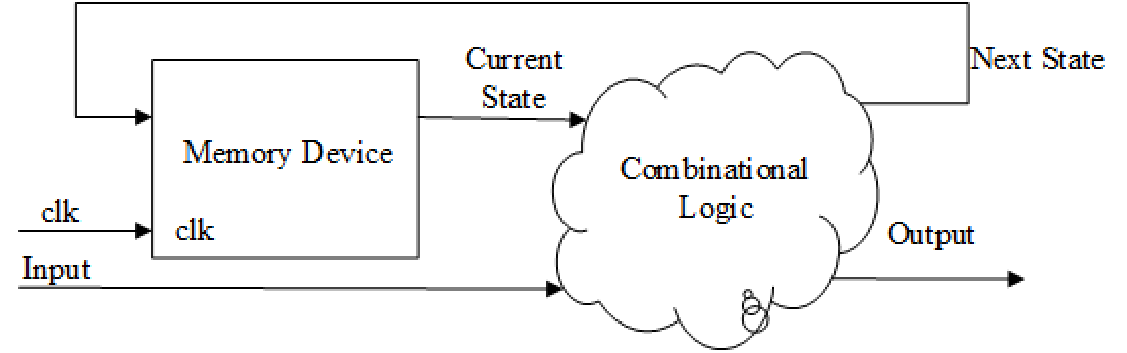
**实验12**

**有限状态机**

# **实验介绍**

我们发现组合逻辑电路有着很多的局限性：不管一个组合逻辑电路设计得有多么精妙，或者有多么复杂，归根结底都只是纯粹的逻辑运算器，这些电路只是把输入通过某种逻辑运算成了输出，而输出与电路本身没有关系。因此，我们引入了时序逻辑电路的概念。

[](https://soc.ustc.edu.cn/Digital/lab5/figs/sequential.png)

时序逻辑电路是一种具有记忆功能的数字电路，对于同一组输入，在不同的时间上会有不同的输出，也就是说输出除了与输入外，还与电路本身的状态有关。一般情况下，这种输出受到时钟信号的影响，只有在时钟信号到达时才会进行计算和输出结果。目前常见的时序逻辑电路一般由触发器、寄存器、计数器、状态机等电路元件组成。

时序逻辑的特点可以概括如下：

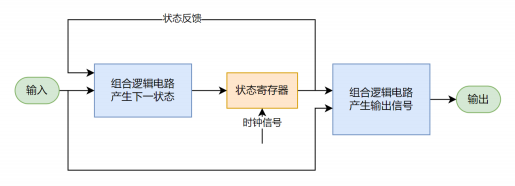
* 具有记忆功能：时序逻辑电路可以存储之前输入的信号值，实现状态的存储和切换。
* 受时钟信号控制：时序逻辑电路的输出结果受到时钟信号的控制，只有在时钟信号到达时才会计算和输出结果。
* 容易产生时序问题：由于时序逻辑电路的输出结果与时钟信号有关，时序问题可能会导致输出值的错误或延迟。

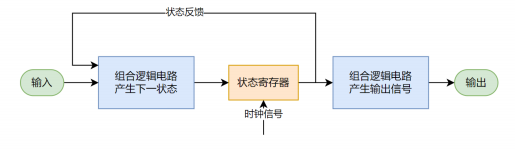
# **实验内容**

## 1 基础知识

状态机是一种能够根据输入信号和当前状态来实现状态转移和输出动作的时序逻辑电路。它能够处理复杂的时序逻辑和控制任务，广泛应用于自动控制系统、通信系统和计算机系统等领域。

一般来说，典型的时序逻辑电路都可以归结为下图所示的两种基本结构， 即便是较为复杂的电路也可以对其进行细化，最终肯定可以对应到其中一种结构上。



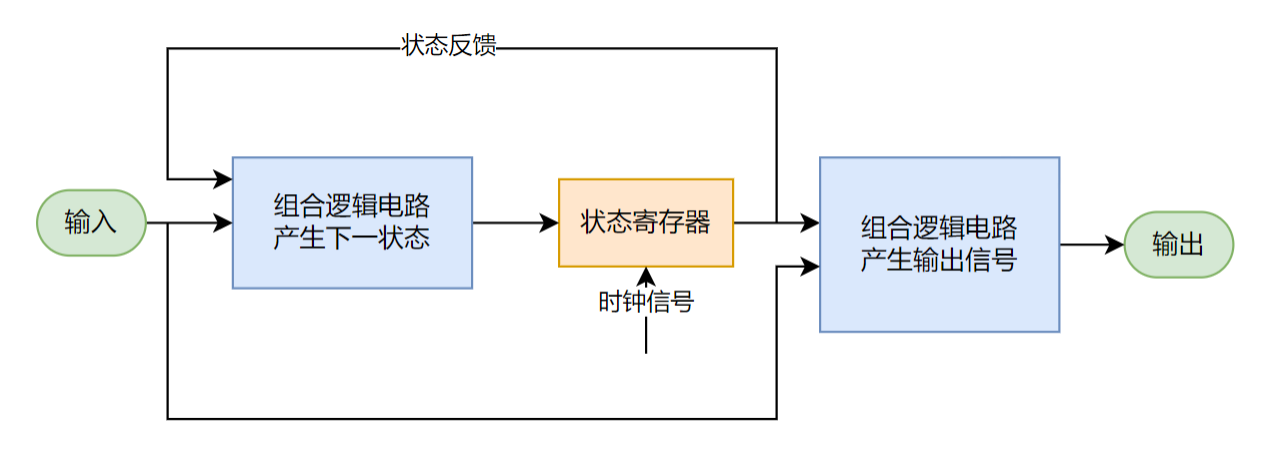


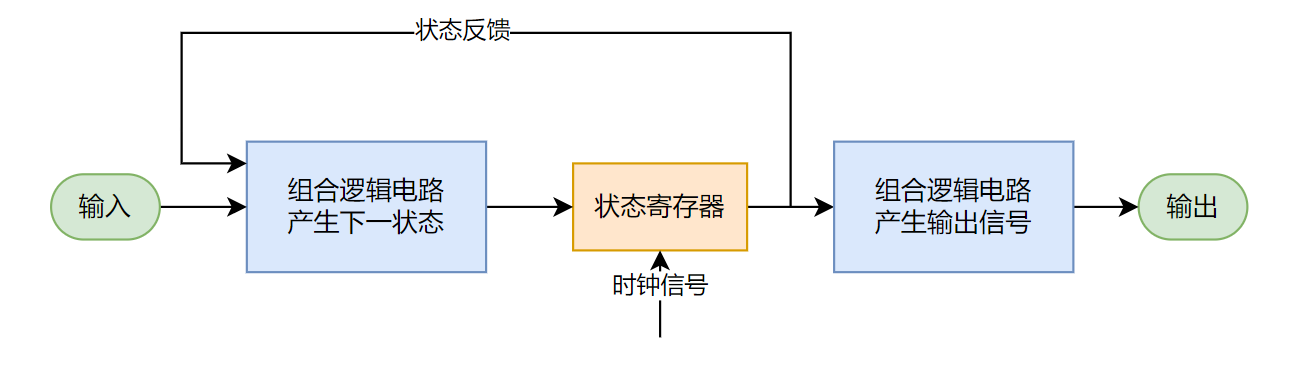
你可能已经注意到了， 电路中存在一个名为状态寄存器的特殊结构， 该结构存储了电路当前的状态信息。假定寄存器的位宽为 n，则该电路的状态数量不会超过 2n，即其状态数量是有限的， 因此这种电路结构称为有限状态机（FinateState Machine, FSM）。在《形式化方法》这门课程中，有限状态机被定义为一个五元组 (Q, Σ, δ, q0 , F)。 此处我们不再展开介绍。

## 2 Verilog实现

### ****2.1 Mealy VS Moore****

有限状态机可以根据状态转移方式和输出信号的生成方式进行分类。摩尔型（Moore）状态机的输出信号只与当前状态有关，输入信号不会直接影响到输出信号，而是与当前状态（简称现态）信号一起生成下一状态（简称次态）信号，在时钟的上升沿之后次态转换为现态，才能影响到输出。而米莉（Mealy）型状态机的输出信号由现态与输入信号共同生成，输入信号可立刻对输出信号产生影响。





在实践中我们发现，Moore 型状态机的优点是输出信号与状态直接关联，因此设计相对简单，易于理解和分析。其缺点是无法在状态之间传递信息，输出信号只能在状态转移之后更新。Mealy 型状态机的优点是可以根据输入信号的变化及时更新输出信号，具有更灵活的输出控制能力。但其缺点是相比于 Moore 型状态机，设计和分析的过程稍微复杂一些，因为输出信号的变化与输入信号紧密相关。

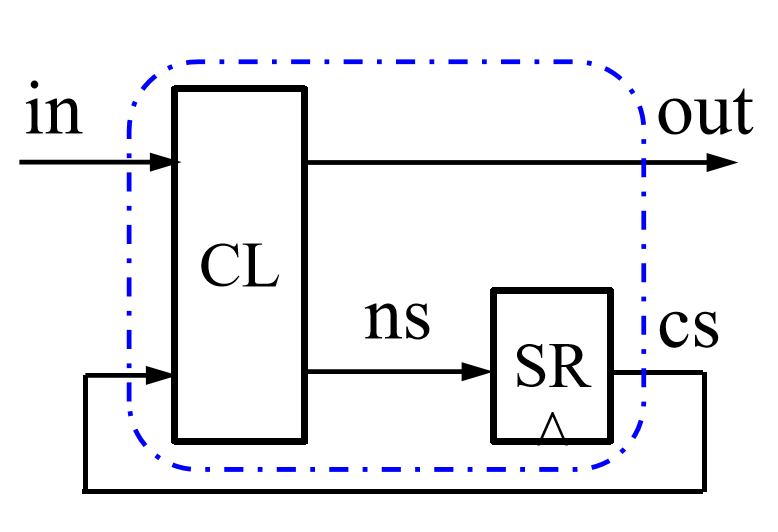
在实际应用中，选择使用哪种类型的状态机取决于具体的需求和设计要求。如果输出信号只与状态相关，且无需根据输入信号的变化而变化，则可以选择 Moore 型状态机。如果输出信号需要根据输入信号的变化及时调整，则可以选择 Mealy 型状态机。一般来说，如果对电路响应速度要求不是非常苛刻的话，我们统一推荐大家使用 Moore 型有限状态机。

### ****2.2 一段式状态机****

现在，我们将使用 Verilog 硬件描述语言实现基础的状态机模块。一般来说，我们主要分为以下三个部分进行描述：

* 状态寄存器
* 状态转移逻辑
* 输出逻辑

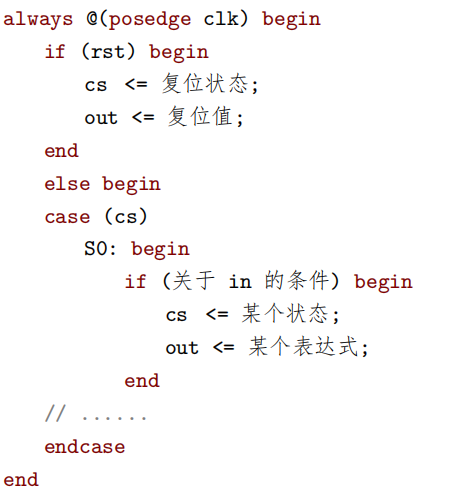
不同的描述方式也就对应了不同的电路结构与编写格式。一段式状态机就是将这三部分内容全部用一个 always 时序块进行描述，其电路图如下图所示。图中的蓝框表示将所有逻辑放在同一个 always 中进行描述。



其中

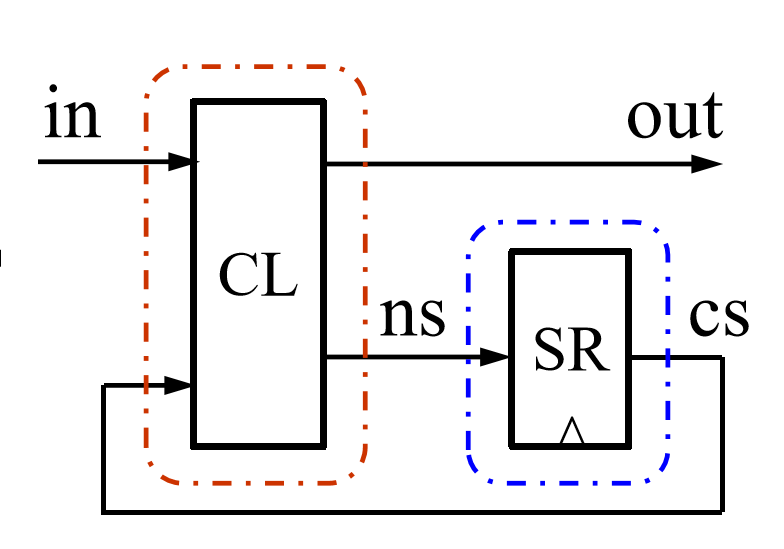
* CL（Combinational Logic）为包括输入信号 in 和当前状态 cs 的组合逻辑电路。
* SR（State Register）为存储当前状态的时序逻辑电路。

一段式状态机的模板代码如下所示：

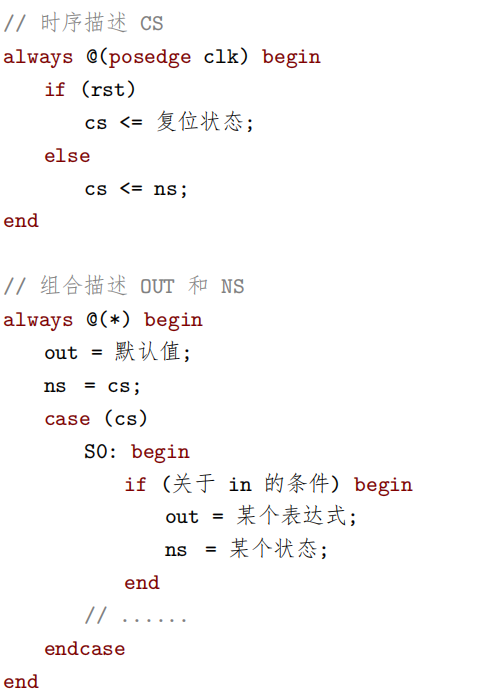


### ****2.3 两段式状态机****

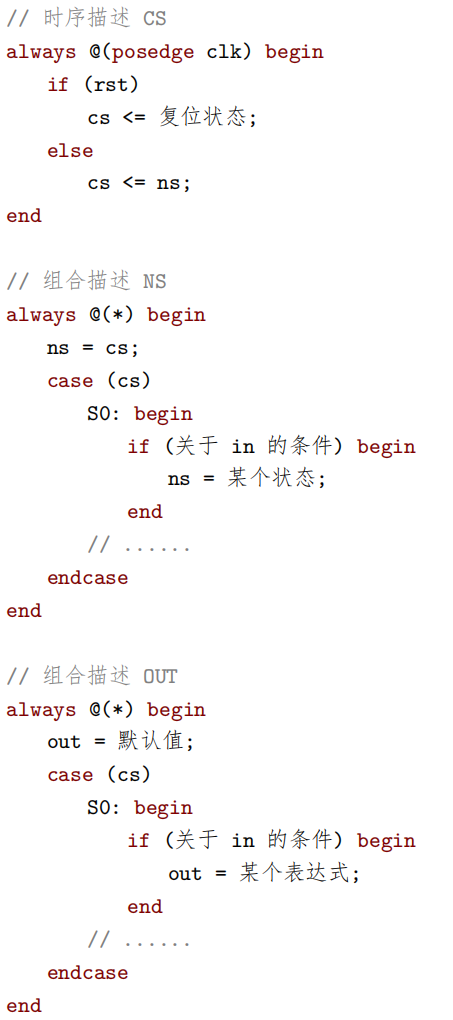
两段式状态机就是将一段式状态机的状态更新和状态转移、输出分开进行描述，其电路图如图所示。此时，图中的红框和蓝框对应着两个不同的 always 块。



两段式状态机使用当前状态 cs 决定输出信号（Mealy 型还需要输入信号 in）。由于输出信号需要经过来自组合逻辑单元 CL 的多选器，因此其组合延迟比较大。但两段式状态机同时也在很多场合是非常必要的，因为它的输出与当前状态是同步变化的。这一特性往往会有较大的应用价值。下面是两段式状态机的模板代码：



需要注意的是，两段式状态机也可以使用三个 always 来实现，但第三段的输出逻辑一定是 always @(\*) 的组合逻辑。下面展示了使用三个 always 语句描述的两段式状态机。



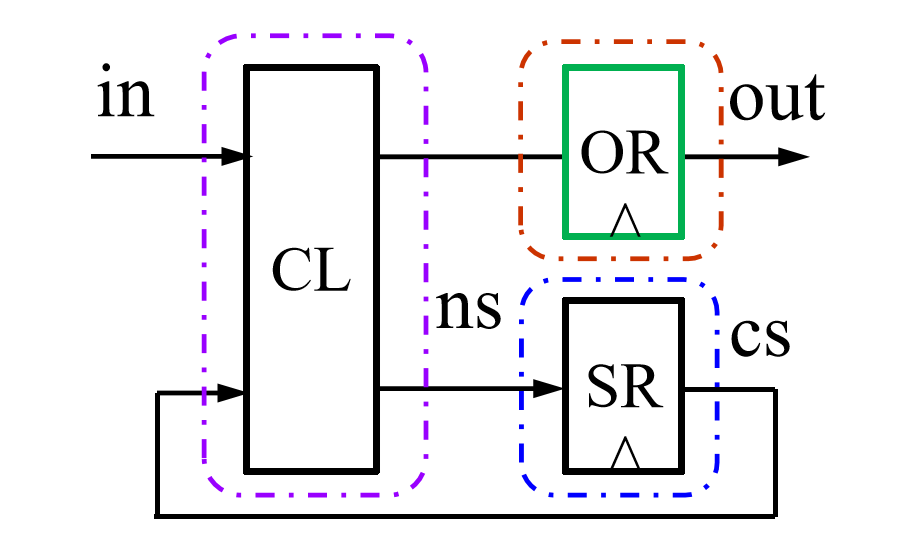
有时候，我们也会习惯性地称上面所示的状态机为「伪三段式状态机」。不难发现，无论是 Mealy 型还是 Moore 型状态机，都可以使用「伪三段式状态机」进行描述。

### ****2.4 三段式状态机****

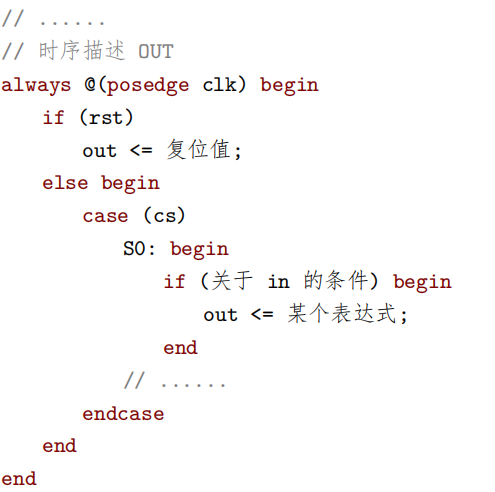
与「伪三段式状态机」不同，「真正的」三段式状态机使用寄存器进行输出，也就是说第三段的输出描述使用语句 always@(posedge clk)。在这里，我们衍生出来两种三段式状态机。

#### 2.4.1 使用CS决定输出

这种状态机的第三段输出是根据当前状态 CS 决定的，其电路图如图所示。



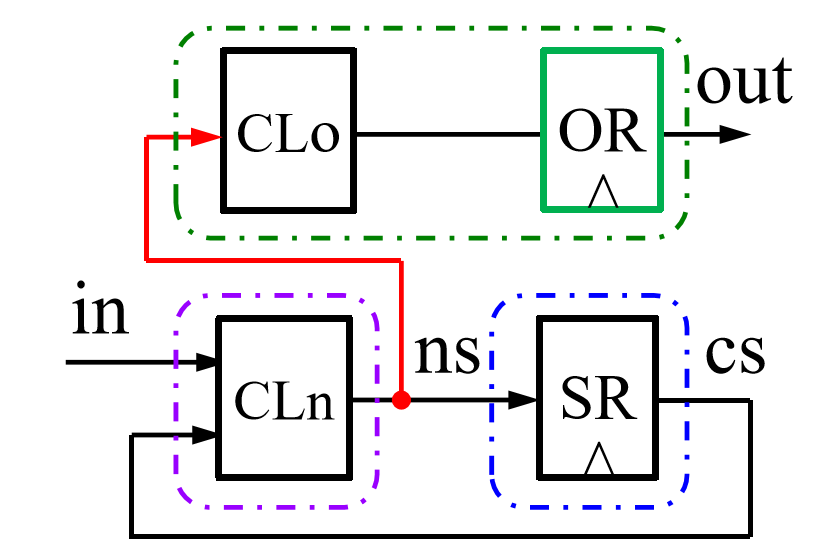
下面是这种状态机的代码模板。



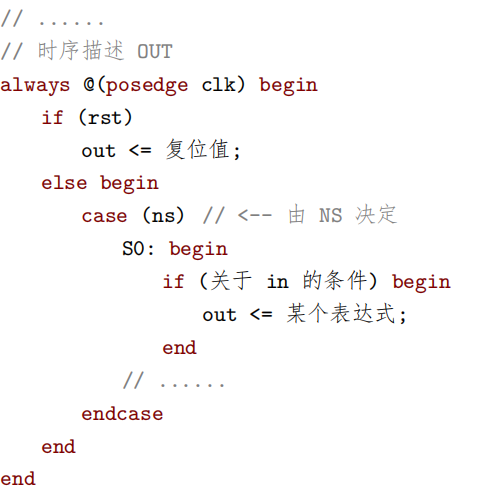
细心的同学可能已经发现了，与伪三段式状态机相比，输出 out 经过了一个额外的寄存器 OR，因此输出会比正常状态下延迟一个周期给出——这在很多情况下是完全无法接受的。很多电路要求输出信号的及时性，一个周期的差距就可能存在很多问题。

#### 2.4.2 使用NS决定输出

这种状态机的第三段输出是根据下一个状态 NS 决定的，其电路图如图所示。



其中，CLo 是决定输出的组合逻辑单元，CLn 是决定次态 NS 的组合逻辑单元。下面是这种状态机的代码模板：



现在，我们通过下一个状态来决定输出信号。如果状态机是 Moore 型状态机，那么输出不会出现任何延迟，且时序较好。但如果状态机是 Mealy 型状态机，这种设计就会出现问题：输出信号 out 无法与输入信号 in 建立联系。因此，如果一定需要使用当前状态和当前的输入决定输出，那么只能使用两段式状态机进行设计。

简而言之，我们建议大家优先选择「伪三段式状态机」完成自己的设计。因为目前的项目规模不需要考虑组合逻辑延迟对电路结果的影响。

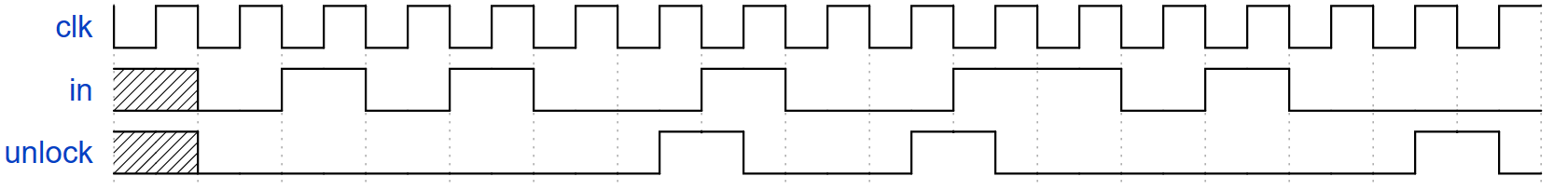
## 3 示例

为了便于理解，我们以一个例子带大家完整体验时序逻辑电路的设计流程。

某同学有一个神奇的锁。锁盘上只有两个按键，我们不妨记为 0 和 1。只有按键按照 0100 的顺序按下时才能解锁成功。例如，连续按下 01010 时并不会解锁，但再按下 0 后便会解锁（因为最近的四次输入为 0100）。我们想用一个数字电路判断给定的按键顺序能否解锁。

模块的输入包含一个时钟信号 clk 以及按下的按键编号 in。由于只有两个按键，所以我们可以根据 in 的高低电平区分按下的按键（例如高电平代表按下 1）。在 clk 的上升沿模块接收一个按键信息，同时输出一个 unlock 信号，当 unlock 信号为高电平时表明最近四次输入的序列可以解锁。

下面是一段波形图示例：

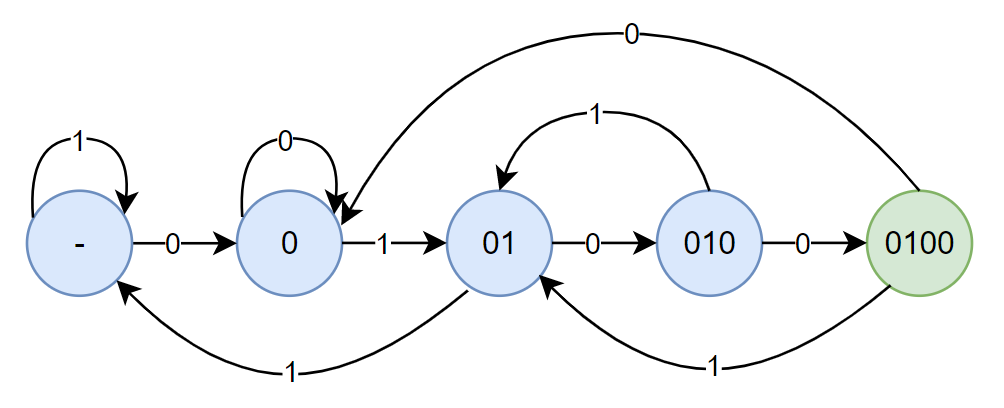
[](https://soc.ustc.edu.cn/Digital/lab5/figs/lock.png)

首先来考虑如何确定状态。自然，我们可以根据当前最近的四个输入标识状态，则对应的状态共有 16 种。但包含十六个状态的有限状态机无论设计上还是实现上都较为复杂，尽管我们可以通过状态化简消去一部分，但这个过程依然是十分繁琐的。

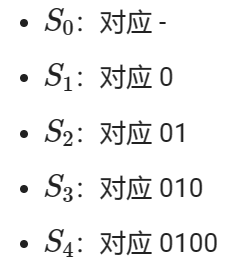
让我们再次分析这个问题。对于一个给定的输入序列，想要判断其能否开锁，我们只需要关注其最近的输入能否组成 0100 序列。先前我们固定观察最近的四次输入，但实际上有些情况近期是一定不能解锁的，例如序列 1110 至少要再经过三次输入才有可能解锁。

基于这一事实，我们可以只关注输入序列是否包含 0100 及其子序列，即考察最近的输入内容为 0、01、010、0010 四种情况。我们称之为后缀识别。

在最开始没有任何输入时，我们可以引入一个初始状态（不妨记作 - ），用于代表不属于上述四种的情况。接下来当输入一个 0 时，我们就识别到了后缀 0，即可进入下一状态；若输入一个 1，则不属于任何一种后缀，因此依然在初始状态。以此类推，我们就得到了下图所示的状态机。

[](https://soc.ustc.edu.cn/Digital/lab5/figs/state_simple.png)

[](https://soc.ustc.edu.cn/Digital/lab5/figs/state_simple_table.jpg)

于是，我们就可以确定下来，这个问题的状态机一共有五个基本状态。我们约定如下的对应关系：

初始状态为S0，仅在 S4 状态时输出解锁信号。由于输出仅和当前状态有关，因此我们可以选择 Moore 型状态机进行设计。五个状态可以使用 3bits 位宽的编码进行处理。接下来我们编写三段式 Moore 型状态机的代码。

首先定义状态变量以及状态名称：



接下来编写第一段：状态更新。假定 reset 信号的效果是清除之前所有的输入，恢复初始状态。则按下 reset 后状态机应当跳转到 S0。



接下来编写第二段：状态转移。根据状态转换图，我们可以编写相应的代码，此处不再列出。最后，我们编写第三段：输出。



这样就完成了模块的编写。

# 思考与练习

1. 请完成一个交通信号灯的编写，并附加功能：绿灯保持 35 个时钟周期，黄灯保持 5 个时钟周期，红灯保持 40 个时钟周期。要求使用三段式结构描述状态机，自行编写仿真文件进行测试。代码框架如下：

