МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра информационной безопасности

ОТЧЕТ

по лабораторной работе №5
по дисциплине «Электроника и схемотехника»
Тема: «Коммуникация по интерфейсу SPI»

Студент гр. 3363 _____ Минко Д.А.

Студент гр. 3363 _____ Овсейчик Н.И.

Студент гр. 3363 _____ Гончаренко О.Д.

Преподаватель _____ Рыбин В.Г.

Санкт-Петербург
2024

Цель работы

Создать два проекта для ведущего и ведомого устройств (для удобства последующей прошивки). В каждом из них описать отдельно модуль SPI и поместить его в оболочку, которая будет передавать сигналы в модуль и выводить передаваемые данные на светодиоды.

Ход работы

1. В проекте для ведущей платы разработан модуль SPI, реализующий управление состояниями передачи и ожидания, а также контроль успешности передачи пакетов. Этот модуль интегрирован в оболочку, обеспечивающую инкрементирование значений и их передачу с использованием кнопок.
2. В отдельном проекте создан модуль SPI для ведомой платы, который также помещен в оболочку. Данная оболочка реализует передачу значений регистра на светодиоды.
3. Для ведущего и ведомого устройств выполнена отдельная разводка. Схемы разводки идентичны, за исключением того, что на ведомом устройстве добавлены кнопки: одна для увеличения двоичного значения числа, другая для выполнения передачи. На ведущем устройстве значение не сбрасывается автоматически и требует ручного обнуления.
4. При конфигурации устройств необходимо установить режим «Embedded Flash Mode» на обеих платах, чтобы обеспечить сохранение прошивки. Платы дополнительно соединены общей линией GND и питающим пином на 5V.

Вывод

В ходе выполнения лабораторной работы были разработаны два SPI-модуля: для Master-устройства и для Slave-устройства. Оба модуля интегрированы в оболочки, обеспечивающие передачу двоичного значения от ведущего устройства к ведомому.

ПРИЛОЖЕНИЕ 1 – ИСХОДНЫЙ КОД ПРОГРАММЫ

1. Основной модуль master_control

```
module master_control
(
     input clk,
     input increment,
     input transmit,
     output [5:0] led,

     output SCLK,
     output MOSI,
     input MISO,
     output SS
);

reg [7:0] data_out;
wire [7:0] data_in;

reg [19:0] start_counter;
wire start;

reg [19:0] inc_counter;

spi_master m (clk, start, data_out, data_in, SCLK, MOSI, MISO,
SS);

always @(posedge clk)
begin
     if (!increment)
     begin
          if (!inc_counter[19])
          begin
               inc_counter = inc_counter + 1;
               if (inc_counter[19])
                    data_out = data_out + 1;
          end
     end
     else
     begin
          inc_counter <= 0;
     end
end

always @(posedge clk)
begin
     if (!transmit)
     begin
          if (!start_counter[19])
               start_counter <= start_counter + 1;
```

```
     end
     else
     begin
          start_counter <= 0;
     end
end

assign led = ~data_out[5:0];
assign start = start_counter[19];

endmodule
```

2. Вспомогательный модуль spi_master

```
module spi_master
(
     input clk,
     input start,
     input [7:0] data_out,
     output reg [7:0] data_in,

     output reg SCLK,
     output reg MOSI,
     input MISO,
     output reg SS
);

reg [1:0] state;
reg [2:0] data_counter;

initial
begin
     data_in <= 0;
     state <= 0;
     SS <= 1;
     MOSI <= 0;
     SCLK <= 0;
     data_counter <= 0;
end

always @(posedge clk)
begin
     case (state)
          0:
          begin
               if (start) // If not transmitting but commanded to
start
               begin
                    state <= 1;

                    SS <= 0;
                    SCLK <= 0;
                    data_counter <= 7;
                    MOSI <= data_out[7];
               end
          end
          1: // If transmitting
          begin
               if (!SCLK) // If making a posedge
               begin
                    data_in[data_counter] <= MISO;

                    if (!data_counter)
                         state <= 2;
                    else
```

```
                    data_counter <= data_counter - 1'b1;
               end
               else
               begin
                    MOSI <= data_out[data_counter];
               end

               SCLK <= ~SCLK;
          end
          2: // If packets sent
          begin
               SCLK <= 0;
               state <= 3;
               MOSI <= 0;
          end
          3: // If ending transmission
          begin
               SS <= 1;
               state <= 0;
          end
     endcase
end

endmodule
```

3. Распиновка для master_control

```
//Copyright (C)2014-2024 Gowin Semiconductor Corporation.
//All rights reserved.
//File Title: Physical Constraints file
//Tool Version: V1.9.9.03 Education (64-bit)
//Part Number: GW1NR-LV9QN88PC6/I5
//Device: GW1NR-9
//Device Version: C
//Created Time: Sun 09 22 13:27:35 2024

IO_LOC "SS" 28;
IO_PORT "SS" IO_TYPE=LVCMOS18 PULL_MODE=UP DRIVE=8 BANK_VCCIO=1.8;
IO_LOC "MOSI" 26;
IO_PORT    "MOSI"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "SCLK" 27;
IO_PORT    "SCLK"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[5]" 16;
IO_PORT    "led[5]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[4]" 15;
IO_PORT    "led[4]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[3]" 14;
IO_PORT    "led[3]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[2]" 13;
IO_PORT    "led[2]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[1]" 11;
IO_PORT    "led[1]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[0]" 10;
IO_PORT    "led[0]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "transmit" 4;
IO_PORT "transmit" IO_TYPE=LVCMOS18 PULL_MODE=UP BANK_VCCIO=1.8;
IO_LOC "increment" 3;
IO_PORT "increment" IO_TYPE=LVCMOS18 PULL_MODE=UP BANK_VCCIO=1.8;
IO_LOC "clk" 52;
IO_PORT "clk" IO_TYPE=LVCMOS18 PULL_MODE=UP BANK_VCCIO=1.8;
IO_LOC "MISO" 25;
IO_PORT "MISO" IO_TYPE=LVCMOS18 PULL_MODE=UP BANK_VCCIO=1.8;
```

4. Основной модуль slave_control

```
module slave_control
(
     output [5:0] led,

     input SCLK,
     input MOSI,
     output MISO,
     input SS
);

reg [7:0] data_out;
wire [7:0] data_in;

spi_slave s (data_out, data_in, SCLK, MOSI, MISO, SS);

assign led = ~data_in[5:0];

endmodule
```

5. Вспомогательный модуль spi_slave

```
module spi_slave
(
     input [7:0] data_out,
     output reg [7:0] data_in,

     input SCLK,
     input MOSI,
     output MISO,
     input SS
);

reg state;
reg [2:0] data_counter;

initial
begin
     data_in <= 0;
     state <= 0;
     data_counter <= 0;
end

always @(posedge SCLK)
begin
     if (!SS)
     begin
          if (!state)
          begin
               state = 1;
               data_counter = 7;
          end

          data_in = { data_in[6:0], MOSI };
          if (data_counter == 0) state = 0;
          data_counter = data_counter - 1;
     end
end

assign MISO = state;

endmodule
```

6. Распиновка для slave_control

```
//Copyright (C)2014-2024 Gowin Semiconductor Corporation.
//All rights reserved.
//File Title: Physical Constraints file
//Tool Version: V1.9.9.03 Education (64-bit)
//Part Number: GW1NR-LV9QN88PC6/I5
//Device: GW1NR-9
//Device Version: C
//Created Time: Sun 09 22 13:47:32 2024

IO_LOC "MISO" 25;
IO_PORT     "MISO"     IO_TYPE=LVCMOS18     PULL_MODE=UP     DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[5]" 16;
IO_PORT    "led[5]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[4]" 15;
IO_PORT    "led[4]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[3]" 14;
IO_PORT    "led[3]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[2]" 13;
IO_PORT    "led[2]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[1]" 11;
IO_PORT    "led[1]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "led[0]" 10;
IO_PORT    "led[0]"    IO_TYPE=LVCMOS18    PULL_MODE=UP    DRIVE=8
BANK_VCCIO=1.8;
IO_LOC "SS" 28;
IO_PORT "SS" IO_TYPE=LVCMOS18 PULL_MODE=UP BANK_VCCIO=1.8;
IO_LOC "MOSI" 26;
IO_PORT "MOSI" IO_TYPE=LVCMOS18 PULL_MODE=UP BANK_VCCIO=1.8;
IO_LOC "SCLK" 27;
IO_PORT "SCLK" IO_TYPE=LVCMOS18 PULL_MODE=UP BANK_VCCIO=1.8;
```