## 实验五 数据通路

### 5.1实验目的

1.理解数据通路的组成结构；

2.熟悉ALU、移位寄存器、通用寄存器组的功能和设计方法；

3.掌握运算器的工作原理和信息传递的控制过程；

4.掌握双倍字长加减运算的方法。

### 5.2实验原理

运算器数据通路实验原理如图 5.1。图中没有标出CLK和RESET信号。CLK 信号是移位寄存器 SHIFTER、标志寄存器 PSW、通用寄存器组 GRS、暂存器 A 的时钟输入；RESET信号是PSW和暂存器 A 的复位输入。需要说明的是RESET信号由KEY0提供，从前面实验知道，KEY0按键按一次，产生的是一个负脉冲；而实验电路设计中的控制信号统一以高电平有效，所以将KEY0信号取反以后作为RESET信号。



图 5.1 运算器数据通路

#### ALU

本实验的ALU在上一个实验加减运算电路的基础上， 扩充了逻辑运算，共10种算术逻辑运算；由于实验板的开关数量有限，对运算控制信号进行了编码，表 5.1给出了编码与运算功能的对应关系。

表 5.1 ALU控制信号编码与运算功能

|  |  |
| --- | --- |
| ALU\_OP | ALU运算 |
| 0000 | 传送 |
| 0001 | ADD |
| 0010 | ADDC |
| 0011 | SUB |
| 0100 | SUBB |
| 0101 | AND |
| 0110 | OR |
| 0111 | NOT |
| 1000 | XOR |
| 1001 | INC |
| 1010 | DEC |

#### 移位寄存器

移位寄存器具有左移、右移和并行输入功能。在时钟上升沿到来时，根据功能选择信号进行左移、右移、保存操作。和实验一的移位寄存器基本相同，不同的是对移位控制信号进行了编码（见表 5.2），以减少对实验板输入开关数量的要求。

表 5.2 移位寄存器控制信号编码与操作

|  |  |
| --- | --- |
| ST\_OP | 移位寄存器操作 |
| 00 | 不变 |
| 01 | 右移 |
| 10 | 左移 |
| 11 | 装入 |

#### 通用寄存器组GRS

通用寄存器组GRS（General Register Set）包含4个4位的通用寄存器R0~R3，用来存放参加运算的操作数和运算结果。

#### 顶层模块

PSW为状态标记寄存器；A暂存器也和一般寄存器相同。寄存器向总线输出需要增加三态缓冲，在前面实验中已经使用过三态缓冲器。在顶层模块，实例化各个模块的实例，并将各个模块连接起来。

### 4.3 预习要求

1）认真阅读、理解实验原理。

2）通过理论分析用铅笔填写实验操作和记录表中空白的栏目，以便实验时与实际结果对比。

### 4.4实验操作和记录

#### 1.DATA→R1，DATA→R2

R1和R2寄存器中的内容在之后的实验操作中，均延用此步骤中存入的数值。为保证结果，后续表格操作前可先确认R1和R2寄存器中的值，若因为误操作发生改变，重新执行此表格，重置R1和R2寄存器的值。

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | DATA | DATAoe | INDEX | GRSce | GRSoe | Soe | CLK | GRS\_Q | BUS | 实验现象分析 |
| ① | 1001 | 1 | 01 | 1 | 0 | 0 |  | 9 | 9 | #1001→R1 |
| ② | 1100 | 1 | 10 | 1 | 0 | 0 |  | C | C | #1100→R2 |
| ③ | — | 0 | 01 | 0 | 0 | 0 | 0 | 9 | — | 确认R1内容 |
| ④ | — | 0 | 10 | 0 | 0 | 0 | 0 | C | — | 确认R2内容 |

#### 2.（R1）+（R2）→ R3

执行下表操作前用RESET对PSW清零（同时也复位A暂存器和移位寄存器，对寄存器组不产生影响）；DATAoe=0，使DATA开关输入不影响总线状态。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | INDEX | GRSce | GRSoe | Ace | ALU\_OP | PSWce | ST\_OP | Soe | CLK | A | BUS | S\_Q | PSW | GRS\_Q |
| ① | 01 | 0 | 1 | 1 | —— | 0 | —— | 0 |  | 9 | 9 | —— | —— | 9 |
| ② | 10 | 0 | 1 | 0 | 0001 | 1 | 11 | 0 |  | 9 | C | 5 | 0011 | C |
| ③ | 11 | 1 | 0 | 0 | —— | 0 | 00 | 1 |  | —— | 5 | 5 | —— | 5 |

写出表中每一行所完成的操作：

1. R1→A；
2. A+R2→ALU.F,ALU.F→SHIFTER（相加）;
3. SHIFTER→R3;

运算完成后读出R3寄存器的值为\_\_\_5\_\_\_\_\_\_\_，运算结果\_\_\_不正确\_\_（正确／不正确）。

实验现象分析：在①中BUS的值是由\_\_\_\_R1\_\_\_\_\_输出，②中BUS值是由\_\_\_\_R2\_\_\_\_\_输出，③中BUS值是由\_\_\_\_\_SHIFTER\_\_\_\_输出。

仿照上述步骤，验证其他算术、逻辑运算。

R0 = 9, R1 = C

**异或：**

#### （R0）XOR（R1）→ R3

1. R0→A；
2. A XOR R2→ALU.F,ALU.F→SHIFTER（异或）
3. SHIFTER→R3

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | INDEX | GRSce | GRSoe | Ace | ALU\_OP | PSWce | ST\_OP | Soe | CLK | A | BUS | S\_Q | PSW | GRS\_Q |
| ① | 00 | 0 | 1 | 1 | —— | 0 | —— | 0 |  | 9 | 9 | —— | —— | 9 |
| ② | 10 | 0 | 1 | 0 | 1000 | 0 | 11 | 0 |  | 9 | C | 5 | —— | C |
| ③ | 11 | 1 | 0 | 0 | —— | 0 | —— | 1 |  | —— | 5 | 5 | —— | 5 |

#### 自加：

#### INC（R0）→ R3

① R0→A；

② A→ALU.F，ALU.F→SHIFTER（自加）；

③ SHIFTER→R3

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | INDEX | GRSce | GRSoe | Ace | ALU\_OP | PSWce | ST\_OP | Soe | CLK | A | BUS | S\_Q | PSW | GRS\_Q |
| ① | 00 | 0 | 1 | 1 | —— | 0 | —— | 0 |  | 9 | 9 | —— | —— | 9 |
| ② | 00 | 0 | 1 | 0 | 1001 | 0 | 11 | 0 |  | 9 | 0 | A | —— | 9 |
| ③ | 11 | 1 | 0 | 0 | —— | 0 | 00 | 1 |  | —— | A | A | —— | A |

#### **非：**

#### NOT（R0）→ R3

① R0→A；

② A→ALU.F，ALU.F→SHIFTER（取反）；

③ SHIFTER→R3

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | INDEX | GRSce | GRSoe | Ace | ALU\_OP | PSWce | ST\_OP | Soe | CLK | A | BUS | S\_Q | PSW | GRS\_Q |
| ① | 00 | 0 | 1 | 1 | —— | 0 | —— | 0 |  | 9 | 9 | —— | —— | 9 |
| ② | 00 | 0 | 0 | 0 | 0111 | 0 | 11 | 0 |  | 9 | 0 | 6 | —— | 9 |
| ③ | 11 | 1 | 0 | 0 | —— | 0 | 00 | 1 |  | —— | 6 | 6 | —— | 6 |

#### 3.（R1）→ R3

将R1的内容送到R3，仍然要经过ALU，并不能在寄存器组内部完成传送。操作步骤如下：

① R1→A；

② A→ALU.F，ALU.F→SHIFTER（保存）；

③ SHIFTER→R3

根据上述操作步骤所完成的操作，填写下表的输入和控制信号，记录结果。操作前设置DATAoe=0，使DATA开关输入不影响总线状态。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | INDEX | GRSce | GRSoe | Ace | ALU\_OP | PSWce | ST\_OP | Soe | CLK | A | BUS | S\_Q | GRS\_Q |
| ① | 01 | 0 | 1 | 1 | —— | —— | —— | 0 |  | 9 | 9 | —— | 9 |
| ② | —— | 0 | 0 | 0 | 0000 | 0 | 11 | 0 |  | 9 | 0 | 9 | —— |
| ③ | 11 | 1 | 0 | 0 | —— | 0 | 00 | 1 |  | —— | 9 | 9 | 9 |

#### 4.（R1）/ 2 → R3

操作步骤如下：

① R1→A；

② A→ALU.F，ALU.F→SHIFTER（右移）；

③ SHIFTER→R3

根据上述操作步骤所完成的操作，填写下表的输入和控制信号，记录结果。操作前设置DATAoe=0，使DATA开关输入不影响总线状态。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | INDEX | GRSce | GRSoe | Ace | ALU\_OP | PSWce | ST\_OP | Soe | CLK | A | BUS | S\_Q | GRS\_Q |
| ① | 01 | 0 | 1 | 1 | —— | —— | —— | 0 |  | 9 | 9 | —— | 9 |
| ② | —— | 0 | 0 | 0 | 0000 | 0 | 01 | 0 |  | 9 | 9 | 4 | —— |
| ③ | 11 | 1 | 0 | 0 | —— | 0 | 00 | 1 |  | —— | 4 | 4 | 4 |

仿照上述步骤，验证左移操作。

**左移：**

#### （R1）\* 2 → R3

操作步骤如下：

① R1→A；

② A→ALU.F，ALU.F→SHIFTER（左移）；

③ SHIFTER→R3

根据上述操作步骤所完成的操作，填写下表的输入和控制信号，记录结果。操作前设置DATAoe=0，使DATA开关输入不影响总线状态。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | INDEX | GRSce | GRSoe | Ace | ALU\_OP | PSWce | ST\_OP | Soe | CLK | A | BUS | S\_Q | GRS\_Q |
| ① | 01 | 0 | 1 | 1 | —— | —— | —— | 0 |  | 9 | 9 | —— | 9 |
| ② | —— | 0 | 0 | 0 | 0000 | 0 | 10 | 0 |  | 9 | 9 | 2 | —— |
| ③ | 11 | 1 | 0 | 0 | —— | 0 | 00 | 1 |  | —— | 2 | 2 | 2 |

#### 5.双倍字长的加法

实验设计的加法器字长为4位，如果要做8位的加法，就要分两次进行；低4位用ADD指令相加，并且将进位情况保存在PSW中；高4位相加时，要考虑低4位的运算是否产生了进位，因此要用ADDC指令。

假设两个运算数X = 00011010，Y = 10000111；首先参照步骤1将运算数Y送入R0、R1，R0存放Y的低4位，R1存放Y的高4位；然后进行00011010＋(R1)(R0) → R3 R2，将操作过程记入下表，操作前用RESET对PSW清零。由于宽度限制，表格中的ALU即ALU\_OP，ST即ST\_OP。

① #1010→A；

② R0→ALU.SRC，ADD，置PSW，ALU.F→SHIFTER（保存）；

③ SHIFTER→R2

④ #0001→A；

⑤ R1→ALU.SRC，ADD，置PSW，ALU.F→SHIFTER（保存）；

⑥ SHIFTER→R3。

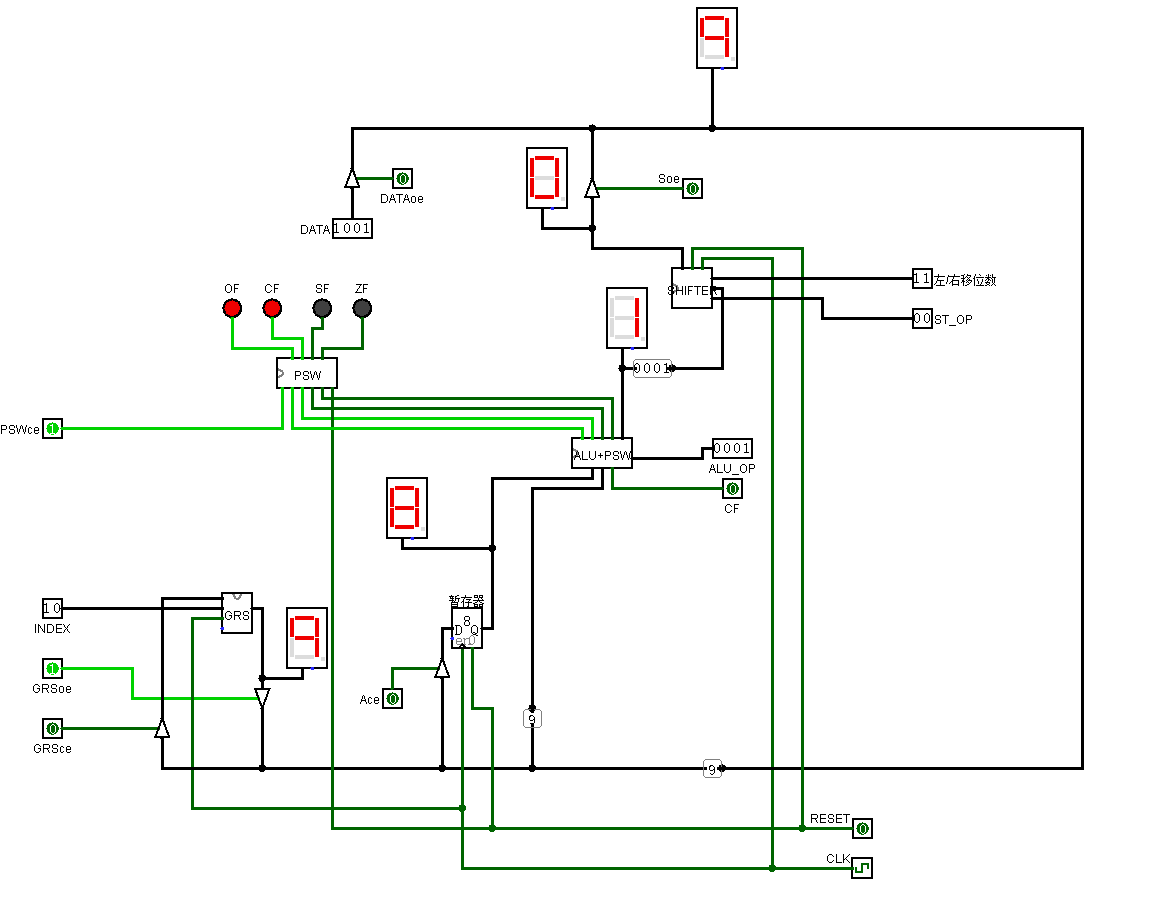
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | DATA | DATAoe | INDEX | GRSce | GRSoe | Ace | ALU | PSWce | ST | Soe | CLK | A | BUS | S\_Q | PSW | GRS\_Q |
| ① | 1010 | 1 | 00 | 0 | 0 | 1 | 0000 | 0 | 00 | 0 |  | A | A | 0 | —— | —— |
| ② | —— | —— | 00 | 0 | 1 | 0 | 0001 | 1 | 11 | 0 |  | A | 7 | 1 | 0001 | 7 |
| ③ | —— | —— | 10 | 1 | 0 | 0 | —— | —— | 00 | 1 |  | A | 1 | 1 | —— | 1 |
| ④ | 0001 | 1 | 00 | 0 | 0 | 1 | 0000 | 0 | 00 | 0 |  | 1 | 1 | 1 | —— | —— |
| ⑤ | —— | —— | 01 | 0 | 1 | 0 | 0010 | 1 | 11 | 0 |  | 1 | 8 | A | 1000 | 8 |
| ⑥ | —— | —— | 11 | 1 | 0 | 0 | —— | —— | 00 | 1 |  | 1 | A | A | —— | A |

C0输入选择的CF输入来自PSW的\_\_\_\_CF\_\_\_\_\_\_\_输出。ADDC运算的C0是\_\_\_\_CF\_\_\_\_\_（0／1／CF），ADD运算的C0是\_\_\_\_0\_\_\_\_\_\_\_ （0／1／CF）。

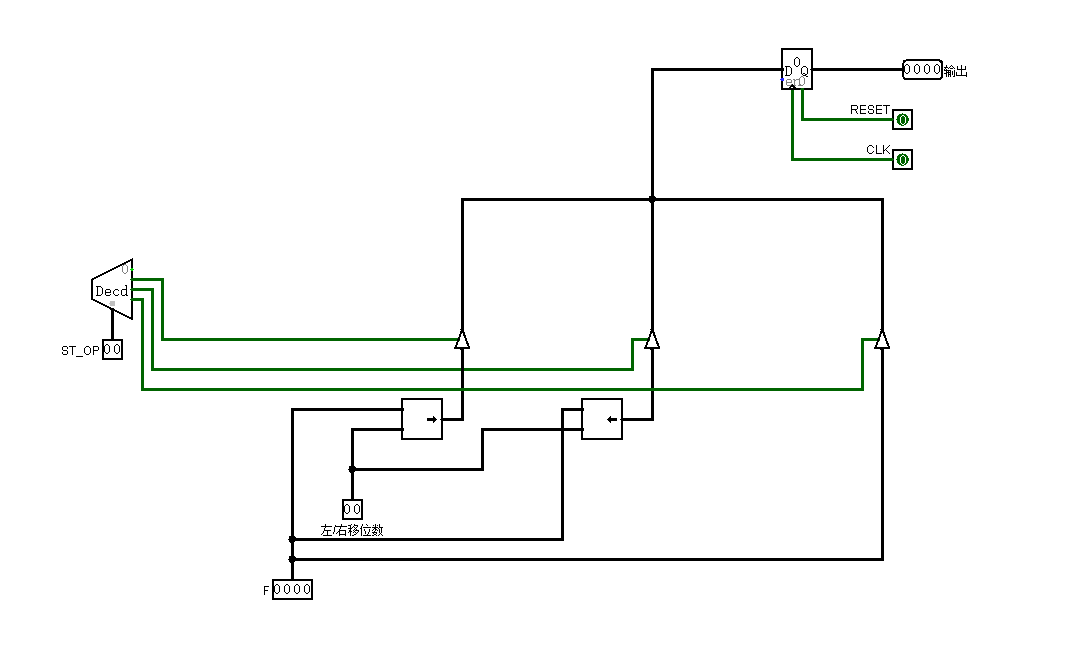
**提高实验：（根据难度选做）**

用Logisim设计上述数据通路。

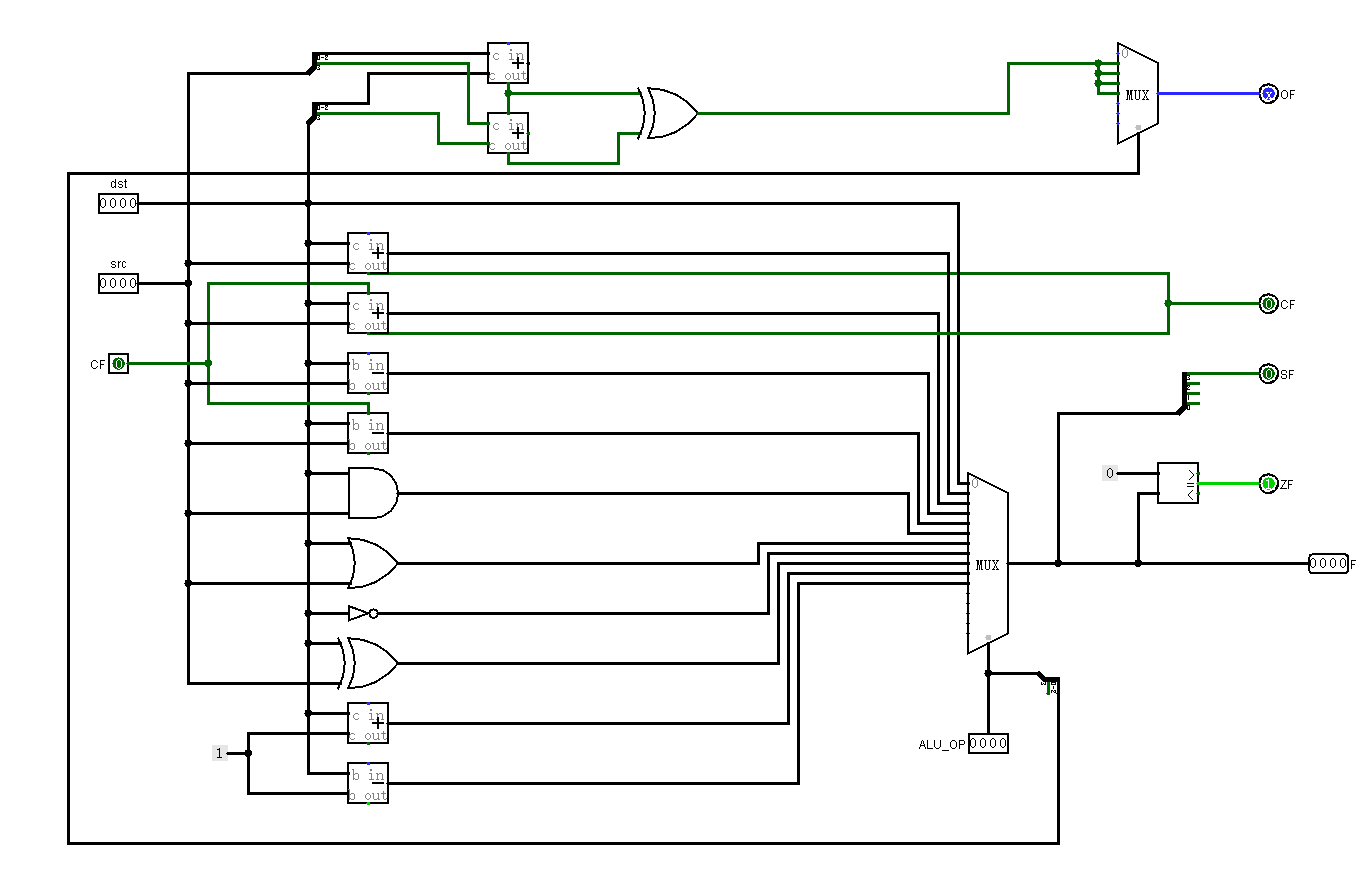
**顶层：**



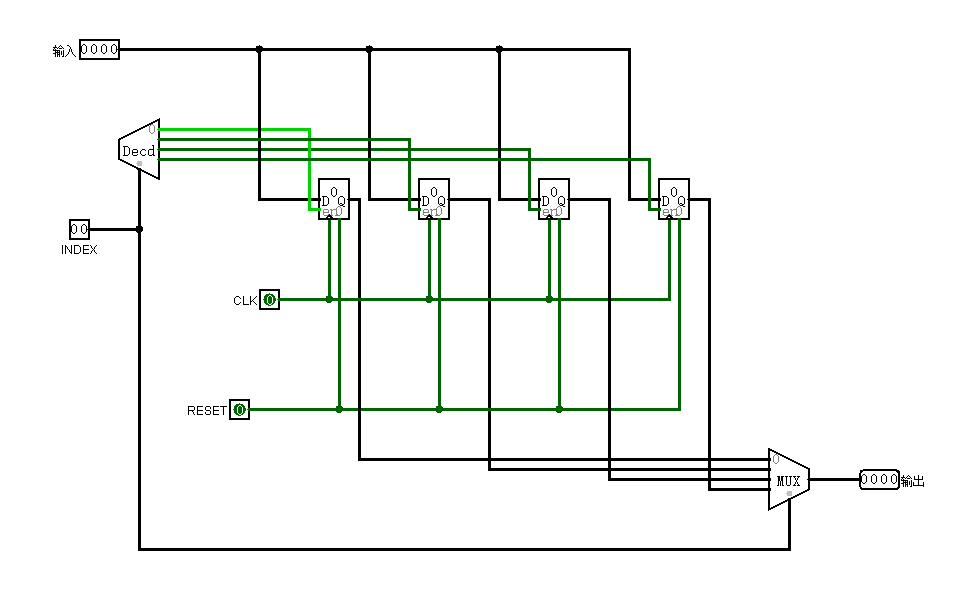
**SHIFTER：**



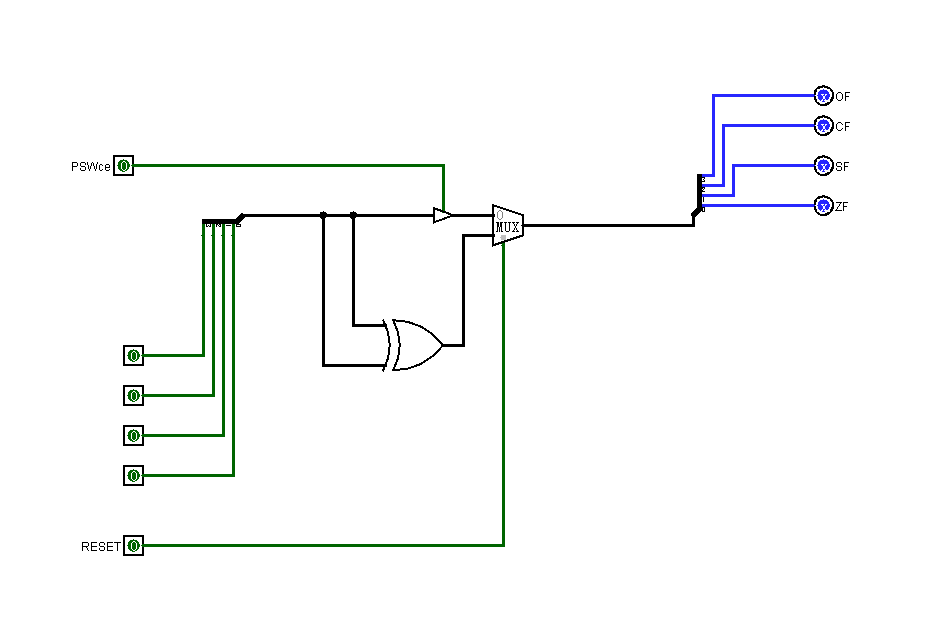
**ALU：**



**GRS：**



**PSW：**



总结：所有单独模块中的时钟信号都不可使用，都必须使用。如果在单独模块中使用，则顶层调用的单独模块里所有使用处都不会在顶层中给出引脚。解决方法：在单独模块中使用，在顶层中在改为即可。