KEYPAD DAN KEYPAD HANDLING

Laporan Individu Tugas Besar Aplikasi Sistem Digital

Disusun Oleh:

Maulana Hafiz 1103223105

Teknik Komputer

Fakultas Teknik Elektro

Universitas Telkom

2024

1. Theory

Kemajuan teknologi telah mendorong pengembangan sistem parkir pintar untuk mengatasi berbagai kendala pada sistem parkir konvensional, seperti antrian panjang dan pengelolaan tempat parkir yang tidak efisien. Dalam sistem ini, penggunaan keypad sebagai media input menjadi elemen penting untuk memasukkan data, seperti nomor plat atau kode parkir. Keypad dan keypad handler mengimplementasikan modul logika digital berbasis Verilog untuk membaca dan memproses input keypad, yang diintegrasikan dengan FPGA. Pendekatan ini memungkinkan sistem parkir pintar bekerja secara efisien dengan respons cepat dan akurasi tinggi, mendukung pengelolaan parkir yang lebih cerdas dan ramah pengguna.Algorithm.

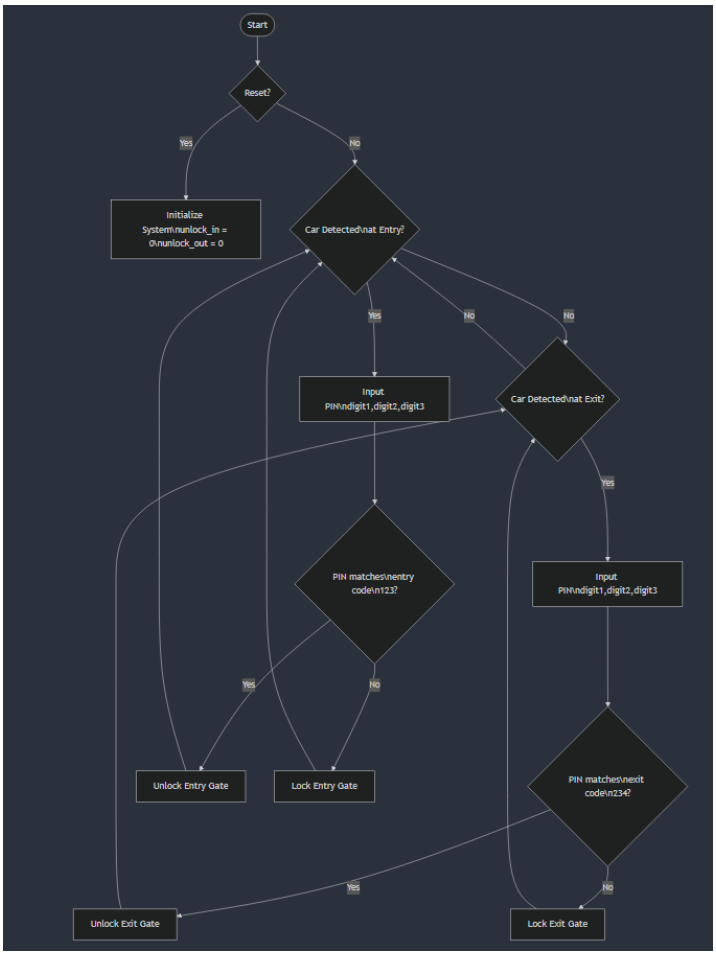
1.1. Algorithm

Langkah langkah yang dibutuhkan untuk dapat memenuhi semua fungsionalitas dari sistem

- Inisialisasi Sistem, Sistem dimulai dengan inisialisasi perangkat keras, di mana FPGA mengaktifkan semua modul yang dibutuhkan, termasuk modul keypad, sensor parkir, dan layar tampilan. Pada tahap ini, port I/O untuk keypad diatur sebagai input untuk membaca sinyal tombol. Semua komponen diinisialisasi agar siap untuk menerima dan memproses data dari pengguna.
- Pemantauan Input Keypad, Setelah sistem diinisialisasi, modul keypad mulai memantau input dari keypad. Proses ini dilakukan dengan metode scanning row-column matrix, di mana sistem secara berkala memeriksa kombinasi baris dan kolom untuk mendeteksi tombol yang ditekan oleh pengguna.
- Pengolahan Input Keypad, Ketika tombol terdeteksi, modul keypad handling memproses input tersebut. Input yang valid disimpan untuk keperluan lebih lanjut, seperti verifikasi data atau eksekusi perintah tertentu. Jika input tidak valid, sistem memberikan notifikasi kepada pengguna.
- Integrasi dengan Sistem Parkir, Setelah input diproses, data tersebut diteruskan ke modul utama sistem parkir untuk diverifikasi. Proses verifikasi ini melibatkan pengecekan data yang dimasukkan, seperti kecocokan kode parkir dengan database atau aturan yang berlaku dalam sistem.
- Pengendalian Mekanisme Parkir, Berdasarkan hasil verifikasi, sistem mengambil tindakan yang sesuai. Jika input valid, mekanisme parkir, seperti pembukaan atau penutupan gerbang, dijalankan. Selain itu, sistem juga dapat mengalokasikan tempat parkir secara otomatis berdasarkan data ketersediaan yang dimiliki.
- Penanganan Kesalahan, Jika terjadi kesalahan selama proses, seperti input yang tidak dikenali atau kegagalan perangkat dalam membaca tombol, sistem akan memberikan notifikasi kepada pengguna. Setelah kesalahan diatasi, sistem kembali ke kondisi siap untuk menerima input ulang tanpa mengganggu operasi lainnya.

a. Pseudocode and Flowchart
   - Start
   - Inisialisasi modul keypad
   - Mulai membaca input dari keypad

- Lakukan scanning baris dan kolom untuk mendeteksi tombol yang ditekan.
- Jika tombol ditekan
- Identifikasi tombol yang ditekan berdasarkan kombinasi baris dan kolom.
- Simpan data input tombol.
- Validasi input PIN
- Jika PIN benar Kirim sinyal ke modul kontrol untuk membuka palang pintu (servo).
- Jika PIN salah Kirim sinyal ke modul kontrol untuk tetap menutup palang pintu (servo).
- Kembali ke langkah membaca input untuk memantau tombol berikutnya
- End



1.2. How the System Works

a. Arsitektur Sistem

Sistem ini dirancang untuk mengelola parkir pintar dengan memanfaatkan berbagai komponen utama yang saling terintegrasi. Komponen pertama adalah motion sensor, yang berfungsi mendeteksi keberadaan kendaraan. Jika kendaraan terdeteksi, sensor ini akan mengaktifkan sistem keypad untuk menerima input dari pengguna. Modul keypad kemudian digunakan untuk mengambil input berupa PIN yang dimasukkan oleh pengguna. PIN ini akan divalidasi, dan hasil validasi tersebut dikirimkan ke mekanisme kontrol untuk mengatur gerakan servo

motor. Servo motor bertugas membuka atau menutup palang pintu berdasarkan hasil validasi PIN. Selain itu, sistem dilengkapi dengan parking slot monitoring untuk memantau ketersediaan slot parkir. Monitoring ini memberikan sinyal ke LED indikator, yang menyala jika slot terisi dan mati jika slot kosong. Semua proses ini dikendalikan oleh FPGA, yang bertindak sebagai pusat kendali untuk mengintegrasikan seluruh modul, memastikan sistem berjalan secara sinkron dan efisien.

b. Input, Output, dan Internal Signals

- Input : Keypad Input, Motion Sensor, Parking Slot Status.
- Output : Servo Signal, LED indikator
- Internal Signals : Keypad Row and Column Signals, PIN Validation Signal, Slot_Occupancy Counter, Motion Detected Signal, Parking Slot Signal

2. Design

This section is about creating and structuring your digital design.

2.1. Block System 1

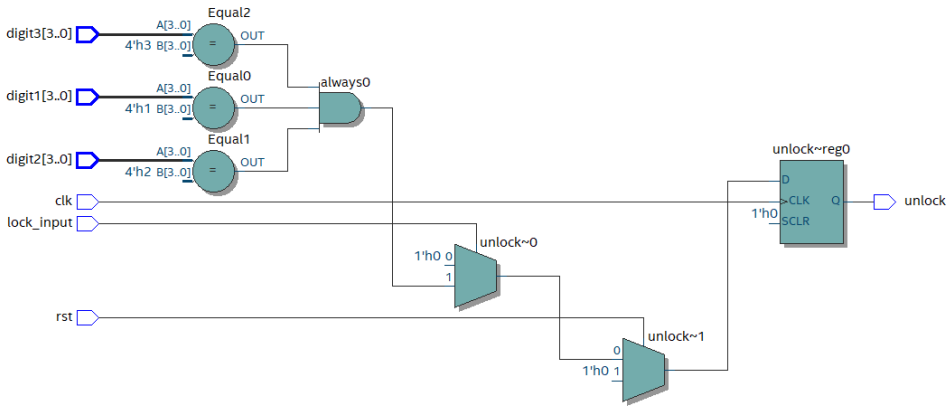


Diagram yang Anda unggah adalah rangkaian logika digital yang tampaknya merepresentasikan sistem pengunci digital dengan kombinasi angka sebagai inputnya. Berikut adalah penjelasan masing-masing blok:

1. Input digit[3:0]
   - Terdapat tiga input digit[3:0], yaitu digit3[3:0], digit1[3:0], dan digit2[3:0].
   - Ketiganya adalah masukan data 4-bit yang merepresentasikan angka kombinasi untuk kunci.
2. Blok Equal (Comparator)
   - Terdapat tiga blok pembanding (Equal2, Equal0, dan Equal1).
   - Fungsi pembanding ini adalah untuk membandingkan masing-masing input dengan nilai tertentu:

    - Equal2 membandingkan digit3[3:0] dengan nilai 4'h3.
    - Equal0 membandingkan digit1[3:0] dengan nilai 4'h1.
    - Equal1 membandingkan digit2[3:0] dengan nilai 4'h2.
  - Output dari pembanding akan bernilai 1 jika input sesuai dengan nilai pembanding.
3. AND Gate (always0)
  - Ketiga output pembanding (Equal2, Equal0, dan Equal1) dihubungkan ke gerbang logika AND.
  - Fungsi dari gerbang AND adalah untuk memastikan semua kondisi (ketiga pembanding) terpenuhi.
  - Outputnya akan bernilai 1 jika semua input sama dengan nilai yang diinginkan (4'h3, 4'h1, 4'h2).
4. Multiplexer
  - Terdapat dua multiplexer yang digunakan dalam rangkaian ini (unlock~0 dan unlock~1).
  - Multiplexer memilih input berdasarkan kondisi:
    - unlock~0: Memilih antara konstanta 1 atau output sebelumnya.
    - unlock~1: Memilih nilai berdasarkan kombinasi output multiplexer sebelumnya.
5. Register (unlock~reg0)
  - Register digunakan untuk menyimpan nilai status unlock.
  - Input register adalah hasil dari multiplexer terakhir.
  - Register ini dikontrol oleh sinyal clock (clk) dan reset (rst).
  - Jika reset aktif, output register akan kembali ke nilai awal (0).
6. Output (unlock)
  - Output utama dari rangkaian adalah sinyal unlock.
  - Sinyal ini akan aktif (1) jika kondisi kombinasi kunci yang dimasukkan sesuai dengan nilai yang diharapkan.

2.2. FSM Design (Finite State Machine Design)

a. Verilog

- code Keypad

```
module keypad(
    input wire [3:0] digit1,       // Digit pertama
    input wire [3:0] digit2,       // Digit kedua
    input wire [3:0] digit3,       // Digit ketiga
    input wire clk,
    input wire rst,
    input wire lock_input,
    output reg unlock
);

    // Password yang benar
    parameter [11:0] PASSWORD_IN  = 12'b000100100011;
    parameter [11:0] PASSWORD_OUT = 12'b001000110100;

    // Logika untuk unlock
    always @(posedge clk or posedge rst) begin
        if (rst) begin
            unlock <= 1'b0;
        end else if (lock_input) begin

            if ((digit1 == PASSWORD_IN[11:8] &&
                 digit2 == PASSWORD_IN[7:4] &&
                 digit3 == PASSWORD_IN[3:0]) ||
                (digit1 == PASSWORD_OUT[11:8] &&
                 digit2 == PASSWORD_OUT[7:4] &&
                 digit3 == PASSWORD_OUT[3:0])) begin
                unlock <= 1'b1;
            end else begin
                unlock <= 1'b0;
            end
        end else begin
            unlock <= 1'b0;
        end
    end

endmodule
```

- keypad handler code

```
module keypad_handling(
    input wire clk,
    input wire rst,
    input wire car_detected_in,
    input wire car_detected_out,
    input wire [3:0] digit1_in,
    input wire [3:0] digit2_in,
    input wire [3:0] digit3_in,
    input wire [3:0] digit1_out,
    input wire [3:0] digit2_out,
    input wire [3:0] digit3_out,
    output reg unlock_in,
    output reg unlock_out
);

    wire keypad_unlock_in;
    wire motion_out_in;
    reg lock_input_in;

    wire keypad_unlock_out;
    wire motion_out_out;
    reg lock_input_out;

    // Instansiasi motion_sensor untuk jalur masuk
    motion_sensor motion_sensor_in (
        .clk(clk),
        .rst(rst),
        .car_detected(car_detected_in),
        .motion_out(motion_out_in)
    );

    // Instansiasi keypad untuk jalur masuk
    keypad keypad_in (
        .digit1(digit1_in),
        .digit2(digit2_in),
        .digit3(digit3_in),
        .clk(clk),
        .rst(rst),
        .lock_input(motion_out_in),
        .unlock(keypad_unlock_in)
    );

    // Instansiasi motion_sensor untuk jalur keluar
    motion_sensor motion_sensor_out (
```

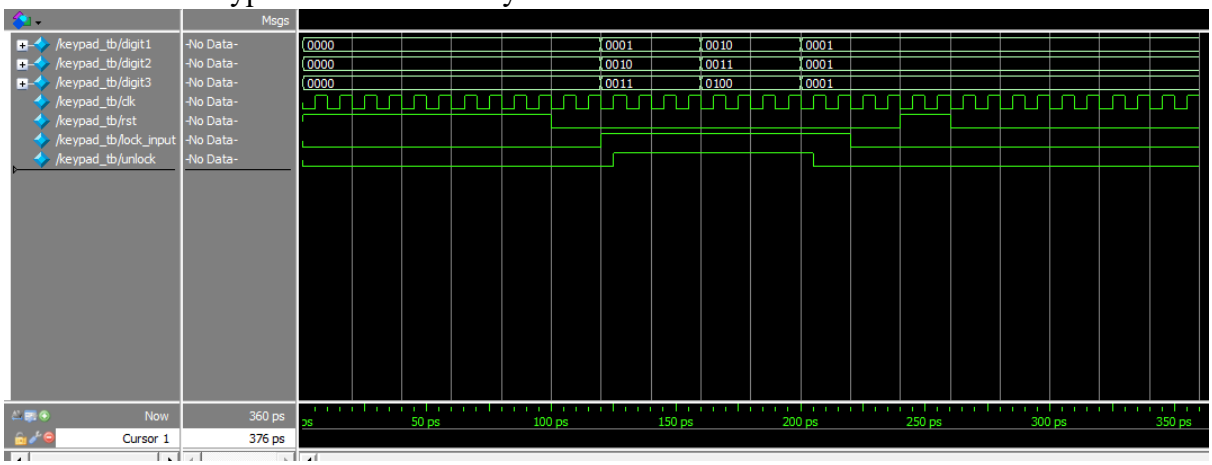
```
    .clk(clk),
    .rst(rst),
    .car_detected(car_detected_out),
    .motion_out(motion_out_out)
);

// Instansiasi keypad untuk jalur keluar
keypad keypad_out (
    .digit1(digit1_out),
    .digit2(digit2_out),
    .digit3(digit3_out),
    .clk(clk),
    .rst(rst),
    .lock_input(motion_out_out),
    .unlock(keypad_unlock_out)
);

// Logika untuk jalur masuk
always @(posedge clk or posedge rst) begin
    if (rst) begin
        unlock_in <= 1'b0;
        lock_input_in <= 1'b0;
    end else begin
        if (motion_out_in == 1'b1) begin
            lock_input_in <= 1'b1;
        end else begin
            lock_input_in <= 1'b0;
        end

        if (car_detected_in && lock_input_in) begin
            if (keypad_unlock_in == 1'b1) begin
                unlock_in <= 1'b1;
            end else begin
                unlock_in <= 1'b0;
            end
        end else begin
            unlock_in <= 1'b0;
        end
    end
end

// Logika untuk jalur keluar
always @(posedge clk or posedge rst) begin
    if (rst) begin
        unlock_out <= 1'b0;
        lock_input_out <= 1'b0;
    end else begin
        if (motion_out_out == 1'b1) begin
```

```
                lock_input_out <= 1'b1;
            end else begin
                lock_input_out <= 1'b0;
            end

            if (car_detected_out && lock_input_out) begin
                if (keypad_unlock_out == 1'b1) begin
                    unlock_out <= 1'b1;
                end else begin
                    unlock_out <= 1'b0;
                end
            end else begin
                unlock_out <= 1'b0;
            end
        end
    end

endmodule
```

3. Hardware Implementation

3.1. Simulation (Waveform Analysis)

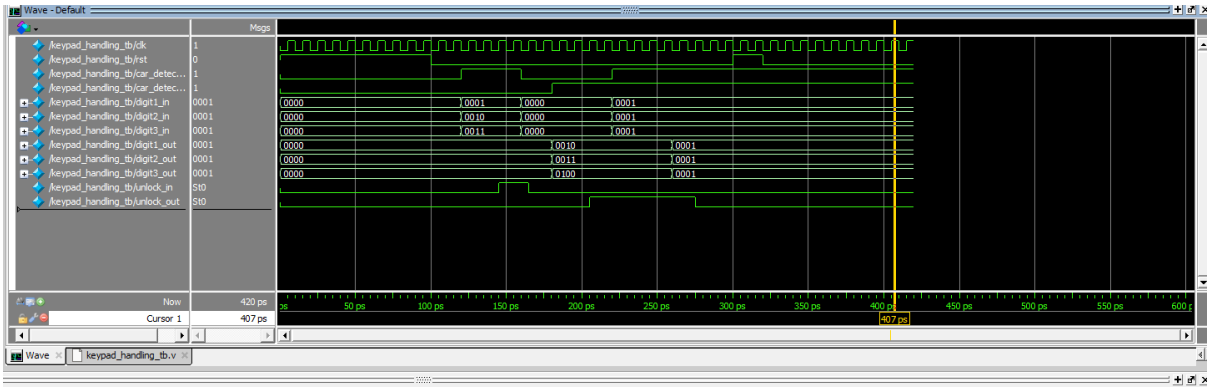
- Keypad Waveform Analysis



1. Input digit dimasukkan secara berurutan:
   - digit1: 0001 (1)
   - digit2: 0010 (2)
   - digit3: 0011 (3)
2. Clock (clk) berjalan normal dan konsisten
3. Sinyal unlock aktif setelah kombinasi "123" dimasukkan dan lock_input aktif, yang menunjukkan password yang dimasukkan benar

simulasi keypad berfungsi sesuai desain - mampu mendeteksi password yang benar dan menghasilkan sinyal unlock yang sesuai.

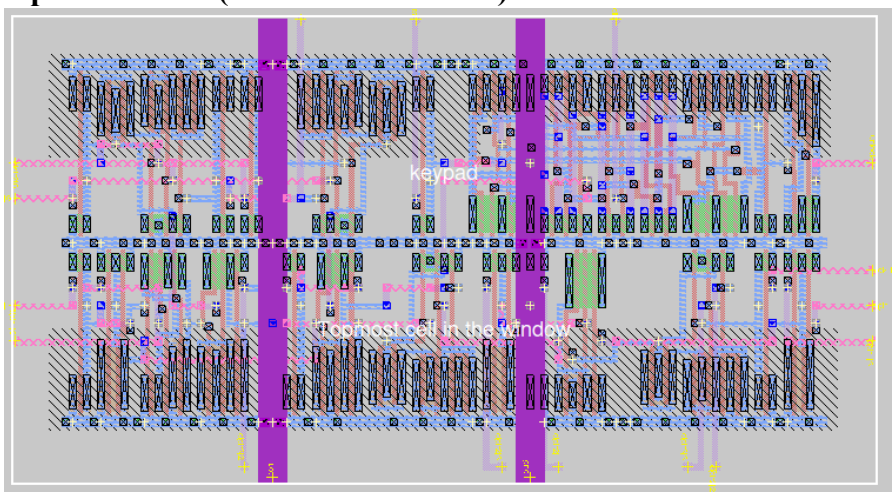
- Keypad Handler Waveform Analysis



1. Sistem berhasil menguji beberapa skenario:
    - Input password masuk yang benar (123) - unlock_in aktif
    - Input password keluar yang benar (234) - unlock_out aktif
    - Input password salah - tidak ada unlock yang aktif
2. Timing dan urutan sudah benar:
    - Reset awal berfungsi
    - Deteksi mobil (car_detected) terjadi sebelum input password
    - Sistem memberi respons unlock hanya ketika password benar dan ada deteksi mobil

Secara keseluruhan, waveform menunjukkan sistem keypad dan deteksi mobil berfungsi sesuai desain yang diinginkan.

3.2. Qflow Implementation (ASIC Semi-Custom)



3.2.1. Place and Route

- Vias: 5 total.
- Subcell Instances: 29 total.
- Pins: 18 total.
- Nets: 37 nets total.
- Special Nets: 2 total.

3.2.2. Power

- Daya Dinamis : 500fF, V=1.8V, frekuensi clock : 10 MHz
- Estimasi Total Daya : 8.1 µW

3.2.3. Area

- Area Total Core: 11,410,300 µm².
- Area Standard Cell: 635.04 µm².
- Efisiensi Area (Utilization): 5.56%.

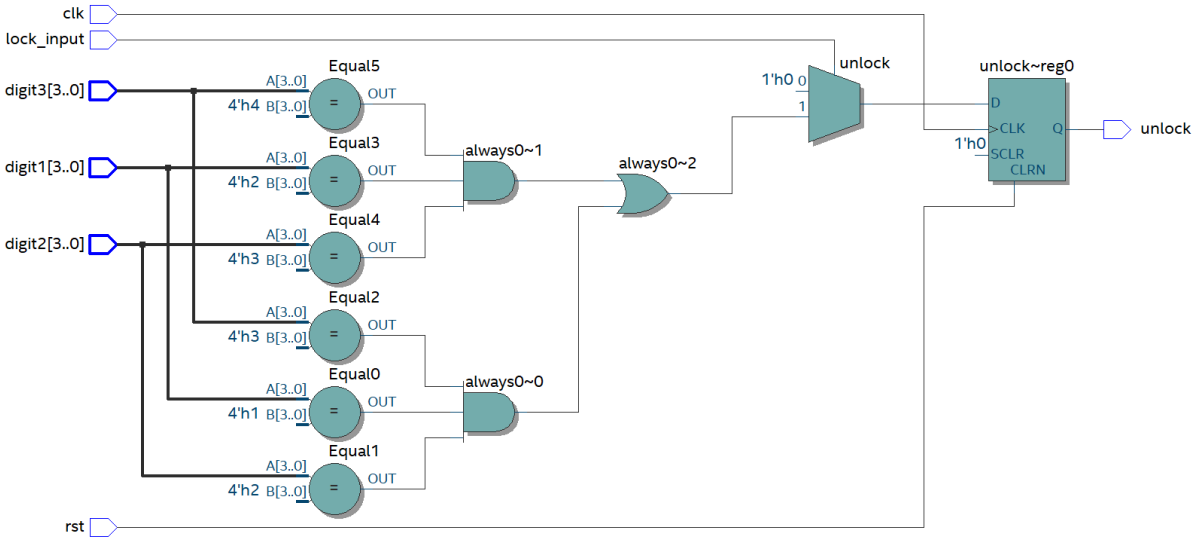
3.2.4. Timing

1. Jalur Waktu Maksimum (Setup Timing):
   - Jalur 1: Dari DFFSR_1/CLK ke pin output unlock delay : 310,068 ps. Dengan Frekuensi clock maksimum yang dicapai : 3225,1 MHz.
   - Jalur 2 : Dari input digit1[1] ke DFFSR_1/D dengan delay 397,804 ps Dengan waktu setup 89,8313 ps.
   - Jalur 3 : Dari clock ke DFFSR_1/CLK dengan delay: 222,929 ps. Dengan waktu setup 222,929 ps.
2. Jalur Waktu Minimum (Hold Timing):
   - Jalur 1: Dari clk ke DFFSR_1/CLK, dengan waktu hold 11,3 ps.
   - Jalur 2: Dari rst ke DFFSR_1/R, dengan waktu hold 10,49 ps.
   - Jalur 3: Dari input digit3[2] ke DFFSR_1/D, dengan waktu hold 26,1193 ps.

4. Results

4.1. Network Architecture

Keypad RTL



4.2. Optimization, Complexity, and Other Observable Variables

4.2.1. Timing

- Fmax : 161.32 MHz
- Slack Setup : 3.801 ns
- Slack Hold : 6.095 ns

4.2.2. Resource Usage

a. Logic Elements (LEs):
- Total: 6/6,272.
- Jenis: 5 kombinasi, 1 register saja.

b. Registers:
- Total: 1/6,684.
- Register ini adalah dedicated logic register.

c. I/O Pins:
- Total: 16/92 (17%).
- Clock Pins: 1/3 (33%).
- Dedicated Input Pins: Tidak ada.

d. Memory dan Multiplier:
- Block Memory: 0/30.
- DSP Elements: 0/30.

e. LABs (Logic Array Blocks):
- 6/392 LAB.

f. Interconnect Usage:
- Average interconnect usage : 0.1%.
- Peak interconnect usage : 0.2%.

4.2.3. Power Consumption

- Total Thermal Power Dissipation : 62.65 mW
- Core Static Thermal Power Dissipation : 42.82 mW
- I/O Thermal Power Dissipation : 19.83 mW
- Power Estimation Confidence : Low

5. Conclusion

Sistem parkir pintar ini, dengan integrasi keypad berbasis FPGA, bekerja dengan sangat efisien dan akurat untuk mengatasi kendala pada sistem parkir konvensional. Setelah inisialisasi perangkat keras yang tepat, sistem memantau dan memproses input dari pengguna. Input yang valid diproses dengan cepat dan diteruskan ke modul utama untuk diverifikasi, memastikan kecocokan dengan data yang berlaku. Berdasarkan hasil verifikasi, sistem dapat mengendalikan mekanisme parkir, seperti membuka gerbang atau mengalokasikan tempat parkir secara otomatis. Jika terjadi kesalahan, sistem memberikan notifikasi kepada pengguna dan siap untuk menerima input ulang, menjamin pengalaman yang lancar, responsif, dan efisien dalam pengelolaan parkir.