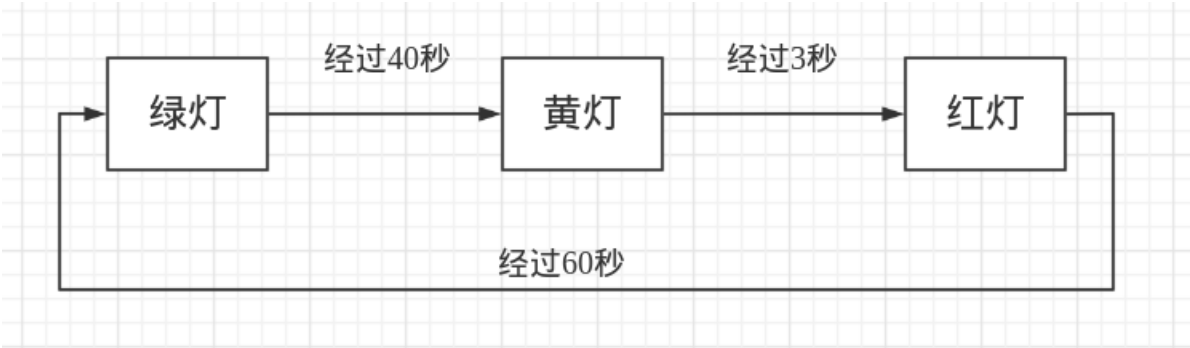
有限状态机

实例

红绿灯大家肯定不陌生，毕竟上学的路上大家都会经过红绿灯，那么，红绿灯是怎样变化的呢?

红绿灯变化示意图:



红绿灯便是一个经典的有限状态机模型。

详解

有限状态机是一种用来进行对象行为建模的工具，其作用主要是描述对象在它的生命周期内所经历的状态序列，以及如何响应来自外界的各种事件。

在上述例子之中，对应的是一个对红绿灯的行为进行建模的有限状态机。在红绿灯的生命周期里，一共有三种状态，绿灯，黄灯，红灯，这便是红绿灯要经历的状态序列，“经过T秒”便可以看成一个外部事件，从绿灯到黄灯的状态转移过程可以这么描述：“在满足‘经过40秒’的条件后，红绿灯将会执行改变灯颜色的动作，其状态会从绿灯变成黄灯”，这便是一次状态转移。

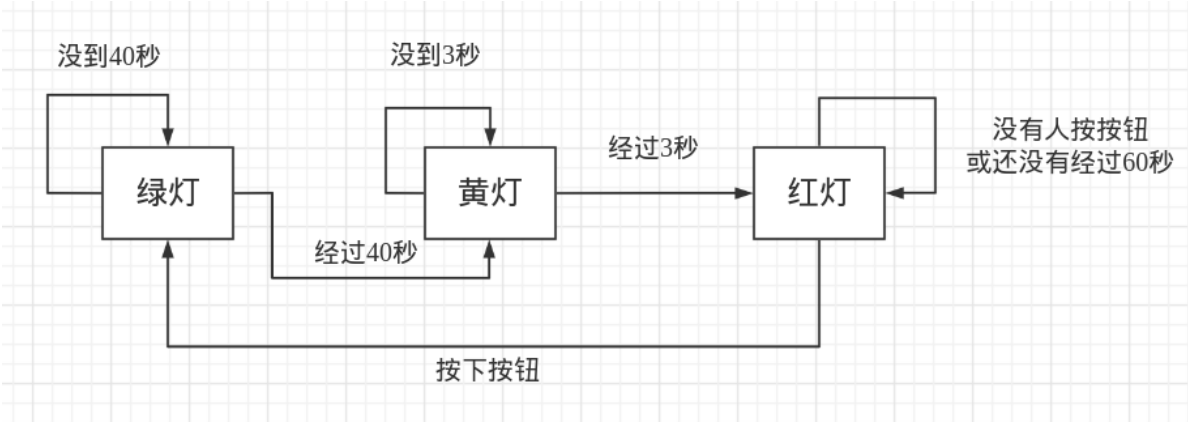
状态机可归纳为4个要素，即现态、条件、动作、次态。“现态”和“条件”是因，“动作”和“次态”是果。详解如下:

1. **现态:** 是指当前所处的状态。
2. **条件:** 又称为“事件”。当一个条件被满足，将会触发一个动作，或者执行一次状态的迁移。
3. **动作:** 条件满足后执行的动作。动作执行完毕后，可以迁移到新的状态，也可以仍旧保持原状态。动作不是必需的，当条件满足后，也可以不执行任何动作，直接迁移到新状态。
4. **次态:** 条件满足后要迁往的新状态。“次态”是相对于“现态”而言的，“次态”一旦被激活，就转变成新的“现态”了。

类比上述红绿灯的例子可以这么描述:

在满足**条件**之后，该对象将会执行对应**动作**，该动作会使得对象的状态从**现态**转移到**次态**。

当然，现态和次态可以是相等的:



红绿灯伪代码

```
logic [5:0]time;
logic [1:0]state;//00 green 01 yellow 10 red

//clk = 1Hz
always_ff @(posedge clk) begin
    if(state == 2'b00) begin
        if(time <= 6'd39) begin
            state <= state ;
            time <= time + 1 ;
        end else if(time == 6'd40) begin
            state <= 2'b01 ;
            time <= 6'd0;
        end
    end
    ......
end
```