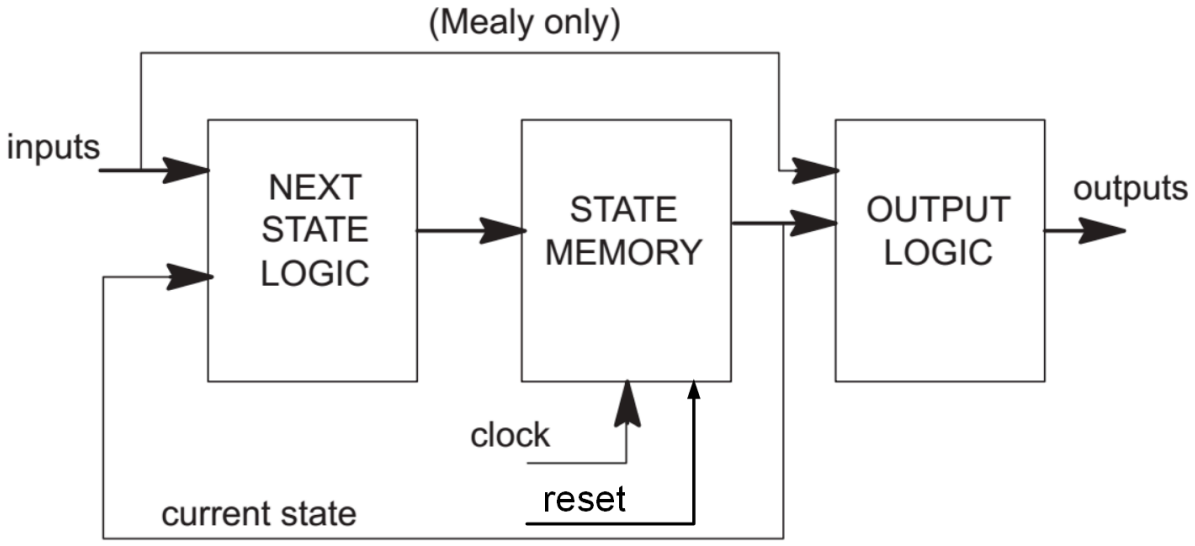
Finite State Machine

I. Định nghĩa về máy trạng thái hữu hạn (finite state machine)

Máy trạng thái hữu hạn, viết tắt là FSM, là một thành phần được sử dụng phổ biến trong thiết kế vi mạch số với ưu điểm là dễ kiểm soát quá trình hoạt động của thiết kế và dễ debug hoạt động của thiết kế.

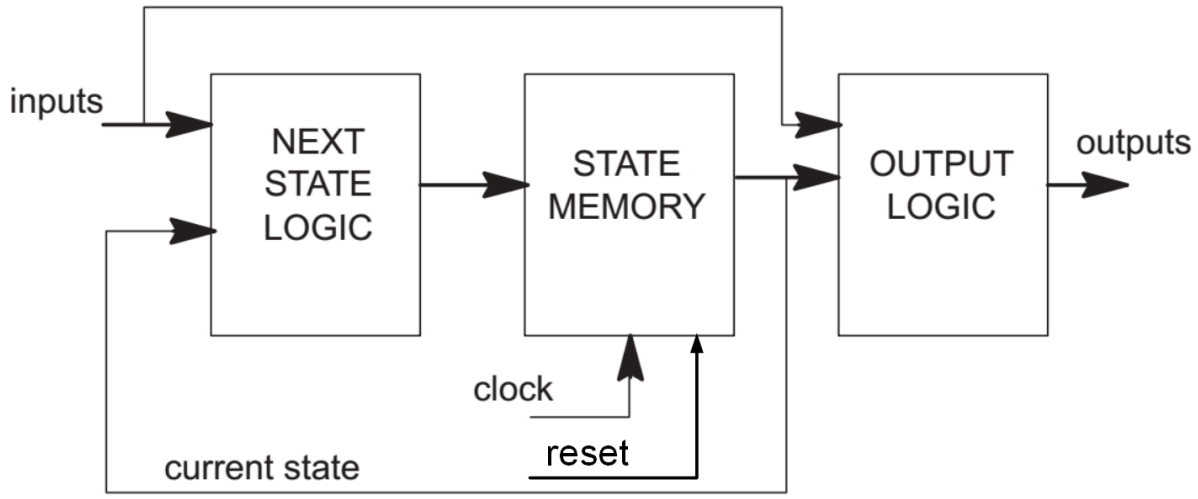
FSM gồm có 3 thành phần cơ bản như sau:

1. Mạch tạo trạng thái kế tiếp (Next state logic) là mạch tổ hợp phụ thuộc vào ngõ vào FSM và giá trị trạng thái hiện tại lấy từ bộ nhớ trạng thái (state memory)
2. Bộ nhớ trạng thái (state memory) là phần tử lưu trạng thái hiện tại của FSM nó có thể là Flip-Flop, Latch, ... lấy ngõ vào từ mạch tạo trạng thái kế tiếp. Bộ nhớ trạng thái thường được sử dụng trong các thiết kế đồng bộ là FF hoạt động theo xung clock. Một tín hiệu reset có thể phải sử dụng để khởi động FSM đến một giá trị ban đầu. Reset không cần sử dụng đối với các FSM luôn hoạt động đúng dù giá trị ban đầu của FF là bao nhiêu.
3. Mạch tạo ngõ ra (output logic) là mạch tổ hợp tạo giá trị ngõ ra tương ứng với trạng thái hiện tại của FSM. Mạch này lấy ngõ vào là giá trị trạng thái hiện tại và có thể tổ hợp thêm ngõ vào của FSM.



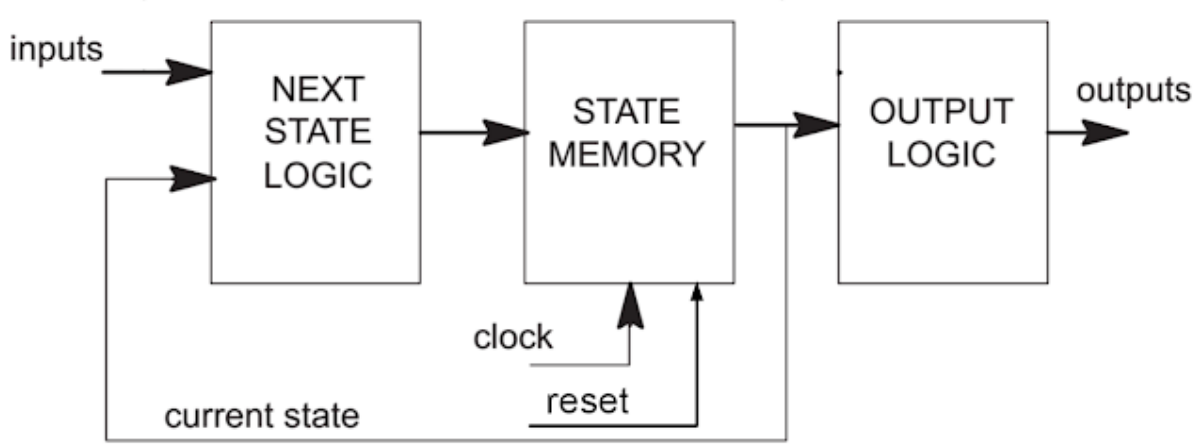
Hình: Mô hình cơ bản của FSM

–Máy trạng thái Mealy (Mealy machine): Máy trạng thái Mealy là một máy trạng thái mà dữ liệu đầu ra được quyết định bởi trạng thái hiện tại và các dữ liệu ngõ vào.



Hình: FSM Mealy

–**Máy trạng thái Moore (Moore machine):** Máy trạng thái Moore là máy trạng thái mà dữ liệu ngõ ra được quyết định duy nhất bởi trạng thái hiện tại.



Hình: FSM Moore

*/*Hình ảnh được lấy từ https://www.semiconvn.com*/*

-Các bước xây dựng một mạch tuần tự

Bước 1: Tìm các biến đầu vào, biến đầu ra và biến trạng thái của các phần tử nhớ.

Bước 2: Xác định số lượng biến trạng thái chính là số lượng flip flop có trong mạch.

Bước 3: Xác định số lượng trạng thái có thể : 2^n, n là số lượng biến trạng thái.

Bước 4: Tìm đầu vào kích thích cho mỗi flip flop.

Bước 5: Dùng bảng đặc trưng và bảng kích thích của mỗi flip flop để xác định trạng thái tiếp theo khi trạng thái hiện tại đã biết.

Bước 6: Chuẩn bị bảng đặc trưng từ phương trình đặc trưng của flip flop và đầu ra. Chuẩn bị bảng chuyển trạng thái từ bảng đặc trưng. Vẽ bảng thể hiện trạng thái hiện tại, trạng thái tiếp theo khi đầu vào là 0, trạng thái tiếp theo khi đầu vào là 1, đầu ra khi đầu vào là 0, đầu ra khi đầu vào là 1.

Bước 7: Gán các kí tự hợp lí cho các biến.

Bước 8: Vẽ sơ đồ trạng thái.

II. Traffic light

Traffic light

Traffic light: Two lane A and B
- **S0:** Wait until there a no people in lane A
- **S1:** Count down before check lane B
- **S2:** Wait until there are no people in lane B
- **S3:** Count down before check A

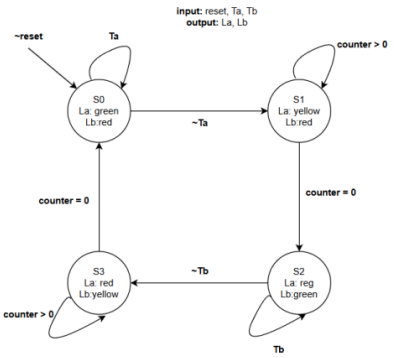


Fig 6. Traffic light FSM

Current State S	Inputs T_A	Inputs T_B	Next State S'
S0	0	X	S1
S0	1	X	S0
S1	X	X	S2
S2	X	0	S3
S2	X	1	S2
S3	X	X	S0

Fig 7. State transition table

State	Output
S0	LA: green, LB: red
S1	LA: yellow, LB: red
S2	LA: red, LB: green
S3	LA: red, LB: yellow

Output table

1. Các trạng thái (State)

Hệ thống có 4 trạng thái chính:

- S0: Đèn La: xanh, Lb: đỏ.
  - Chờ đến khi làn A không còn người (`~Ta`).
- S1: Đèn La: vàng, Lb: đỏ.
  - Đếm ngược trước khi kiểm tra làn B.
- S2: Đèn La: đỏ, Lb: xanh.
  - Chờ đến khi làn B không còn người (`~Tb`).

- S3: Đèn La: đỏ, Lb: vàng.
    - Đếm ngược trước khi kiểm tra làn A.

2. Cấu trúc FSM

FSM được mô tả bằng hình trạng thái (Fig 6), trong đó:

- Mỗi trạng thái được liên kết với một logic điều kiện để chuyển đổi.
- Các điều kiện bao gồm:
    - `Ta`: Có người trong làn A.
    - `Tb`: Có người trong làn B.
    - Counter > 0: Bộ đếm chưa hết.

Quy trình chuyển đổi trạng thái:

- Từ S0 đến S1: Khi làn A không còn người (`~Ta`).
- Từ S1 đến S2: Khi bộ đếm ở trạng thái S1 bằng 0.
- Từ S2 đến S3: Khi làn B không còn người (`~Tb`).
- Từ S3 đến S0: Khi bộ đếm ở trạng thái S3 bằng 0.

3. Bảng chuyển trạng thái (Fig 7)

Bảng này mô tả cách FSM chuyển đổi giữa các trạng thái:

- Cột Current State S: Trạng thái hiện tại.
- Cột Inputs Ta, Tb: Các tín hiệu đầu vào từ cảm biến.
- Cột Next State S': Trạng thái kế tiếp dựa trên tín hiệu.

Ví dụ:

- Ở trạng thái S0:
    - Nếu `Ta = 0` (không có người ở làn A): Chuyển sang S1.
    - Nếu `Ta = 1` (có người ở làn A): Giữ nguyên S0.

4. Bảng đầu ra (Output table)

Bảng này hiển thị trạng thái của các đèn giao thông:

- La: Đèn giao thông ở làn A (RGB: xanh, vàng, đỏ).
- Lb: Đèn giao thông ở làn B (RGB: xanh, vàng, đỏ).

Ví dụ:

- Ở trạng thái S0:

- La: Xanh (xe ở làn A được phép đi).
- Lb: Đỏ (xe ở làn B phải dừng).

5. Ứng dụng

FSM này được sử dụng để:

- Điều khiển đèn giao thông tự động tại ngã tư có hai làn đường.
- Đảm bảo mỗi làn được ưu tiên lần lượt dựa trên trạng thái giao thông.
- Giảm thiểu xung đột giữa các làn và tăng hiệu quả giao thông.

6.Code HDL

Các khối chính trong đoạn mã

Khối chia tần số (`clk_div_period`)

```
always @(posedge clk, negedge reset_n) begin
    if (~reset_n)
        counter_clk <= 0;
    else
        if (counter_clk == clk_div_period - 1)
            counter_clk <= 0;
        else
            counter_clk <= counter_clk + 1;
end
```

- Chia xung nhịp thành 1 giây bằng cách sử dụng `counter_clk`.
- Khi đạt đến giá trị `clk_div_period - 1`, bộ đếm được đặt lại.

Khối đếm ngược thời gian

```
always @(posedge clk, negedge reset_n) begin
    if (~reset_n)
        counter_sec <= 4;
    else begin
```

```
        if (counter_reset)
            counter_sec <= 4;
        else if (counter_clk == clk_div_period - 1)
            counter_sec <= counter_sec - 1;
    end
end
```

- `counter_sec`: Đếm số giây cho mỗi trạng thái đèn.
- Bộ đếm được đặt lại khi tín hiệu `counter_reset` được kích hoạt.

Khối trạng thái (`current_state`)

```
always @(posedge clk, negedge reset_n) begin
    if (~reset_n)
        current_state <= S0;
    else
        current_state <= next_state;
end
```

- `current_state`: Trạng thái hiện tại của hệ thống (S0, S1, S2, S3).
- `next_state`: Trạng thái kế tiếp dựa trên logic chuyển đổi trạng thái.

Chuyển đổi trạng thái (`current_state -> next_state`)

```
always @* begin
    case (current_state)
        S0:
            if (~Ta)
                next_state = S1;
            else
```

```
                next_state = S0;
        S1:
            if (counter_sec == 0)
                next_state = S2;
            else
                next_state = S1;
        S2:
            if (~Tb)
                next_state = S3;
            else
                next_state = S2;
        S3:
            if (counter_sec == 0)
                next_state = S0;
            else
                next_state = S3;
        default:
            next_state = S0;
    endcase
end
```

- **S0** → **S1**: Khi `Ta = 0` (xe ở làn A đã hết).
- **S1** → **S2**: Khi thời gian đếm ngược (`counter_sec`) của trạng thái S1 kết thúc.
- **S2** → **S3**: Khi `Tb = 0` (xe ở làn B đã hết).
- **S3** → **S0**: Khi thời gian đếm ngược (`counter_sec`) của trạng thái S3 kết thúc.

Đầu ra đèn giao thông (`current_state -> output`)

```
always @* begin
    case (current_state)
        S0:
            {La, Lb} = {GREEN, RED};
        S1:
            {La, Lb} = {YELLOW, RED};
        S2:
            {La, Lb} = {RED, GREEN};
        S3:
            {La, Lb} = {RED, YELLOW};
        default:
            {La, Lb} = 0;
    endcase
end
```

- Tại mỗi trạng thái, đèn `La` và `Lb` được đặt theo quy định:
  - `S0`: La xanh, Lb đỏ.
  - `S1`: La vàng, Lb đỏ.
  - `S2`: La đỏ, Lb xanh.
  - `S3`: La đỏ, Lb vàng.

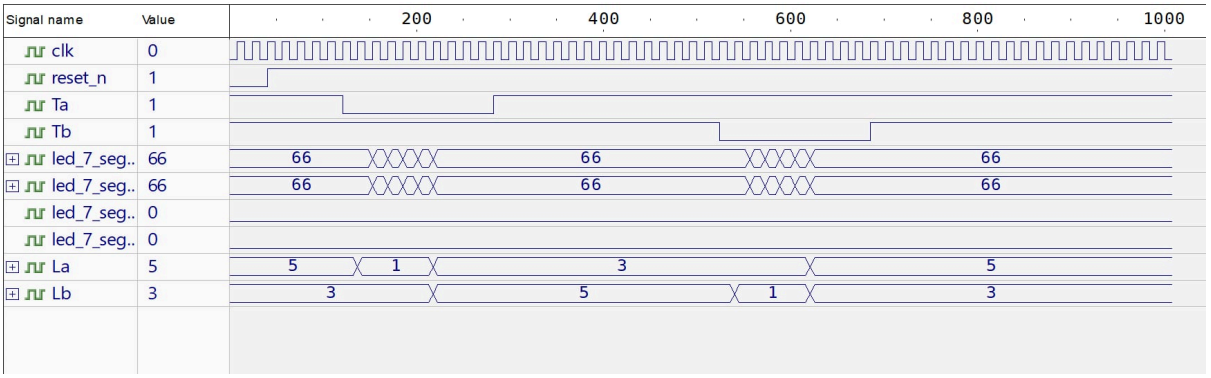
Hiển thị số giây trên LED 7 đoạn

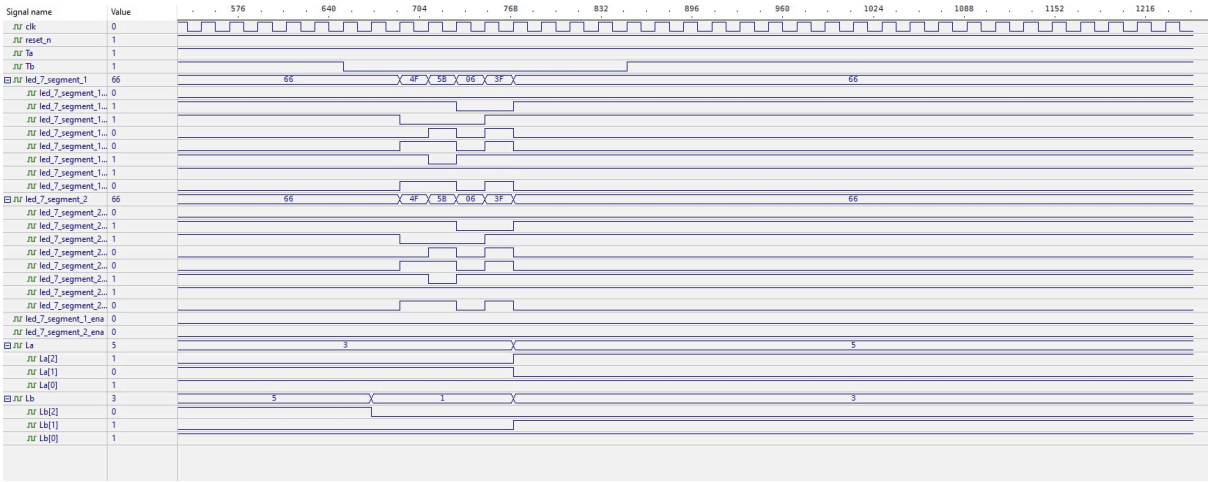
```
always @* begin
    case (counter_sec_wire)
        0: led_7_segment_1 = 8'h3F;
        1: led_7_segment_1 = 8'h06;
```

```
            ...
            default: led_7_segment_1 = 8'hFF;
        endcase
end
```

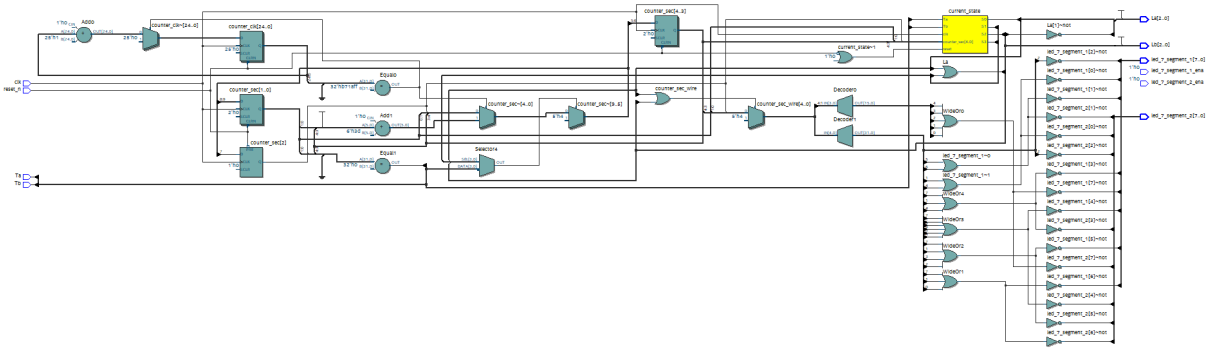
- Hiển thị số giây đếm ngược còn lại lên `led_7_segment_1` và `led_7_segment_2`.
- Các giá trị HEX tương ứng với chữ số hiển thị trên LED 7 đoạn.

7. Testbench và mô phỏng mạch trên ModelSim





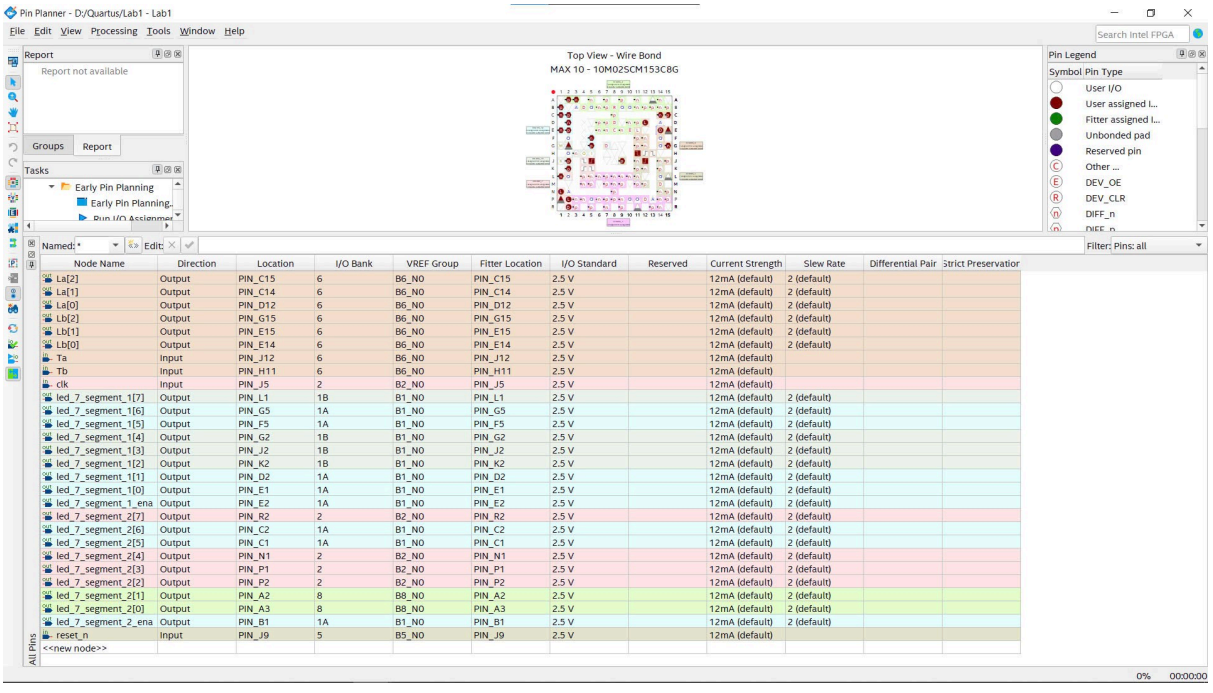
8. Mô phỏng mạch trên Quartus



9.Mô phỏng trên bo mạch FPGA

Video mô phỏng trên KIT

👉 TraficLight.mp4



Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Slew Rate	Differential Pair	Strict Preservation
La[2]	Output	PIN_C15	6	B6_N0	PIN_C15	2.5 V		12mA (default)	2 (default)		
La[1]	Output	PIN_C14	6	B6_N0	PIN_C14	2.5 V		12mA (default)	2 (default)		
La[0]	Output	PIN_D12	6	B6_N0	PIN_D12	2.5 V		12mA (default)	2 (default)		
Lb[2]	Output	PIN_G15	6	B6_N0	PIN_G15	2.5 V		12mA (default)	2 (default)		
Lb[1]	Output	PIN_E15	6	B6_N0	PIN_E15	2.5 V		12mA (default)	2 (default)		
Lb[0]	Output	PIN_E14	6	B6_N0	PIN_E14	2.5 V		12mA (default)	2 (default)		
Ta	Input	PIN_J12	6	B6_N0	PIN_J12	2.5 V		12mA (default)			
Tb	Input	PIN_H11	6	B6_N0	PIN_H11	2.5 V		12mA (default)			
clk	Input	PIN_J5	2	B2_N0	PIN_J5	2.5 V		12mA (default)			
led_7_segment_1[7]	Output	PIN_L1	1B	B1_N0	PIN_L1	2.5 V		12mA (default)	2 (default)		
led_7_segment_1[6]	Output	PIN_G5	1A	B1_N0	PIN_G5	2.5 V		12mA (default)	2 (default)		
led_7_segment_1[5]	Output	PIN_F5	1A	B1_N0	PIN_F5	2.5 V		12mA (default)	2 (default)		
led_7_segment_1[4]	Output	PIN_G2	1B	B1_N0	PIN_G2	2.5 V		12mA (default)	2 (default)		
led_7_segment_1[3]	Output	PIN_J2	1B	B1_N0	PIN_J2	2.5 V		12mA (default)	2 (default)		
led_7_segment_1[2]	Output	PIN_K2	1B	B1_N0	PIN_K2	2.5 V		12mA (default)	2 (default)		
led_7_segment_1[1]	Output	PIN_D2	1A	B1_N0	PIN_D2	2.5 V		12mA (default)	2 (default)		
led_7_segment_1[0]	Output	PIN_E1	1A	B1_N0	PIN_E1	2.5 V		12mA (default)	2 (default)		
led_7_segment_1_ena	Output	PIN_E2	1A	B1_N0	PIN_E2	2.5 V		12mA (default)	2 (default)		
led_7_segment_2[7]	Output	PIN_R2	2	B2_N0	PIN_R2	2.5 V		12mA (default)	2 (default)		
led_7_segment_2[6]	Output	PIN_C2	1A	B1_N0	PIN_C2	2.5 V		12mA (default)	2 (default)		
led_7_segment_2[5]	Output	PIN_C1	1A	B1_N0	PIN_C1	2.5 V		12mA (default)	2 (default)		
led_7_segment_2[4]	Output	PIN_N1	2	B2_N0	PIN_N1	2.5 V		12mA (default)	2 (default)		
led_7_segment_2[3]	Output	PIN_P1	2	B2_N0	PIN_P1	2.5 V		12mA (default)	2 (default)		
led_7_segment_2[2]	Output	PIN_P2	2	B2_N0	PIN_P2	2.5 V		12mA (default)	2 (default)		
led_7_segment_2[1]	Output	PIN_A2	8	B8_N0	PIN_A2	2.5 V		12mA (default)	2 (default)		
led_7_segment_2[0]	Output	PIN_A3	8	B8_N0	PIN_A3	2.5 V		12mA (default)	2 (default)		
led_7_segment_2_ena	Output	PIN_B1	1A	B1_N0	PIN_B1	2.5 V		12mA (default)	2 (default)		
reset_n	Input	PIN_J9	5	B5_N0	PIN_J9	2.5 V		12mA (default)			
<<new node>>											