状态机

同步状态机

● 同步状态机的结构

- 由状态寄存器（触发器）作为状态记忆部件（常用正跳边沿触发的D触发器）
- 仅当触发信号到达时刻才可能发生状态改变
- n个触发器最多有2^n 个状态
- 两种同步状态机：

Mealy型----- 输出是当前状态和输入的函数
次态是当前状态和输入的函数

Moore型 -----输出是当前状态的函数
次态是当前状态和输入的函数

● 同步状态机的实现

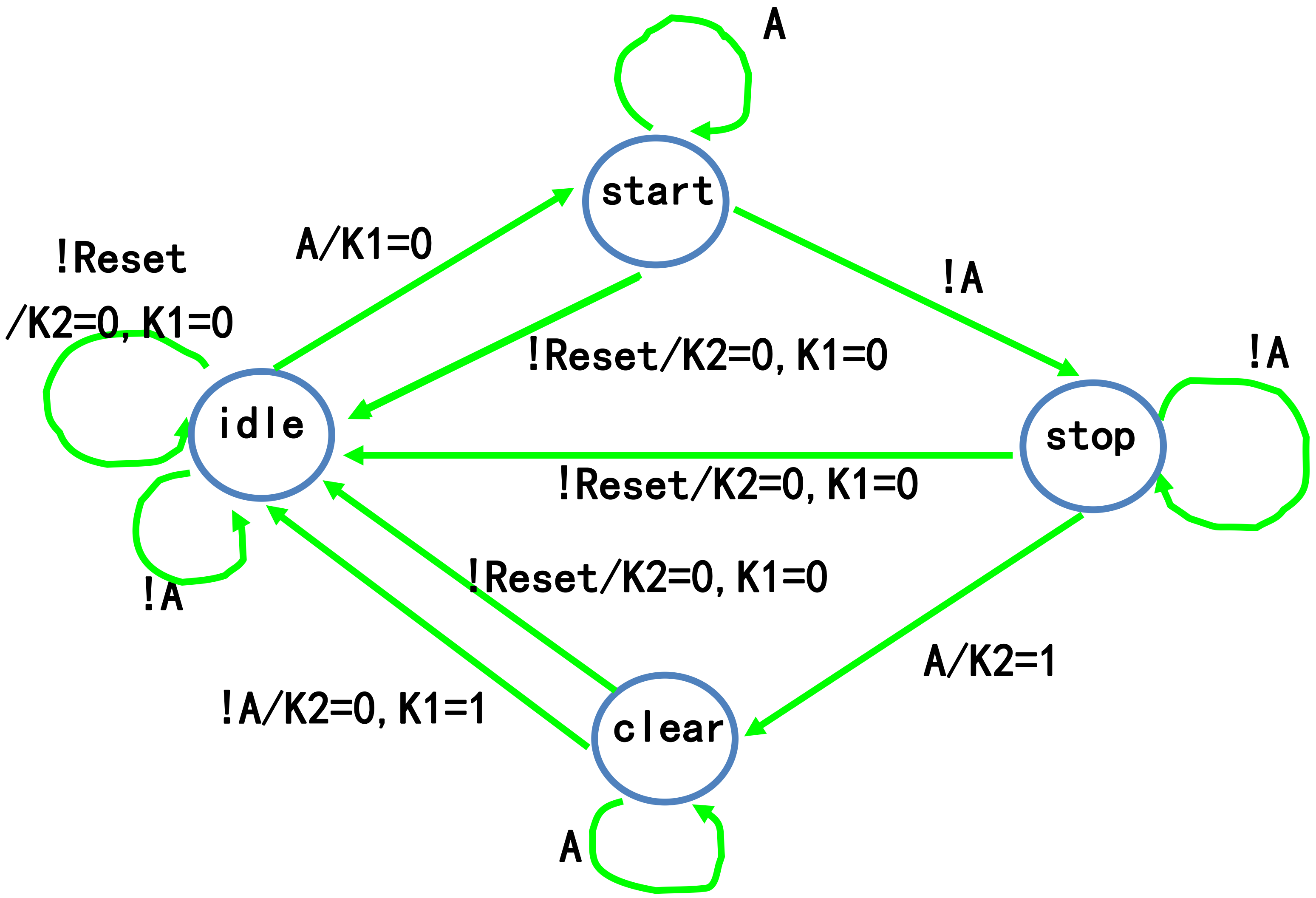
状态分配：

- 二进制码表示状态的状态机
- 格雷（Gray）码表示状态的状态机
- 独热（One-hot）码表示状态的状态

有限状态机的描述风格：

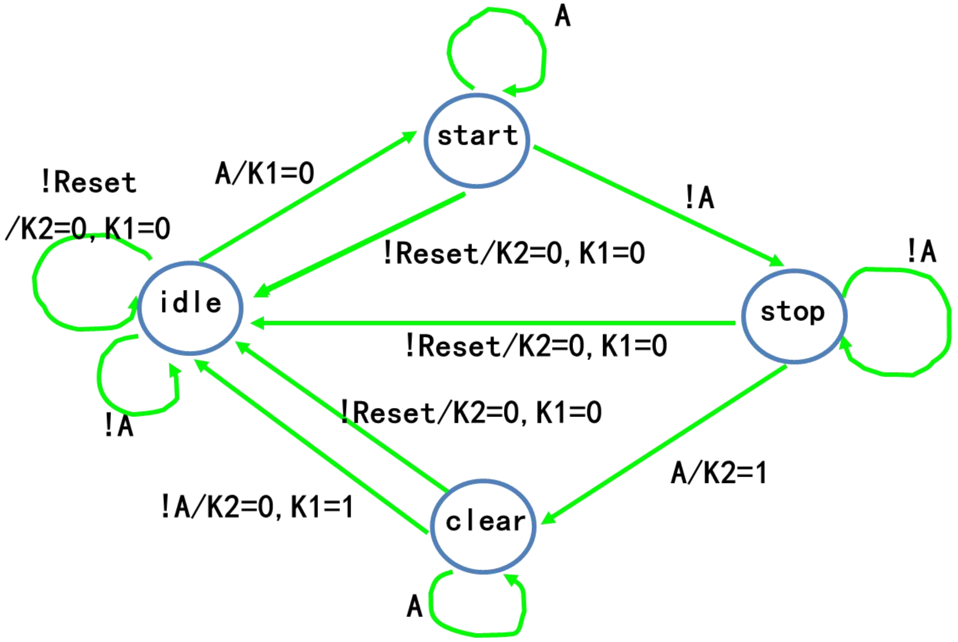
- One always风格
- Two always风格
- Three always风格

设计举例

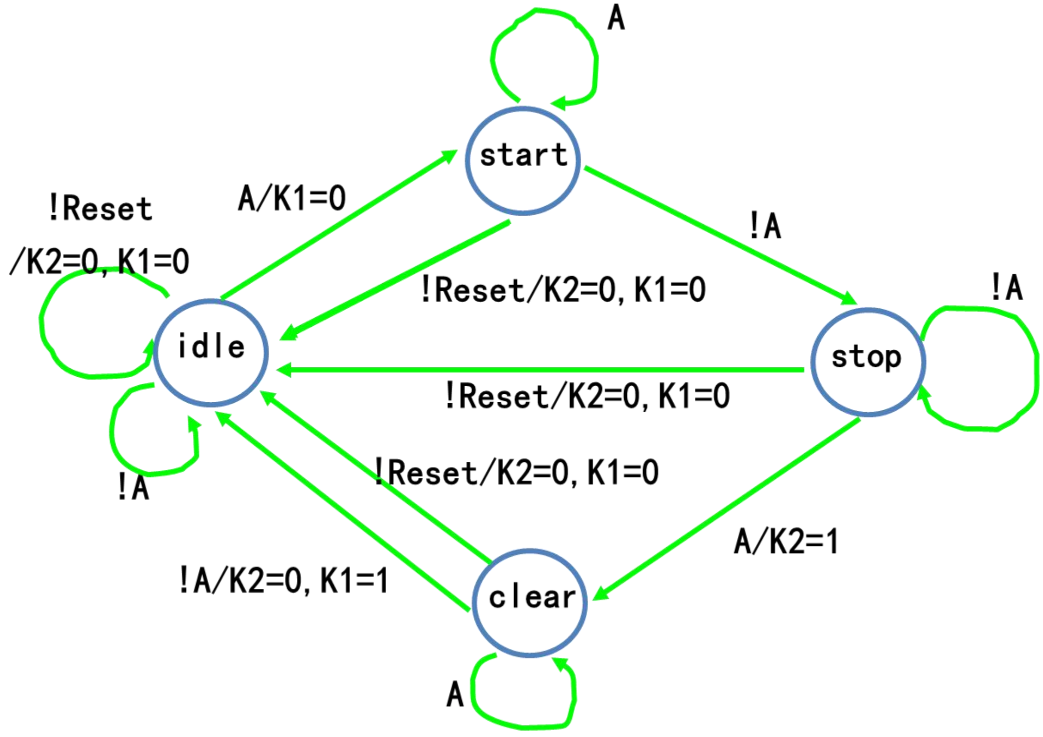

A
start
A/K1=0
!Reset
/K2=0, K1=0
!A
!Reset/K2=0, K1=0
!A
idle
stop
!Reset/K2=0, K1=0
!A
!Reset/K2=0, K1=0
A/K2=1
!A/K2=0, K1=1
clear
A

用可综合Verilog模块设计状态机的典型方法

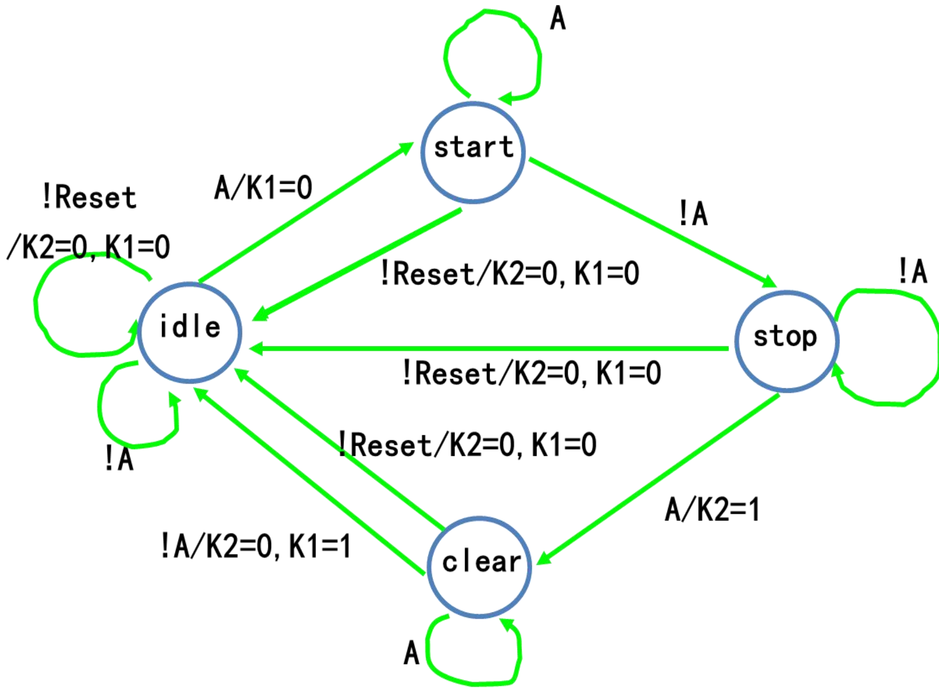
```
module fsm(clock,reset,a,k2,k1);
input clock,reset,a;
output k2,k1;
reg k2,k1;
reg[1:0] state;
parameter idle=2'b00,start=2'b01,stop=2'b10,clear=2'b11;
always @(posedge clock)
if(!reset)
begin
    state<=idle;k2<=0;k1<=0;
end
else
```



```
case(state)
    idle:begin
        if(a)
        begin
                state<=start;
                k1<=0;
        end
        else
                state<=idle;
      end
   start:begin
        if(!a)  state<=stop;
        else    state<=start;
       end
```



```
stop:begin
        if(a)
        begin
                state<=clear;
                k2<=1;
        end
        else    state<=stop;
     end
clear:begin
        if(!a)
        begin
                state<=idle;
                k2<=0;k1<=1;
        end
        else    state<=clear;
      end
   endcase
endmodule
```



用可综合Verilog模块设计、用独热码表示状态的状态机

```
module fsm(clock,reset,a,k2,k1);
input clock,reset,a;
output k2,k1;
reg k2,k1;
reg[3:0] state;

parameter
    idle=4'b1000,start=4'b0100,stop=4'b0010,clear=4'b0001;
always @(posedge clock)
if(!reset)
begin
    state<=idle;k2<=0;k1<=0;
end
else
```

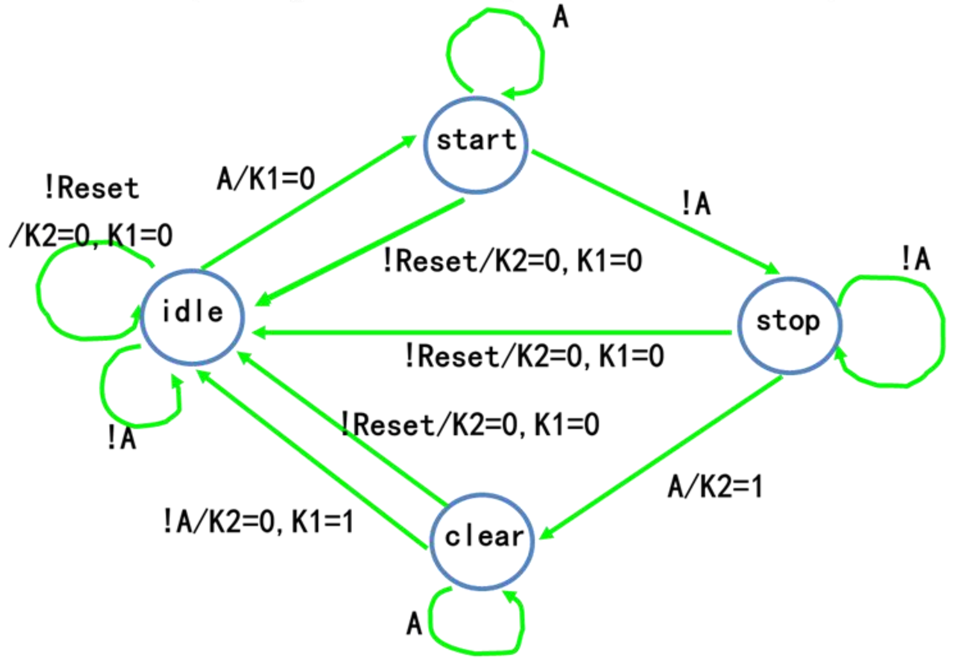
```
case(state)
    idle:begin
           if(a)
           begin
                      state<=start;
                      k1<=0;
           end
           else
                      state<=idle;
           end
    start:begin
           if(!a)  state<=stop;
           else    state<=start;
         end
```

```
stop:begin
         if(a)
         begin
                  state<=clear;
                  k2<=1;
         end
         else     state<=stop;
       end
   clear:begin
         if(!a)
         begin
                  state<=idle;
                  k2<=0;k1<=1;
         end
         else     state<=clear;
        end
   default:state<=idle;
   endcase
endmodule
```

用可综合Verilog模块设计的多输出状态机时常用的方法

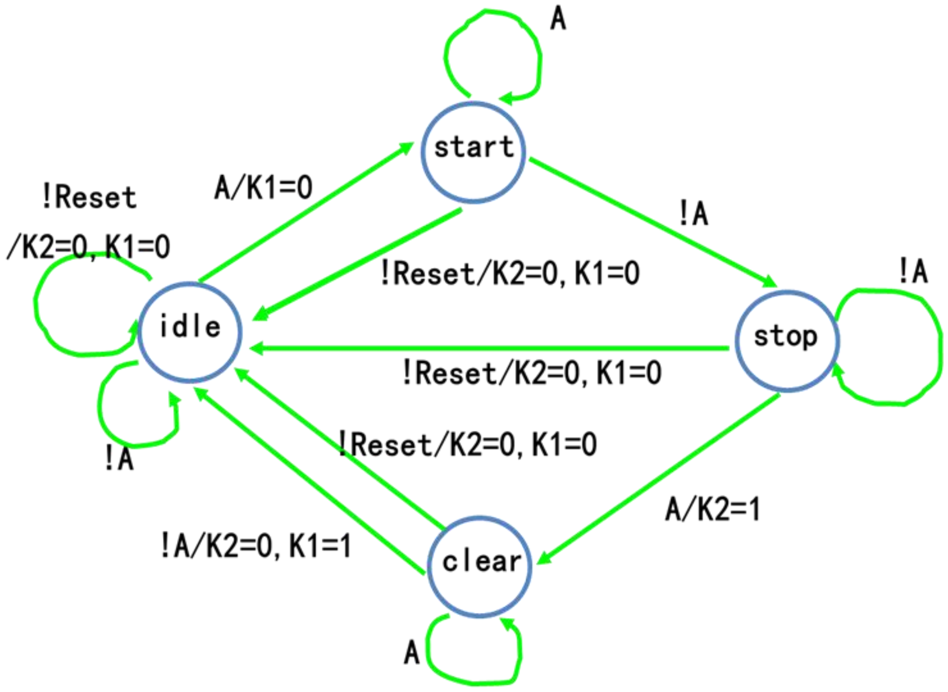
```
module fsm(clock,reset,a,k2,k1);
input clock,reset,a;
output k2,k1;
reg k2,k1;
reg[1:0] state;

parameter idle=2'b00,start=2'b01,stop=2'b10,clear=2'b11;
always @(posedge clock)
if(!reset)
begin
        state<=idle;
end
else
```

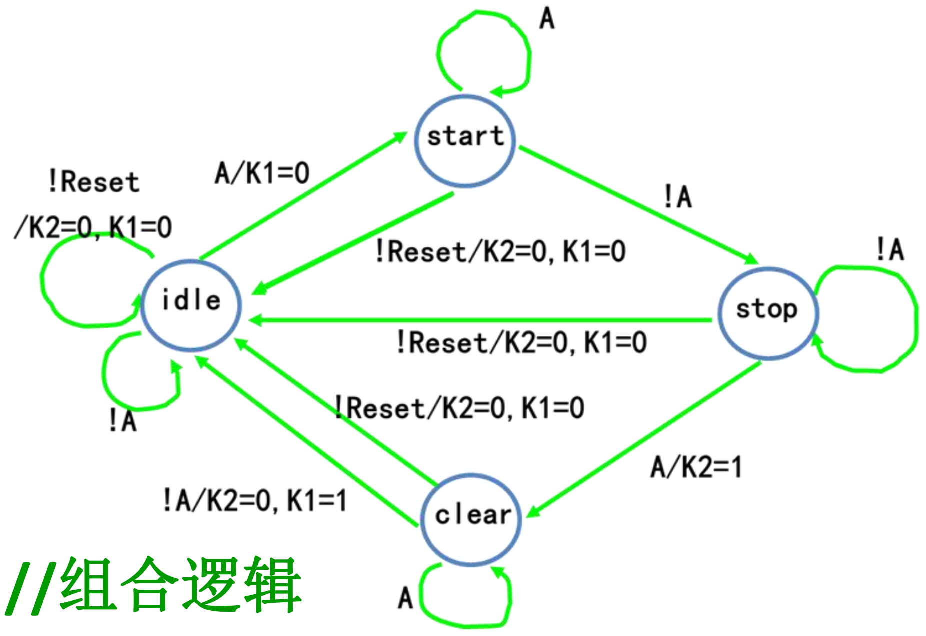


//状态机

```
case(state)
     idle:begin
             if(a)
             begin
                          state<=start;
             end
             else
                          state<=idle;
         end
     start:begin
             if(!a)  state<=stop;
             else    state<=start;
          end
   stop:begin
             if(a)
             begin
                          state<=clear;
             end
             else         state<=stop;
         end
```



```
clear:begin
          if(!a)
          begin
                    state<=idle;
          end
          else      state<=clear;
        end
default:state<=2'bxx;
endcase
always @(state or reset or a)   //组合逻辑
if(!reset) k2=0;
else if((state==stop)&&a)
      k2=1;
      else k2=0;
always @(state or reset or a)       //组合逻辑
if(!reset) k1=0;
else if((state==clear)&&!a)
      k1=1;
      else k1=0;
endmodule
```


A
start
A/K1=0
!Reset
/K2=0, K1=0
!A
!Reset/K2=0, K1=0
!A
idle
stop
!Reset/K2=0, K1=0
!A
!Reset/K2=0, K1=0
A/K2=1
!A/K2=0, K1=1
clear
A