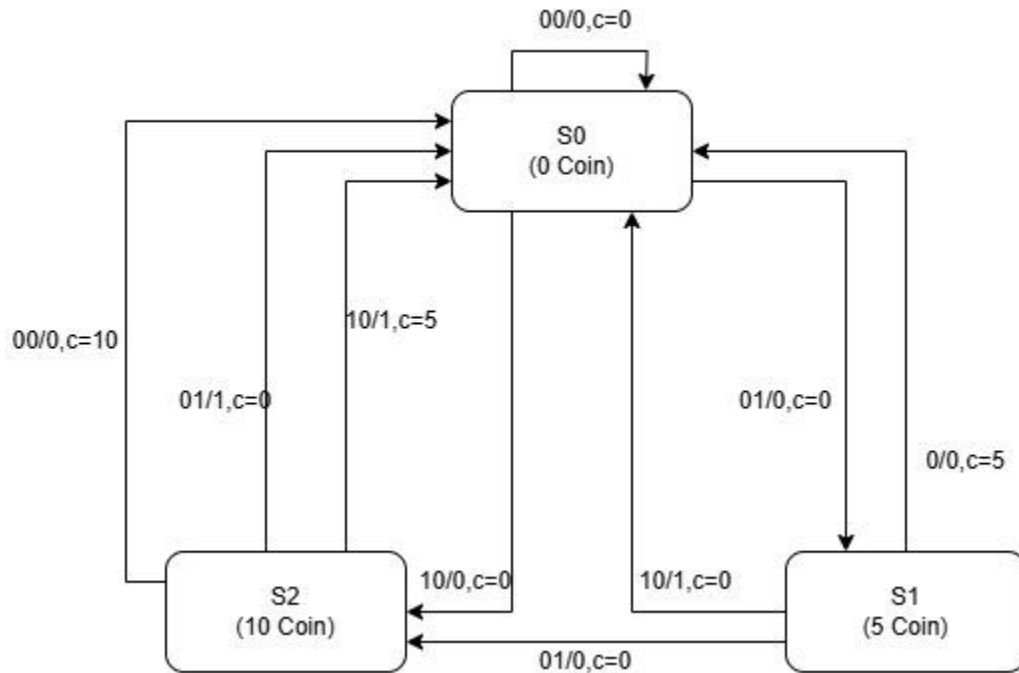
#### b. Transisi Keadaan:

- **s0 (0 coin):**
  - Jika input **01**, masuk ke s1 (5 coin).
  - Jika input **10**, masuk ke s2 (10 coin).
  - Jika input **00**, tetap di s0.
- **s1 (5 coin):**
  - Jika input **01**, masuk ke s2 (10 coin).
  - Jika input **10**, mengeluarkan produk (**out** = 1) dan kembali ke s0.
  - Jika input **00**, memberikan kembalian 5 coin dan kembali ke s0.
- **s2 (10 coin):**
  - Jika input **00**, memberikan kembalian 10 coin dan kembali ke s0.
  - Jika input **01**, mengeluarkan produk (**out** = 1) dan kembali ke s0.
  - Jika input **10**, mengeluarkan produk dan memberikan kembalian 5 coin, lalu kembali ke s0.

#### c. Output:

- Produk dikeluarkan (**out** = 1) jika total mencapai 10 coin.
- Kembalian dihitung berdasarkan kondisi input yang melebihi harga produk

### 3.3 Finite State Machines:



FSM memiliki tiga state utama:

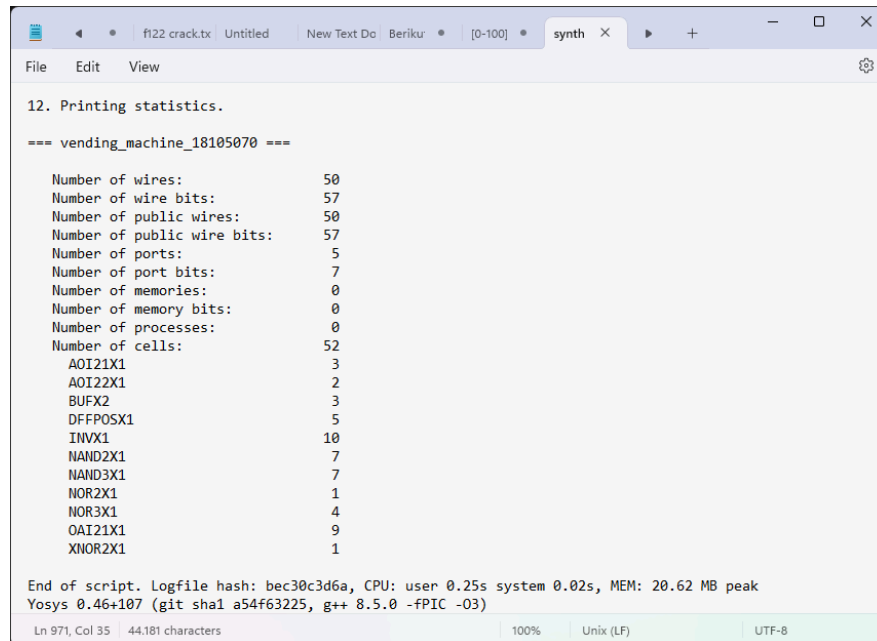
- **s0 (0 coin):**
  - Masuk ke **s1** atau **s2** berdasarkan input.
- **s1 (5 coin):**
  - Mengeluarkan produk jika input = 10 coin.
  - Memberikan kembalian jika tidak ada input tambahan.
- **s2 (10 coin):**
  - Mengeluarkan produk.
  - Memberikan kembalian bila ada input lebih dari 10 coin.

### 3.4 Hasil yang Diharapkan

Rangkaian menghasilkan output yang benar (produk atau kembalian) sesuai dengan input uang. Simulasi akan memastikan fungsi bekerja tanpa error.

## 4. Implementasi dan Pengujian

- Sintesis (List Cells) : 52 Cells yang digunakan



```
12. Printing statistics.

=== vending_machine_18105070 ===

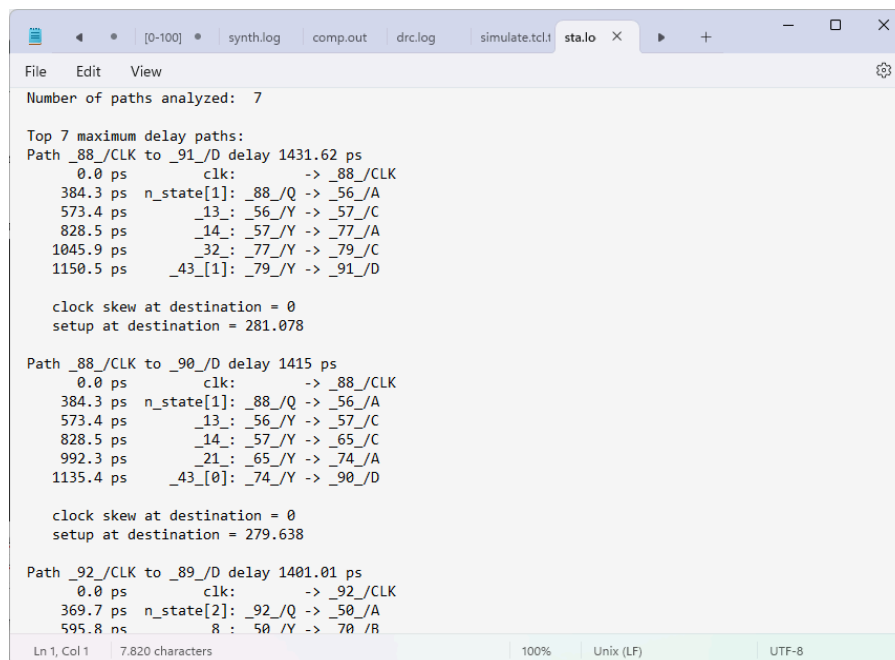
Number of wires:          50
Number of wire bits:      57
Number of public wires:   50
Number of public wire bits: 57
Number of ports:          5
Number of port bits:      7
Number of memories:       0
Number of memory bits:    0
Number of processes:      0
Number of cells:          52
  AOI21X1                  3
  AOI22X1                  2
  BUF2                     3
  DFFPOSX1                 5
  INVX1                    10
  NAND2X1                  7
  NAND3X1                  7
  NOR2X1                   1
  NOR3X1                   4
  OAI21X1                  9
  XNOR2X1                  1

End of script. Logfile hash: bec30c3d6a, CPU: user 0.25s system 0.02s, MEM: 20.62 MB peak
Yosys 0.46+107 (git sha1 a54f63225, g++ 8.5.0 -fPIC -O3)

Ln 971, Col 35 | 44.181 characters | 100% | Unix (LF) | UTF-8
```

Flip-Flop DFFPOSX1 digunakan untuk menyimpan status dalam node `n_state[1]`. Status ini penting untuk menjaga operasi sirkuit tetap sinkron dengan sinyal clock, yang diperlukan dalam banyak aplikasi seperti mesin status (FSM).

- Hasil Analisis Timing Pre-Route



```
Number of paths analyzed: 7

Top 7 maximum delay paths:
Path _88_/CLK to _91_/D delay 1431.62 ps
  0.0 ps      clk:      -> _88_/CLK
 384.3 ps    n_state[1]: _88_/Q -> _56_/A
 573.4 ps    _13_: _56_/Y -> _57_/C
 828.5 ps    _14_: _57_/Y -> _77_/A
1045.9 ps    _32_: _77_/Y -> _79_/C
1150.5 ps    _43_[1]: _79_/Y -> _91_/D

  clock skew at destination = 0
  setup at destination = 281.078

Path _88_/CLK to _90_/D delay 1415 ps
  0.0 ps      clk:      -> _88_/CLK
 384.3 ps    n_state[1]: _88_/Q -> _56_/A
 573.4 ps    _13_: _56_/Y -> _57_/C
 828.5 ps    _14_: _57_/Y -> _65_/C
 992.3 ps    _21_: _65_/Y -> _74_/A
1135.4 ps    _43_[0]: _74_/Y -> _90_/D

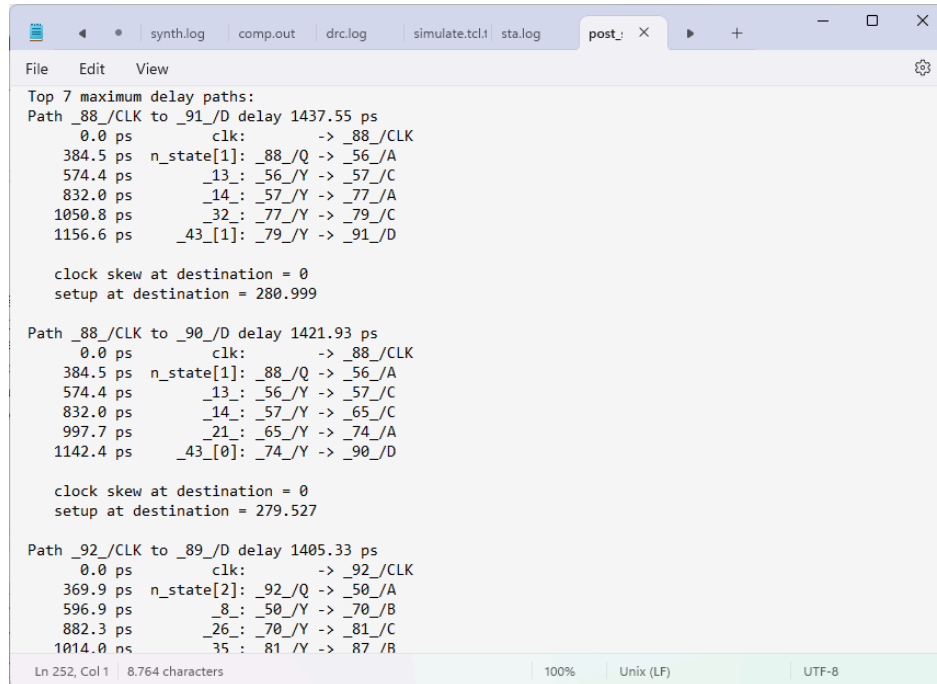
  clock skew at destination = 0
  setup at destination = 279.638

Path _92_/CLK to _89_/D delay 1401.01 ps
  0.0 ps      clk:      -> _92_/CLK
 369.7 ps    n_state[2]: _92_/Q -> _50_/A
 595.8 ps    8: _50_/Y -> _70_/B

Ln 1, Col 1 | 7.820 characters | 100% | Unix (LF) | UTF-8
```

Pre-Route ini menunjukkan hasil analisis timing statik. Analisis mengidentifikasi jalur kritis dengan delay maksimum dari 88/CLK ke 91/D sebesar 1431.62 ps, menetapkan frekuensi clock maksimum sekitar 698.509 MHz. Jalur dengan delay minimum dari 91/CLK ke output change[1] tercatat 160.329 ps. Desain telah memenuhi persyaratan hold timing minimum, memastikan stabilitas sinyal dan keandalan operasi pada frekuensi yang dirancang.

- **Hasil Analisis Timing Post-Route**



```

File Edit View
Top 7 maximum delay paths:
Path _88_/CLK to _91_/D delay 1437.55 ps
  0.0 ps clk: -> _88_/CLK
 384.5 ps n_state[1]: _88_/Q -> _56_/A
 574.4 ps _13_: _56_/Y -> _57_/C
 832.0 ps _14_: _57_/Y -> _77_/A
1050.8 ps _32_: _77_/Y -> _79_/C
1156.6 ps _43_[1]: _79_/Y -> _91_/D

  clock skew at destination = 0
  setup at destination = 280.999

Path _88_/CLK to _90_/D delay 1421.93 ps
  0.0 ps clk: -> _88_/CLK
 384.5 ps n_state[1]: _88_/Q -> _56_/A
 574.4 ps _13_: _56_/Y -> _57_/C
 832.0 ps _14_: _57_/Y -> _65_/C
 997.7 ps _21_: _65_/Y -> _74_/A
1142.4 ps _43_[0]: _74_/Y -> _90_/D

  clock skew at destination = 0
  setup at destination = 279.527

Path _92_/CLK to _89_/D delay 1405.33 ps
  0.0 ps clk: -> _92_/CLK
 369.9 ps n_state[2]: _92_/Q -> _50_/A
 596.9 ps _8_: _50_/Y -> _70_/B
 882.3 ps _26_: _70_/Y -> _81_/C
1014.0 ps _35_: _81_/Y -> _87_/B

Ln 252, Col 1 | 8.764 characters | 100% | Unix (LF) | UTF-8

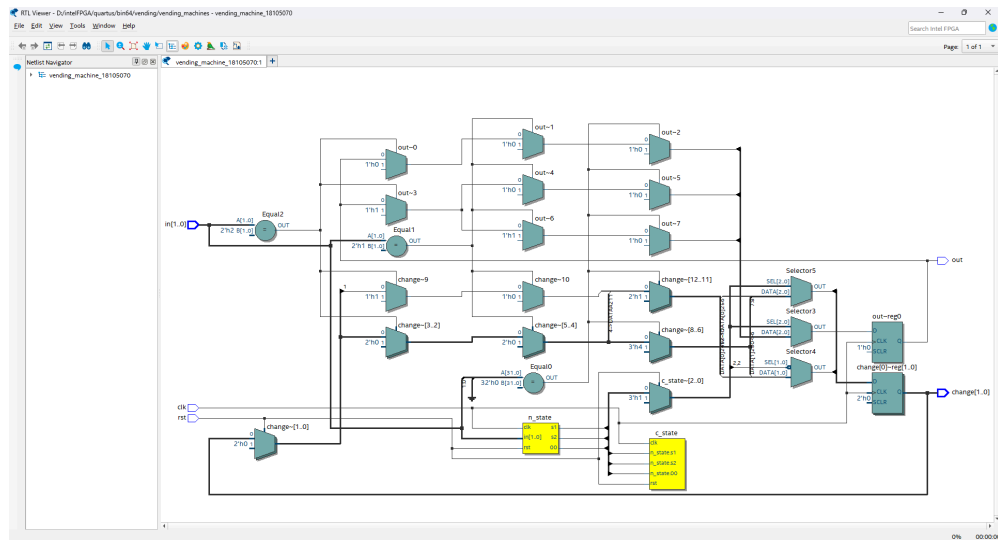
```

Log analisis timing statis menunjukkan hasil post-route untuk mesin penjual otomatis "vending\_machine\_18105070". Proses ini melibatkan konversi output dari qrouter ke berbagai format delay (Vesta, SPEF, SDF) dan analisis timing menggunakan alat Vesta.

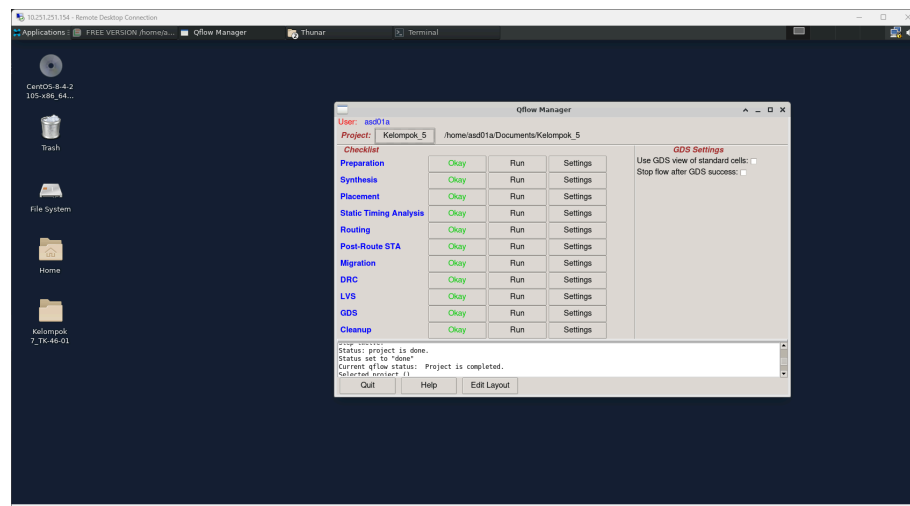
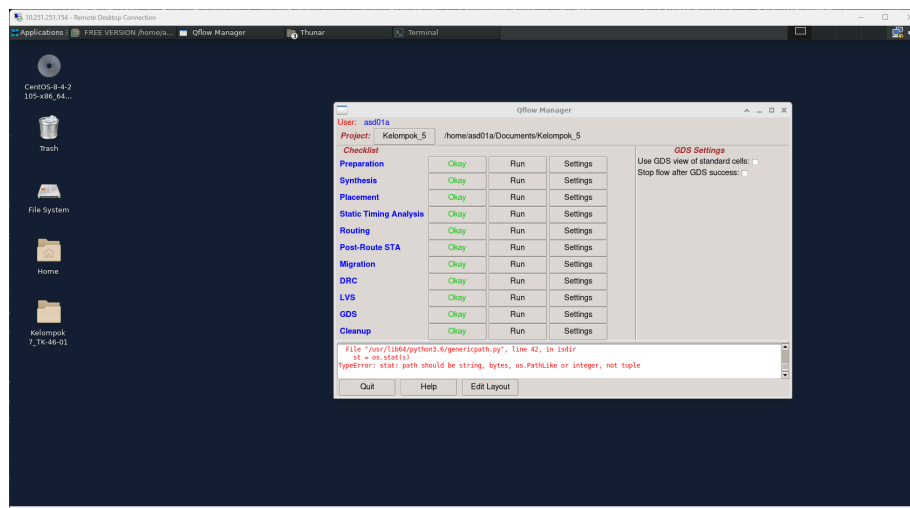
Hasilnya menunjukkan jalur dengan delay maksimum dan minimum, baik untuk sinyal dari input ke register (setup) maupun dari register ke output (hold). Jalur dengan delay maksimum memiliki delay hingga 1437,55 ps dengan frekuensi clock maksimum 695,626 MHz, sementara jalur dengan delay minimum terdeteksi 58,98 ps. Beberapa jalur mengalami ketidaksesuaian fanout pada sinyal reset.

Hasil ini dibandingkan dengan analisis pre-route menunjukkan bahwa post-route memberikan nilai delay yang lebih akurat karena memperhitungkan efek wire delay dan parasitic setelah routing.

- RTL Viewer dari desain "vending\_machine"

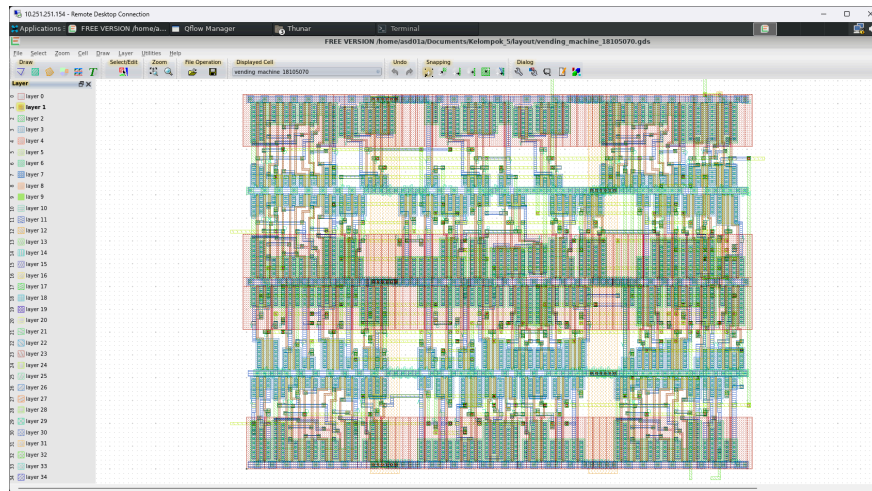


- Sintesis :





- **Layout GDS:**



- **Kode RTL dan Testbench:**

Modul desain utama

```
module vending_machine_18105070(
    input clk,
    input rst, input [1:0]in, // 01 = 5 rs, 10 = 10 rs
    output reg out,
    output reg[1:0] change
);

parameter s0 = 2'b00;
parameter s1 = 2'b01;
parameter s2 = 2'b10;

reg[1:0] c_state,n_state;
always@ (posedge clk)
begin
    if(rst == 1)
    begin
        c_state = 0;
        n_state = 0;
        change = 2'b00;
    end
    else
        c_state = n_state;

case(c_state)
```

```

s0: //state 0 : 0 rs
if(in == 0)
    begin
        n_state = s0;
        out = 0;
        change = 2'b00;
    end
else if(in == 2'b01)
    begin
        n_state = s1;
        out = 0;
        change = 2'b00;
    end
else if(in == 2'b10)
    begin
        n_state = s2;
        out = 0;
        change = 2'b00;
    end
end
s1: //state 1 : 5 rs
if(in == 0)
    begin
        n_state = s0;
        out = 0;
        change = 2'b01; //change returned 5 rs
    end
else if(in == 2'b01)
    begin
        n_state = s2;
        out = 0;
        change = 2'b00;
    end
else if(in == 2'b10)
    begin
        n_state = s0;
        out = 1;
        change = 2'b00;
    end
end
s2: //state 2 : 10 rs
if(in == 0)
    begin
        n_state = s0;
        out = 0;
        change = 2'b10;
    end
else if(in == 2'b01)

```

```

begin
    n_state = s0;
    out = 1;
    change = 2'b00;
end
else if(in == 2'b10)
begin
    n_state = s0;
    out = 1;
    change = 2'b01; //change returned 5 rs and 1 bottle
end
endcase
end
endmodule

```

#### Modul desain testbench

```

module vending_machine_tb;

//inputs
reg clk; reg[1:0] in;
reg rst;

//output
wire out;
wire[1:0] change;

vending_machine_18105070 uut(
    .clk(clk),
    .rst(rst),
    .in(in),
    .out(out),
    .change(change)
);

initial begin

    //initialise inputs
    $dumpfile("vending_machine_18105070.vcd");
    $dumpvars(0,vending_machine_tb);
    rst = 1;
    clk = 0;

```

```

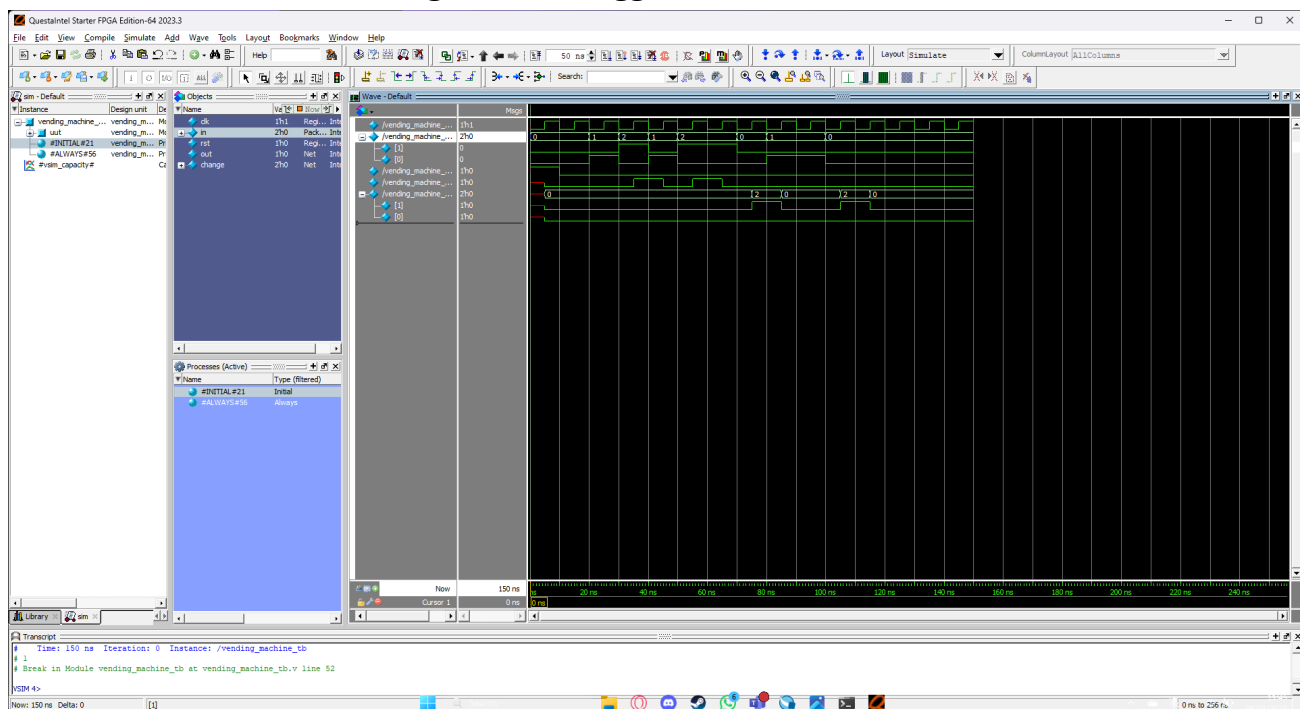
#6 rst = 0;
in = 2;
#19 in = 2;
#25 $finish;

end
always #5 clk = ~clk;

endmodule

```

- Hasil Simulasi RTL dengan 50ns menggunakan Questa



## 5. Kesimpulan

- Rangkaian vending machine berhasil diimplementasikan menggunakan PDK CMOS 0.35 $\mu$ m.
- Analisis post-layout menghasilkan frekuensi kerja maksimum 698.509 MHz dan dimensi chip sebesar 0.81mm<sup>2</sup>.
- Desain memenuhi kriteria efisiensi area, konsumsi daya, dan keandalan timing.

### Daftar Pustaka

- [1] A. Luthra, A. Jain, P. Mishra, V. Gupta, and S. Aggarwal, "Design and Implementation of Vending Machine using Verilog HDL on FPGA," *International Journal of Innovative Research in Science, Engineering and Technology*, vol. 4, no. 11, pp. 11610–11614, Nov. 2015, doi: 10.15680/IJIRSET.2015.0411137.
  
- [2] M. H. Fuad, R. Yeassin, K. M. M. Hassan, M. M. Sykot, and M. F. Nayan, "Design of a Vending Machine Using Verilog HDL and Implementation in Genus & Encounter," *European Journal of Electrical Engineering and Computer Science*, vol. 7, no. 6, pp. 88–95, Dec. 2023, doi: 10.24018/ejece.2023.7.6.595.
  
- [3] Alrehily A, Fallatah R, and Thayananthan V., "Design of vending machine using finite state machine and visual automata simulator.", *Int J Comput Appl.* Apr 2015;115(18):37–42. doi: 10.5120/20254-2623.