浙江大学

本科实验报告

课程名称：	数字逻辑设计
姓　　名：	沈一芃
学　　院：	计算机学院
系：	计算机系
专　　业：	计算机科学与技术
学　　号：	3220101827
指导教师：	马德

2023 年 12 月 11 日

浙江大学实验报告

课程名称： 数字逻辑设计 实验类型： ISE 电路设计

实验项目名称： 寄存器和寄存器传输设计

学生姓名： 沈一芃 专业： 计算机科学与技术 学号： 322010827

同组学生姓名： 无 指导老师： 马德

实验地点： 东 4 509 实验日期： 2023 年 12 月 11 日

一、 实验目的和要求：

- 掌握寄存器传输电路的工作原理
- 掌握寄存器传输电路的设计方法
- 掌握 ALU 和寄存器传输电路的综合应用

二、 实验内容和原理

1. 实验内容：

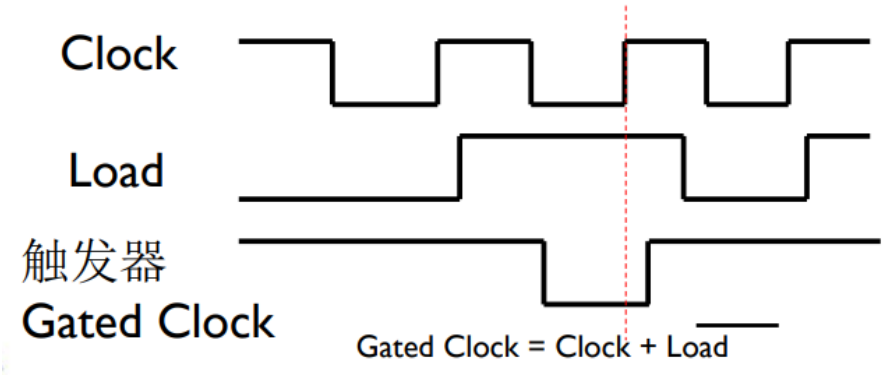
- 完成寄存器的传输设计
- 完成基于多路选择器的寄存器传输设计
- 完成基于 ALU 的数据传输应用设计

2. 实验原理：

2.1. 寄存器

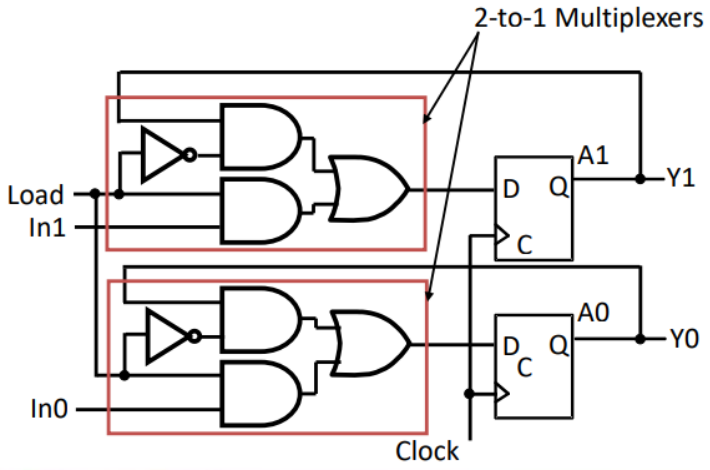
- 一组二进制存储单元。
- 一个寄存器可以用于存储一列二进制值，通常用于进行简单数据存储、移动和处理等操作，能存储信息并保存多个时钟周期，能用信号来控制“保存”或“加载”信息。

- 采用门控时钟的寄存器:、
  - 如果 Load 信号为 1，允许时钟信号通过，如果为 0 则阻止时钟信号通过
  - 例如： 对于上升沿触发的边沿触发器或负向脉冲触发的主从触发器：



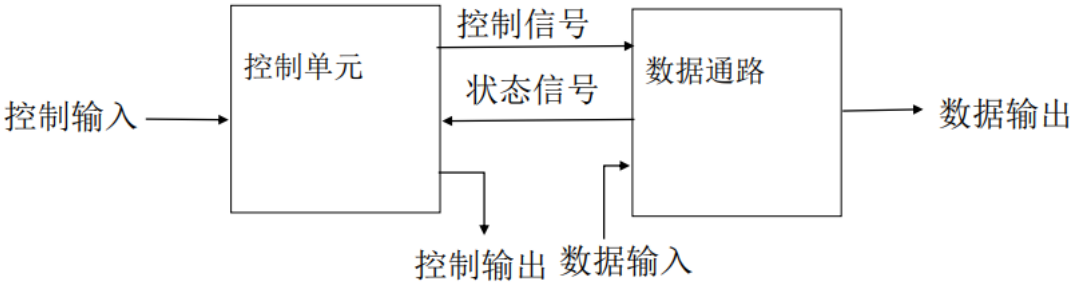
- 采用 Load 控制反馈的寄存器

  进行有选择地加载寄存器的更可靠方法是：
  - 保证时钟的连续性，
  - 选择性地使用加载控制来改变寄存器的内容



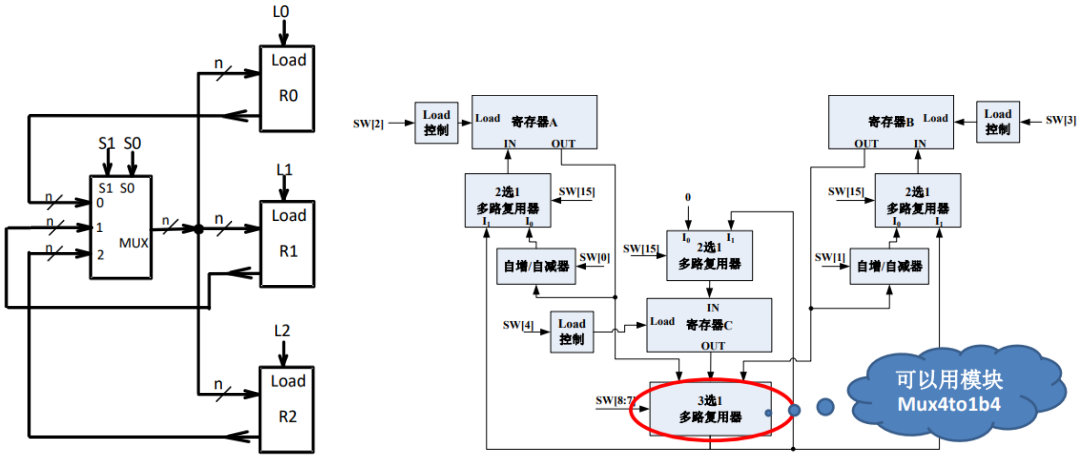
2.2. 寄存器的传输

- 三个基本单元：寄存器组、操作、操作控制
- 基本操作：加载、计数、移位、加法、按位操作



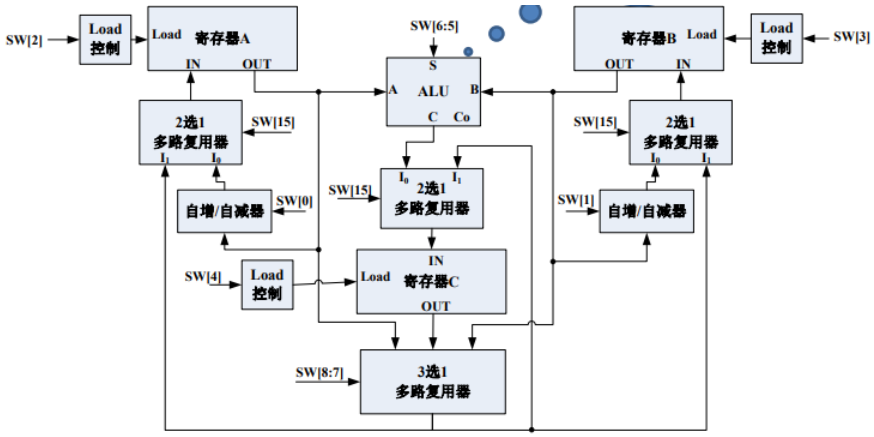
2.3. 基于多路选择器总线的寄存器传输

- 有一个多路选择器驱动的总线可以降低硬件开销
- 但这个结构不能实现多个寄存器相互之间并行传输



2.4. 寄存器传输应用设计

- 将寄存器与 ALU 结合进行应用设计



三、 主要仪器设备

- **实验设备：**

1. 装有 Xilinx ISE 14.7 的计算机　　1 台
2. SWORD 开发板　　1 套

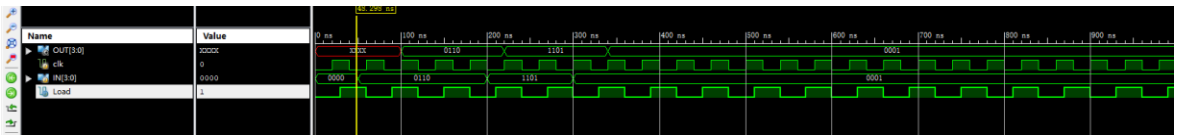
四、 操作方法与实验步骤

1. 采用寄存器传输原理设计计数器

- 新建工程 RegTrans
- Verilog 方式设计寄存器 Reg

```
21  module Register4b(
22     input wire clk,
23     input wire [3:0]IN,
24     input wire Load,
25     output reg [3:0]OUT
26      );
27
28     always @ (posedge clk) begin
29        if (Load) OUT <= IN;
30     end
31
32
33
34  endmodule
```

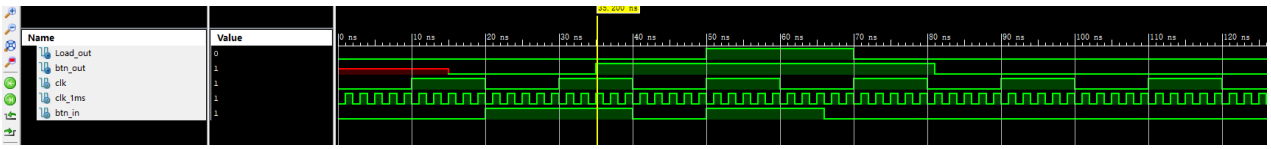
- 进行波形仿真



- Verilog 设计 LoadGen 模块控制寄存器传输信号

```
21  module Load_Gen(
22     input wire clk,
23     input wire clk_lms,
24     input wire btn_in,
25     output reg Load_out
26      );
27
28      initial Load_out = 0;
29      wire btn_out;
30      reg old_btn;
31      pbdebounce p0(clk_lms, btn_in, btn_out);
32
33      always @ (posedge clk) begin
34        if((old_btn == 1'b0) && (btn_out == 1'b1))
35           Load_out <= 1'b1;
36        else
37           Load_out <= 1'b0;
38      end
39
40      always @ (posedge clk) begin
41        old_btn <= btn_out;
42      end
43
44  endmodule
```

- 进行波形仿真



- 建立 top 顶层文件，采用 Verilog 设计

```
module TOP(
    input wire clk,
    input wire [15:0]SW,
    output wire [3:0]AN,
    output wire [7:0]SEGMENT
     );

     wire Load_A, Co;
     wire [3:0] A, A_IN, A1;
     wire [31:0] clk_div;

     clkdiv clk0(clk, 1'b0, clk_div);

     Register4b Reg0(.clk(clk), .IN(A_IN), .Load(Load_A), .OUT(A));

     Load_Gen load0(.clk(clk), .clk_1ms(clk_div[17]), .btn_in(SW[2]), .Load_out(Load_A));

     AddSub4b AddSub0(.A(A), .B(4'b0001), .Ctrl(SW[0]), .S(A1));

     assign A_IN = (SW[15] == 1'b0) ? A1 : 4'b0000;

     DispNum dispnum0(.clk(clk), .HEXS({A, A1, A_IN, 4'b0000}), .LES(4'b0), .points(4'b0), .RST(1'b0), .AN(AN), .Segment(SEGMENT));


endmodule
```

2. 基于多路选择器总线的寄存器传输

- 新建工程 RegTransMux

- 建立 top 顶层文件，采用 Verilog 设计，寄存器和传输信号同 1

```
module TOP(
    input wire clk,
    input wire [15:0]SW,
    output wire [3:0]AN,
    output wire [7:0]SEGMENT
     );

     wire [31:0]clk_div;
     wire A_LOAD, B_LOAD, C_LOAD;
     wire [3:0]A_IN, B_IN, C_IN;
     wire [3:0]A_OUT, B_OUT, C_OUT, F_OUT;
     wire [3:0]A_AS, B_AS;

    clkdiv m_clk(clk, 1'b0, clk_div);

    Load_Gen m_loadA(.clk(clk), .clk_1ms(clk_div[17]), .btn_in(SW[2]), .Load_out(A_LOAD));
    Load_Gen m_loadB(.clk(clk), .clk_1ms(clk_div[17]), .btn_in(SW[3]), .Load_out(B_LOAD));
    Load_Gen m_loadC(.clk(clk), .clk_1ms(clk_div[17]), .btn_in(SW[4]), .Load_out(C_LOAD));

    Mux4to1b4_SCH Mux(.s(SW[8:7]), .I0(A_AS), .I1(B_AS), .I2(C_OUT), .I3(4'b1111), .o(F_OUT));

    Register4b regA(.clk(clk), .IN(A_IN), .Load(A_LOAD), .OUT(A_OUT));
    Register4b regB(.clk(clk), .IN(B_IN), .Load(B_LOAD), .OUT(B_OUT));
    Register4b regC(.clk(clk), .IN(C_IN), .Load(C_LOAD), .OUT(C_OUT));

    AddSub4b asA(.A(A_OUT), .B(4'b0001), .Ctrl(SW[0]), .S(A_AS));
    AddSub4b asB(.A(B_OUT), .B(4'b0001), .Ctrl(SW[1]), .S(B_AS));

    assign A_IN = (SW[15] == 1'b0) ? A_AS : F_OUT;
    assign B_IN = (SW[15] == 1'b0) ? B_AS : F_OUT;
    assign C_IN = (SW[15] == 1'b0) ? 4'b0 : F_OUT;

    DispNum m_disp(.clk(clk), .HEXS({F_OUT, A_OUT, B_OUT, C_OUT}), .LES(4'b0), .points(4'b0), .RST(1'b0), .AN(AN), .Segment(SEGMENT));


endmodule
```

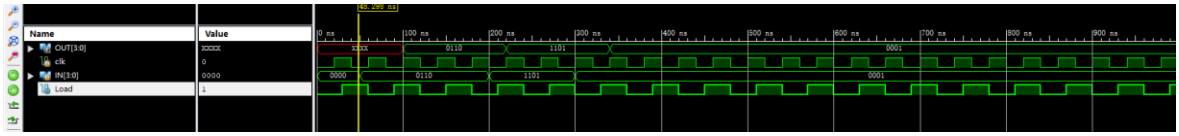
3. 基于 ALU 的数据传输应用设计

- 新建工程 RegTransALU
- 建立 top 顶层文件，采用 Verilog 设计，ALU 同实验 8&9

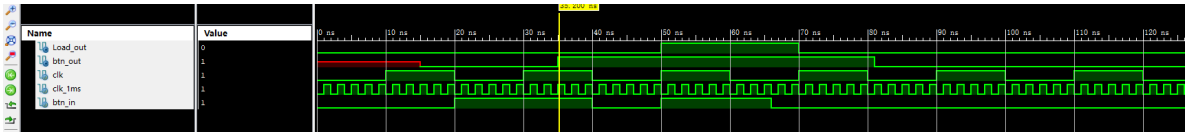
```
module TOP(
    input wire clk,
    input wire [15:0]SW,
    output wire [3:0]AN,
    output wire [7:0]SEGMENT
     );

     wire [31:0]clk_div;
     wire A_LOAD, B_LOAD, C_LOAD;
     wire [3:0]A_IN, B_IN, C_IN;
     wire [3:0]A_OUT, B_OUT, C_OUT, F_OUT;
     wire [3:0]A_AS, B_AS, C_ALU;

    clkdiv m_clk(clk, 1'b0, clk_div);

    Load_Gen m_loadA(.clk(clk), .clk_1ms(clk_div[17]), .btn_in(SW[2]), .Load_out(A_LOAD));
    Load_Gen m_loadB(.clk(clk), .clk_1ms(clk_div[17]), .btn_in(SW[3]), .Load_out(B_LOAD));
    Load_Gen m_loadC(.clk(clk), .clk_1ms(clk_div[17]), .btn_in(SW[4]), .Load_out(C_LOAD));

    Mux4to1b4_SCH Mux(.s(SW[8:7]), .I0(A_AS), .I1(B_AS), .I2(C_OUT), .I3(4'b1111), .o(F_OUT));

    Register4b regA(.clk(clk), .IN(A_IN), .Load(A_LOAD), .OUT(A_OUT));
    Register4b regB(.clk(clk), .IN(B_IN), .Load(B_LOAD), .OUT(B_OUT));
    Register4b regC(.clk(clk), .IN(C_IN), .Load(C_LOAD), .OUT(C_OUT));

    AddSub4b asA(.A(A_OUT), .B(4'b0001), .Ctrl(SW[0]), .S(A_AS));
    AddSub4b asB(.A(B_OUT), .B(4'b0001), .Ctrl(SW[1]), .S(B_AS));
    myALU_SCH ALU(.S(SW[6:5]), .A(A_OUT), .B(B_OUT), .C(C_ALU));

    assign A_IN = (SW[15] == 1'b0) ? A_AS : F_OUT;
    assign B_IN = (SW[15] == 1'b0) ? B_AS : F_OUT;
    assign C_IN = (SW[15] == 1'b0) ? C_ALU : F_OUT;

    DispNum m_disp(.clk(clk), .HEXS({F_OUT, A_OUT, B_OUT, C_OUT}), .LES(4'b0), .points(4'b0), .RST(1'b0), .AN(AN), .Segment(SEGMENT));

endmodule
```

五、 实验数据记录和处理

- 4 位寄存器波形仿真：



- Load 信号生成的波形仿真：



六、 实验结果与分析

通过对波形进行分析，寄存器和传输信号的设计均正确。通过实验验收也可以发现，基于 Mux 的寄存器传输设计正确。拨动开关 SW2/3/4 可以 load 寄存器 A/B/C。而 load 的值首先由 SW15 确定。当 SW15==1 时，用多路选择器 SW7/8 控制选择传入数据；当 SW15=0 时，用 SW0/1 控制对应寄存器自增自减。通过实验验收也可以发现，基于 ALU 的寄存器应用设计正确。在 MUX 的基础上，该设计可以将寄存器 AB 中的值传入 ALU，通过 SW5/6 控制 ALU 对输入数据的计算，将输出结果经 SW15 选择 load 输入寄存器 C。总体实验设计成功。

七、 讨论、心得

在这次实验中，我深入理解了寄存器传输电路的工作原理，明晰了寄存器传输电路的设计方法，并掌握了 ALU 和寄存器传输电路的综合应用。

我先理解了寄存器传输电路的工作原理。寄存器传输电路是数字逻辑电路的重要组成部分，它的主要任务是完成数据的接收、保持和转移，起到临时存储信息的作用。

在此之上，我掌握了寄存器传输电路的设计方法。设计寄存器传输电路需要明确输入信号。本次实验我正确设计了 Load_Gen 数据传输信号，通过仿真验证正确性后将其与寄存器组合，设计出信号传输寄存器。

最后，我掌握了 ALU 和寄存器传输电路的综合应用。通过使用 MUX 对寄存器进行选择，并将 ALU 嵌入其中，两者的结合使计算机的运算和存储更加高效。实现更多功能。

总的来说，这次实验让我对寄存器有了更深入的了解，也对寄存器的实现原理有了更清晰的认识。希望本次的实验设计能对以后的复杂电路设计起到帮助

实验日期：2023.12.11