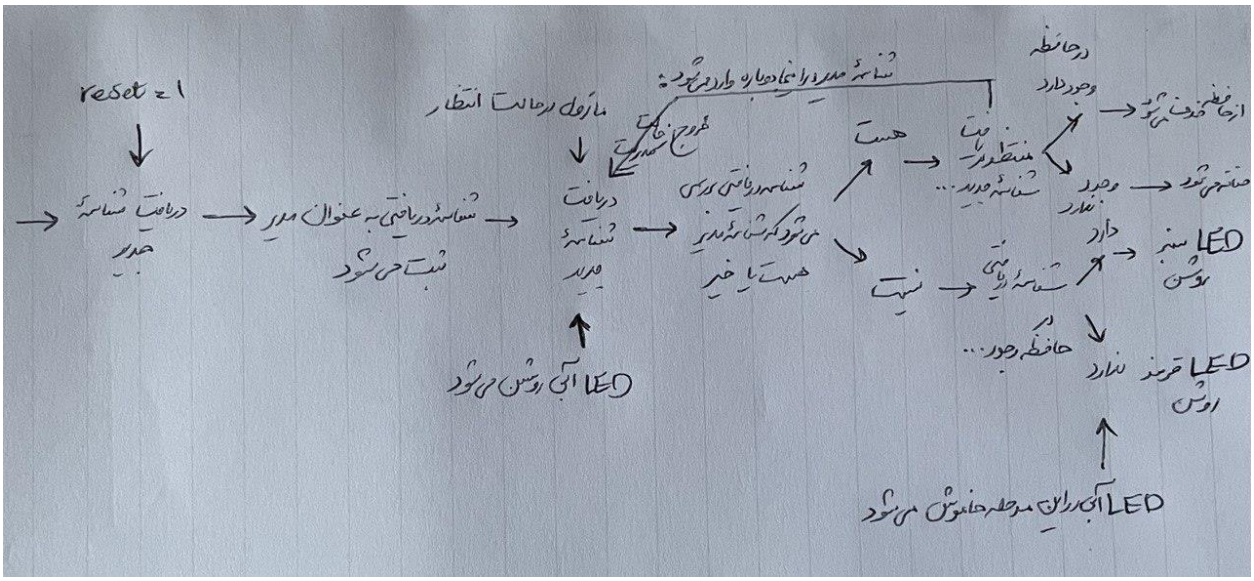
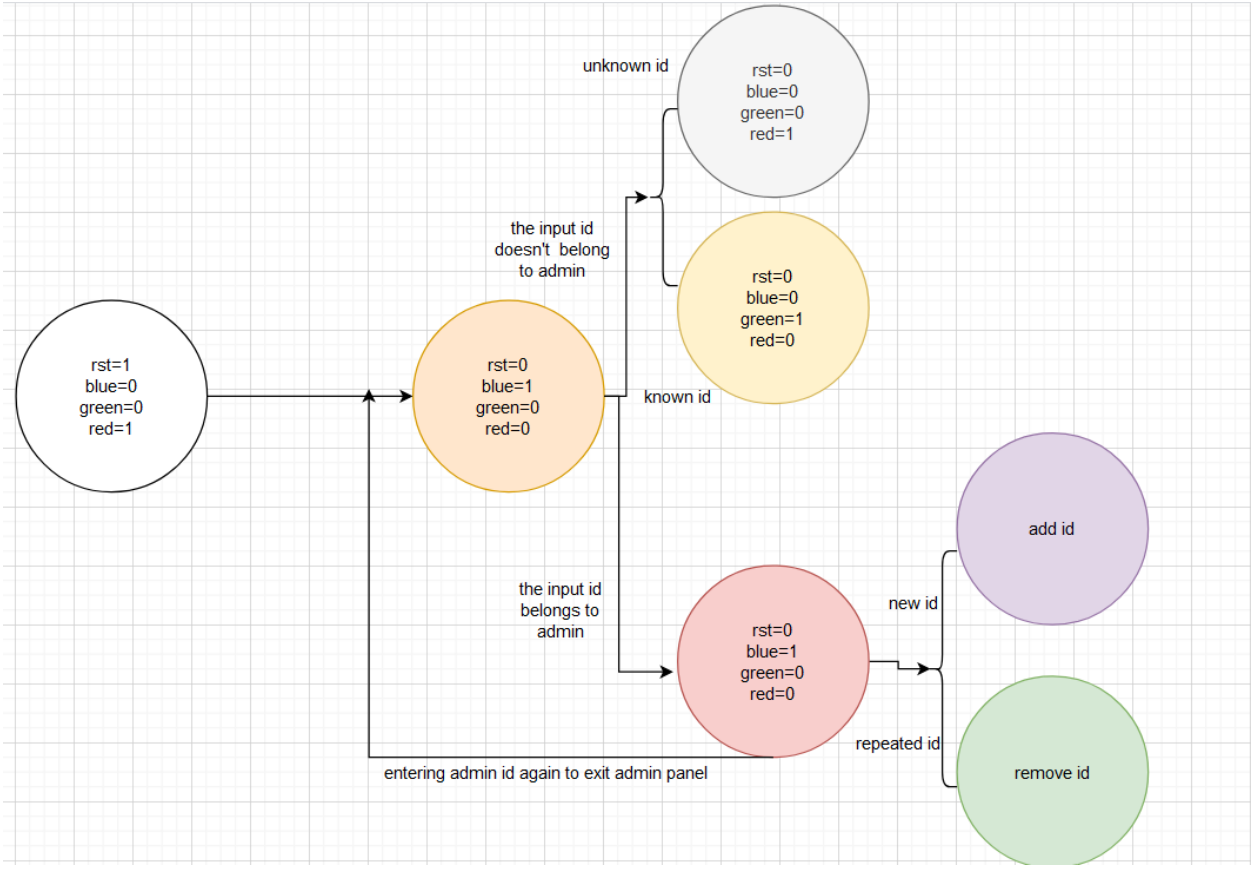
# گزارش کد چهارم

نگار هنرور صدیقیان
99243076

در این تمرین به دنبال پیاده‌سازی یک ماژول کنترل دستی هستیم. برای اینکار لازم است ابتدا درک درستی از خواسته سوال داشته باشیم:




دیاگرام فوق خلاصه ای از آنچه که انتظار می‌رود توسط ماژول دستی انجام شود را نشان می‌دهد. برای تبدیل آن به کد آن را به State Diagram تبدیل می‌کنیم:



حال کافی است تا state diagram صفحه قبل را به کد تبدیل کنیم . برای اینکار به مجموعه ای از شروط نیاز داریم که شرط های هر مرحله با توجه به state diagram فوق قابل برداشت است. بدین ترتیب خواهیم داشت:

- ورودی های کد ما به صورت زیر هستند:

1-یک clk داریم که در هرلبه بالارونده آن از یک وضعیت به وضعیت دیگرجابه‌جا می‌شویم.

2- یک رجیستر داریم که شناسه 4 بایتی ورودی را در خود نگه میدارد و آن را ID می‌نامیم.

3- سه ورودی تک بیتی تحت عنوانred_led,green_led و blue_led داریم که هریک نماینده یکی از چراغ هاست،در زمان روشن شدن مقدار آن‌ها برابر 1 و در زمان خاموشی برابر 0 است.

4- یک سیگنال rst داریم که در زمان 1 شدن ماژول کنترل‌کننده به وضعیت آغازین رفته و اولین شناسه دریافتی پس از آن را به عنوان شناسه مدیر ثبت می‌کند.

5- متغیر تک بیتی Lock که در صورت 1 بودن به معنای باز بودن در و در زمان 0 بودن به معنای بسته شدن در است.

حال به دنبال تبدیل state های state diagram به وضعیت هایی شرط بندی شده هستیم. طبق آنچه در state diagram می‌بینیم باید به کمک localparam ، 7 وضعیت تعریف کنیم چون 7 state داریم.دلیل استفاده از localparam این است که در مقایسه با پارامتر معمولی مشابه با constant رفتار کرده و از بروز خطای احتمالی جلوگیری می‌کند.این متغیر 4 بیتی تعریف میشود و هر بیت آن به ترتیب نماینده چراغ ال ای دی آبی،سبز،قرمز و قفل بودن در است.

ورودی ها و وضعیت‌های تعریف شده تا به اینجای کار مشابه قطعه کد زیر هستند:

```
module controller (
    input clk,
    input rst,
    input[31:0] ID,
    output reg red_led,
    output reg blue_led,
    output reg green_led,
    output reg Lock
);
    localparam st0 = 4'd0; // first state , the rst value = 0 and the input id=admin id
    localparam st1 = 4'd1; // add new id, check if it belongs to admin or not
    localparam st2 = 4'd2; // go to admin Mode
    localparam st3 = 4'd3; // add new user to memory
    localparam st4 = 4'd4; // delete user from memory
    localparam st5 = 4'd6; // known user in user mode (green Led)
    localparam st6 = 4'd7; // unkown user in user mode (red Led)
```

حال نیازمند تعریف مجموعه‌ای از متغیرها هستیم که در روند کد به ما کمک خواهند کرد:

- به دو متغیر یکی برای ذخیره وضعیت فعلی و دیگری برای ذخیره وضعیت بعدی ای که به آن جابه‌جا میشویم نیاز داریم. به این منظور دو متغیر 4 بیتی nextState و State را تعریف میکنیم که اطلاعات هر وضعیتی که در آن قرار میگیریم و میخواهیم به آن جابه‌جا شویم را در خود ذخیره میکنند. چون محتویات آن یکی از state های بخش قبل است پس مشابه آن 4 بیتی تعریف میشود.
- نیازمند ساختن یک مموری هستیم به این منظور آرایه‌ای 9 تایی از رجیسترهای 32 بیتی به نام memory تعریف میکنیم که شناسه های خوانده شده در آن ذخیره شوند. از اینجا واضح است که بیشتر از 9 شناسه را نمیتوانیم ذخیره کنیم.
- به دو متغیر عددی size و I نیاز داریم که اولی نشان‌دهنده تعداد خانه های پر شده مموری ای که ساخته ایم میباشد؛در صورت اضافه شدن ای دی جدید یکی به آن افزوده شده و با حذف ای دی یک واحد از آن کم میشود. متغیر عددی دوم I است. از این متغیر در حلقه و برای پیمایش مموری استفاده میکنیم.

متغیرهای گفته شده مطابق توضیح فوق هستند:

```
reg [31:0] memory [8:0];
reg [3:0] state, nextState;
integer size  = 0;
integer i;
```

در مرحله بعدی به دنبال تعریف تعدادی متغیر هستیم که به ما در عملکرد صحیح هریک از وضعیت‌ها کمک میکنند:

- ابتدا یک رجیستر 32 بیتی برای نگهداری مقدار شناسه ادمین در نظر میگیریم.
- یک متغیر 8 بیتی که شماره ایندکس شناسه‌ای که قرار است حذف شود را در خود نگه میدارد تا طی فرآیند حذف خانه‌های پس از آن به درستی به یک خانه قبل از خود شیفت شوند.
- یک متغیر تک بیتی flag تعریف میکنیم که اگر 1 باشد یعنی در وضعیتی هستیم که یوزر کنترل را در دست دارد و اگر 0 باشد یعنی در وضعیتی هستیم که کنترل دست ادمین است. این متغیر به ما درجابه‌جا شدن میان وضعیت‌ها کمک می‌کند.

```
reg[31:0] admin = 32'd0;
reg flag = 0;
reg [7:0] removed_index = 7'd0;
```

در مرحله بعدی کلاک را تعریف میکنیم:

```
always @(posedge clk,posedge rst) begin
    if (rst) begin
        state = st0;
    end else begin
        state = nextState;
    end
end
```

از این بخش با توجه به وضعیت سیگنال rst و لبه کلاک ، اگر rst=1 باشد وارد state 0 کنترل کننده میشویم ، در غیر این صورت با هر لبه بالارونده کلاک از وضعیت فعلی به وضعیت بعدی جابه‌جا می‌شویم.

بخش بعدی بخش تعریف وضعیت هاست، در این بخش با دریافت ایدی و مقدار state به یکی از 7 وضعیت تعریف شده جابه‌جا میشویم و متغیرهای اولیه براساس اینکه در کدام وضعیت هستیم مقداردهی میشوند.

1- وضعیت 0 – state 0

این وضعیت که در ماشین حالت با رنگ سفید مشخص شده وضعیتی است که منتظر دریافت ای دی جدید و اختصاص آن به عنوان ای دی ادمین هستیم. در این حالت هیچ یک از لامپ ها روشن نیستند و اگر ایدی ادمین مخالف صفر باشد به وضعیت بعدی که وضعیت 1 است منتقل میشویم.

```
always @(ID, state) begin
    case (state)
        st0: begin
            for (i = 0;i < size ; i= i + 1 ) begin
                memory[i] = 32'bz;
            end
        red_led = 0;
        blue_led = 0;
        green_led = 0;
        Lock = 0;
        admin = ID;
        if (admin!=0) begin
            nextState = st1;
        end
    end
```

2- وضعیت state 1-1

در این وضعیت که با رنگ نارنجی مشخص شده است ابتدا بررسی میشود که ایدی ورودی به ادمین تعلق دارد یا خیر. اگر متعلق به ادمین بود در لبه بالارونده بعدی کلاک به وضعیت دوم (قرمز) منتقل میشویم، در غیر این صورت ال ای دی آبی روشن میشود و چک میکنیم که آیا ایدی جدید است یا خیر؛اگر در مموری حضور داشت به وضعیت پنجم (زرد) و اگر حضور نداشت به وضعیت ششم (آبی) میرویم.

```
    st1: begin
        red_led = 0;
        blue_led = 1;
        green_led = 0;
        Lock = 0;
        if (ID == admin) begin
            nextState = st2;
        end
        else if (ID != admin) begin
            red_led = 0;
            blue_led = 1;
            green_led = 0;
            flag = 0;
            for (i = 0; i<size; i = i+1) begin
                if (memory[i]== ID) begin
                    flag = 1;
                end
            end
            if (flag) begin
                nextState = st5;
            end else begin
             nextState = st6;
            end
        end
    end
```

### 3-وضعیت state 2-2

در این وضعیت که با رنگ قرمز مشخص شده است در حالتی هستیم که کنترل در دست ادمین است(falg=0) در این وضعیت ایدی ورودی را بررسی میکنیم.اگر برابر با یکی از ایدی‌های ذخیره شده در مموری باشد از مموری پاک میشود،flag=1 و به وضعیت چهارم(سبز)منتقل میشویم،اگر در مموری موجود نباشد به وضعیت سوم (بنفش) میرویم و اگر برابر با ایدی ادمین باشد به معنای خروج از حالت ادمین است و مارا به وضعیت 1 منتقل میکند.

```
st2: begin
    flag = 0;
    red_led = 0;
    blue_led = 1;
    green_led = 0;
    if (ID != admin) begin
        for (i = 0;i < size ; i = i + 1) begin
            if (ID == memory[i]) begin
                removed_index = i;
                flag = 1;
            end
        end
         if (flag) begin
            nextState = st4;
        end
        if(!flag) begin
            nextState = st3;
        end
    end
    if (ID == admin) begin
        nextState = st1;
    end
end
```

### 4-وضعیت state 3-3

در این وضعیت که با رنگ بنفش مشخص شده است ایدی جدید به مموری اضافه میشود،مقدار متغیرsize یک واحد افزایش میابد و مجددا به وضعیت دوم بازمیگردیم.

```
st3: begin
    memory[size] = ID;
    size <= size + 1;
    nextState = st2;
    red_led = 1;
    blue_led = 1;
    green_led = 1;
    Lock = 1;
end
```

5-وضعیت State 4-4

در این وضعیت که با رنگ بنفش مشخص شده است ایدی هایی که بعد از ایدی تکراری حذف شده در حافظه قرار داشتند را یک خانه به چپ شیفت میدهیم،مقدارsizeرا یک واحد کم میکنیم ،در این وضعیت نیز مانند وضعیت قبلی همه‌ی ال ای دی ها در حالت روشن (میتوانستند خاموش هم باشند) قرار میگیرند و با کلاک بعدی به وضعیت دوم برمیگردیم.

```
st4: begin
    for (i = removed_index;i<size;i = i + 1 ) begin
        memory[i] = memory[i+1];
    end
    size = size - 1;
    red_led = 1;
    blue_led = 1;
    green_led = 1;
    Lock = 1;
    nextState = st2;
end
```

6- وضعیت state 5-5

در این حالت (زرد) کنترل در دست یوزر است و ایدی وارد شده از ایدی های موجود در حافظه میباشد. بنابراین در باز شده و ال ای دی سبز روشن میشود و با کلاک بعدی به وضعیت اول منتقل میشویم.

```
st5: begin
    red_led = 0;
    blue_led = 0;
    green_led = 1;
    Lock = 1;
    state = st1;
end
```

7-وضعیت state6-6

این وضعیت (آبی) مشابه وضعیت قبلی بوده با این تفاوت که ای دی ناشناخته است و در باز نمیشود و ال ای دی قرمز روشن میشود.

```
st6: begin
    red_led = 1;
    blue_led = 0;
    green_led = 0;
    Lock = 0;
    state = st1;
end
```

در انتها نیز طبق قاعده سوییچ کیس نوشتن در کد یک حالت دیفالت قرار دارد که در آن همه متغیرها مساوی 0 ست شده اند.

```
    default: begin
    red_led = 0;
    blue_led = 0;
    green_led = 0;
    Lock = 0;
    end
```

حال برای کد یک تست بنچ نوشته و آن را تست میکنیم:

خروجی کد مشابه زیر است: