

# ПРОЕКТИРОВАНИЕ СНК С ПРОГРАММИРУЕМОЙ АРХИТЕКТУРОЙ

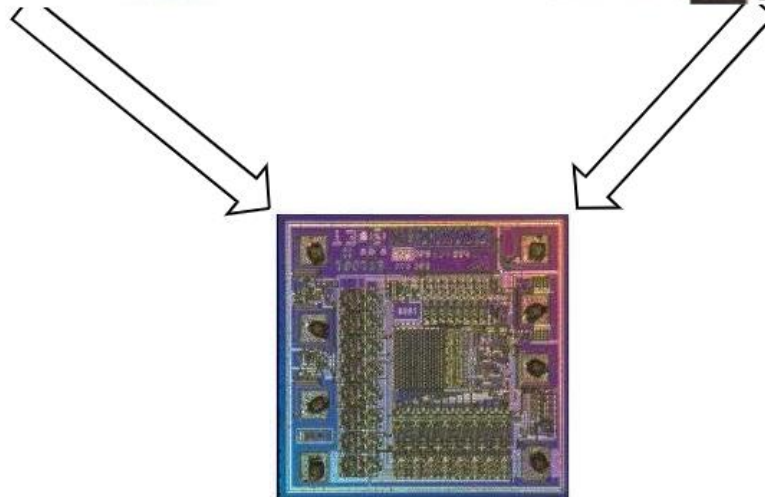
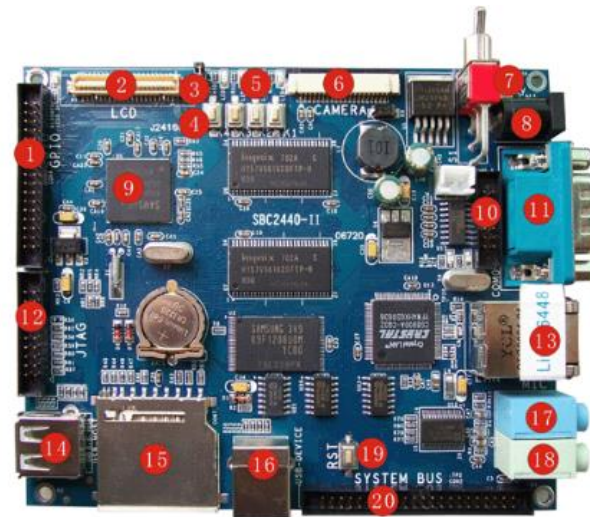
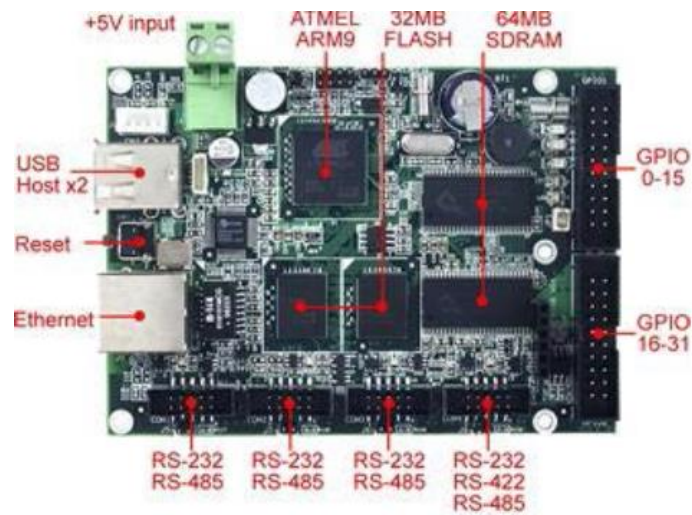
---

Лабораторная работа №1

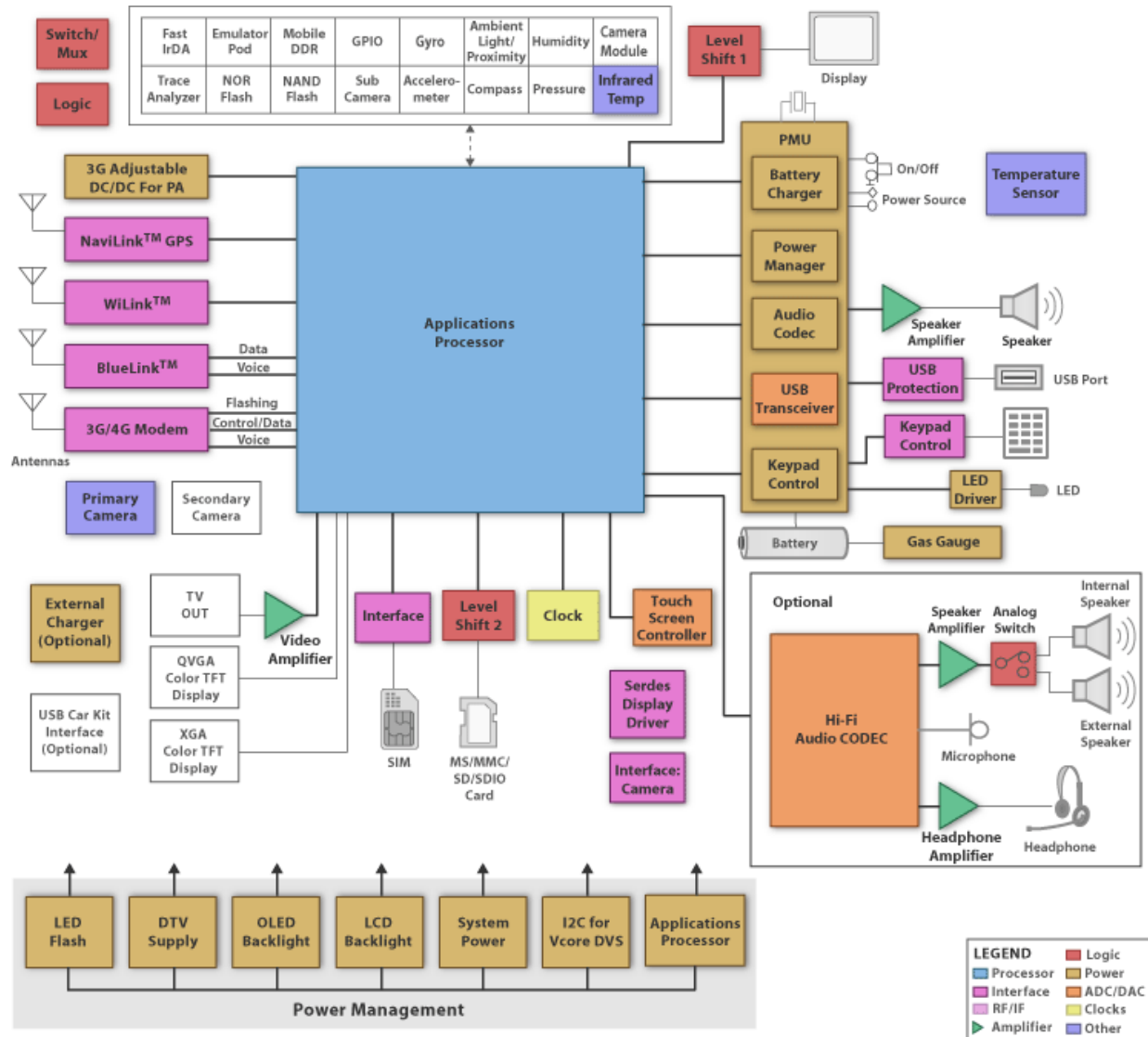
# System on chip

- Definition
  - (nearly) complete embedded system on a single chip
- Usually includes
  - Programmable processor(s)
  - Memory
  - Accelerating function units
  - Input/output interfaces
  - Software
  - Re-usable intellectual property blocks (HW + SW)

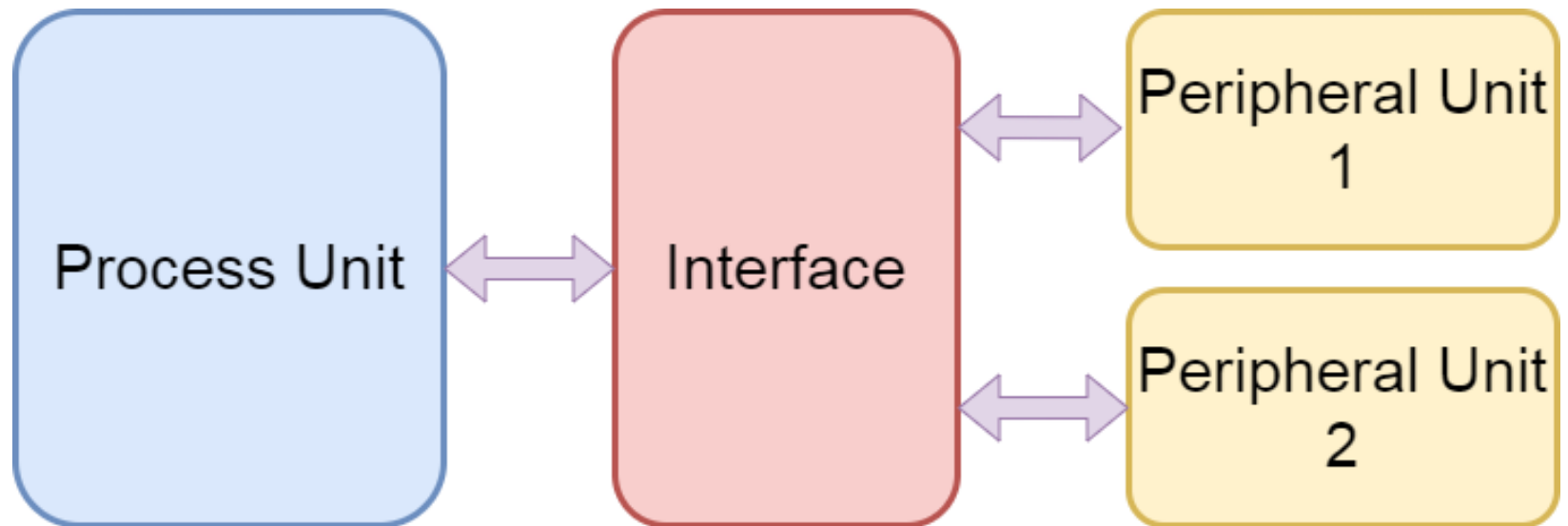
# SoC Design Goal



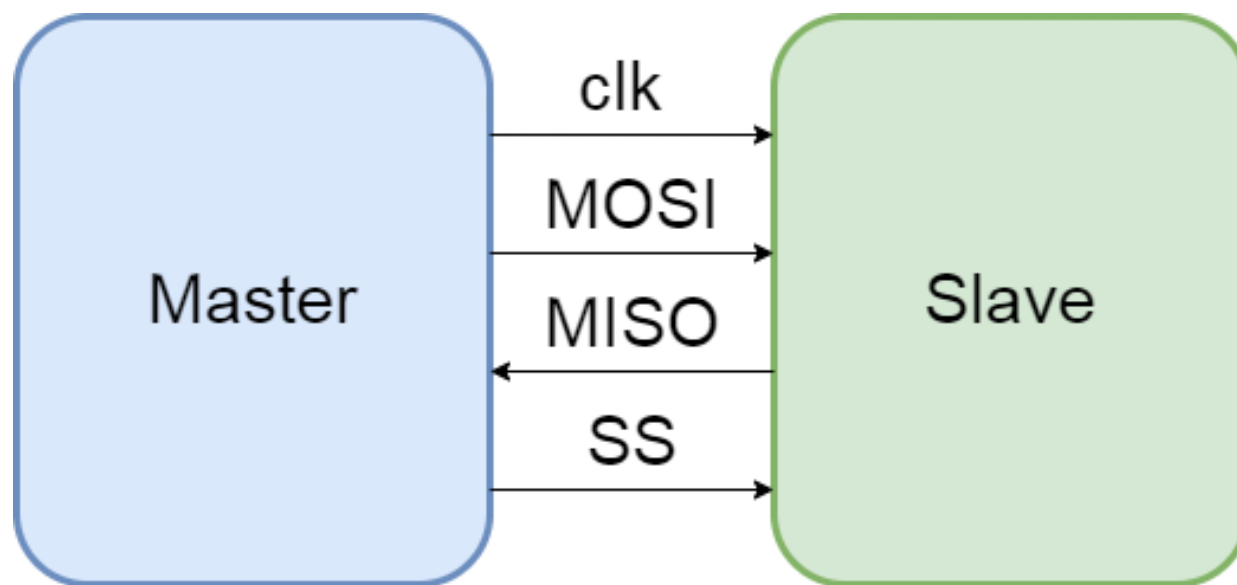
# SoC example



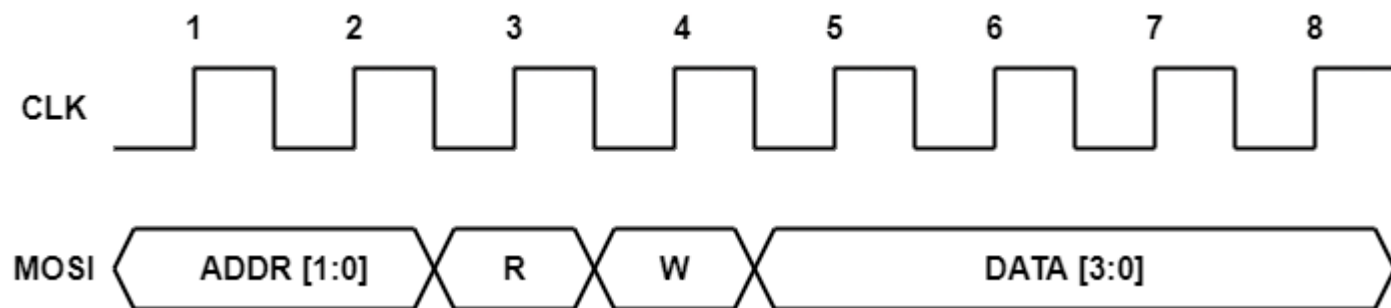
# Интерфейс



# Интерфейс SPI

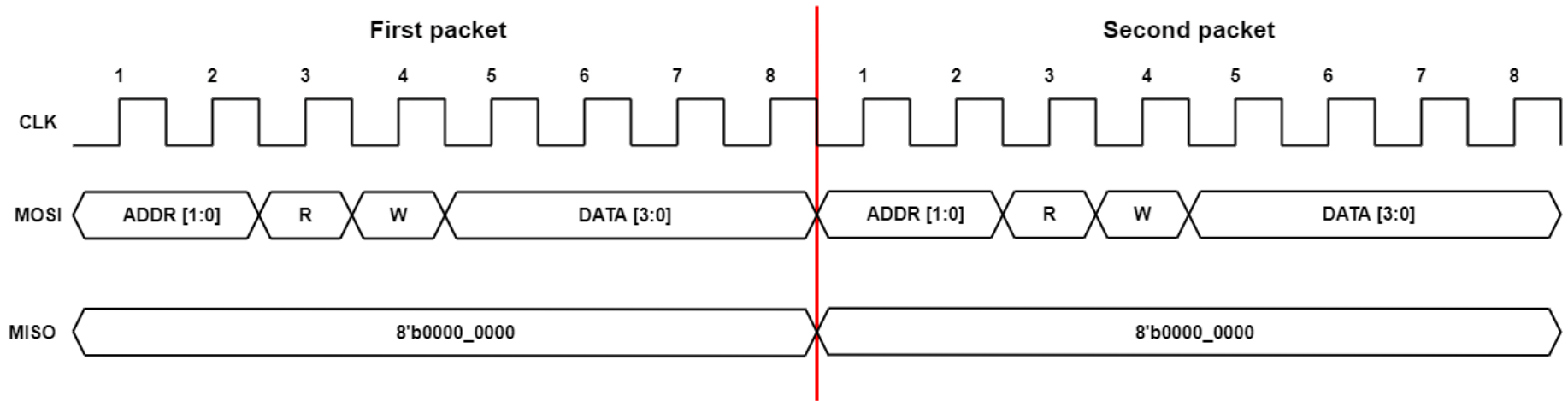


# Интерфейс SPI

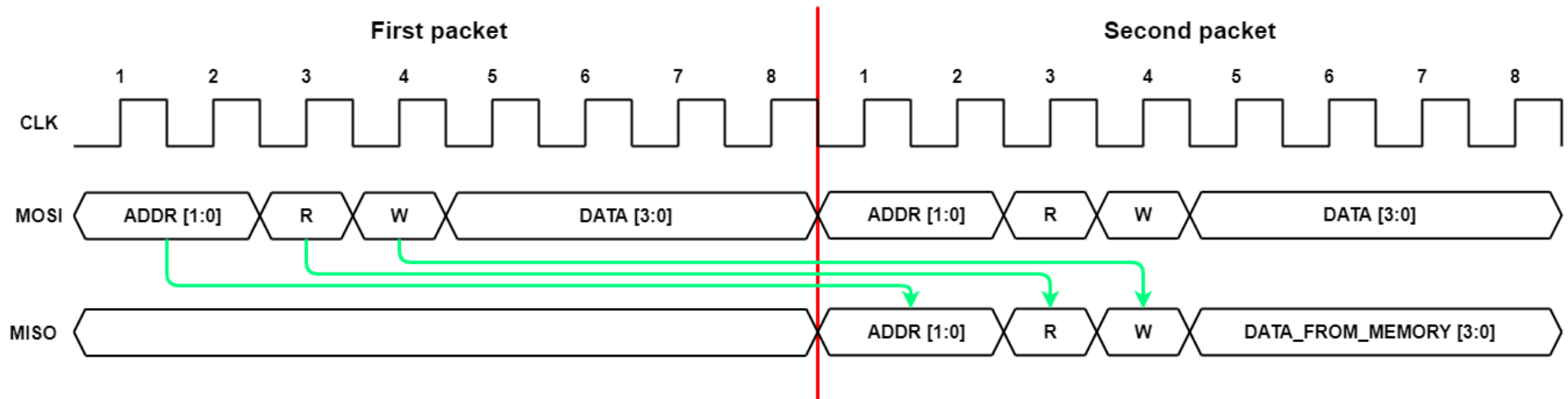


# Интерфейс SPI

## Запись



## Чтение





# Описание интерфейса SPI на SV

**Интерфейс (в SystemVerilog)** – метод инкапсуляции сигналов в логическую группу с целью упрощения взаимодействия с необходимой группой сигналов. Используется для дальнейшего упрощения использования группы сигналов в разрабатываемых модулях.

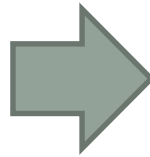
## Свойства интерфейсов:

- Параметризация;
- Можно задать внешние сигналы (clk, rst и д.р.);
- Определение направления каждого из сигналов через modport;
- Наследуемость параметров.

```
1. interface SPI#(  
2.     parameter  SLAVES_NUM = 2  
3. )(  
4.     input      clk,  
5.     input      rst  
6. );  
7.  
8.     logic              MOSI;  
9.     logic              MISO;  
10.    logic [SLAVES_NUM-1:0] SS;  
11.  
12. endinterface : SPI
```

# Описание интерфейса SPI на SV

```
1. module Slave(  
2.     input  clk,  
3.     input  mosi,  
4.     input  ss,  
5.     output miso  
6. );  
7.  
8. /* logic */  
9.  
10. endmodule : Slave
```



```
1. module Slave(  
2.     SPI_if spi  
3. );  
4.  
5. /* logic */  
6.  
7. endmodule : Slave
```

# Пример описания лог. выражения с использованием интерфейсов

```
1.  logic [3:0]      cnt;
2.  logic [7:0]      mosi_pkg;
3.  logic [7:0]      miso_pkg;
4.  logic [3:0] [3:0] memory;
5.
6.  always_ff@(negedge spi.clk or negedge spi.rst)
7.  if (!spi.rst)
8.      cnt <= 4'hF;
9.  else if ( cnt <= 4'h7 )
10.      cnt <= cnt + 1;
11.  else
12.      cnt <= 4'h0;
13.
14. always_ff @( posedge spi.clk or negedge spi.rst )
15.  if(!spi.rst)
16.      mosi_pkg <= 8'h00;
17.  else
18.      mosi_pkg <= {mosi_pkg[6:0], spi.MOSI};
```

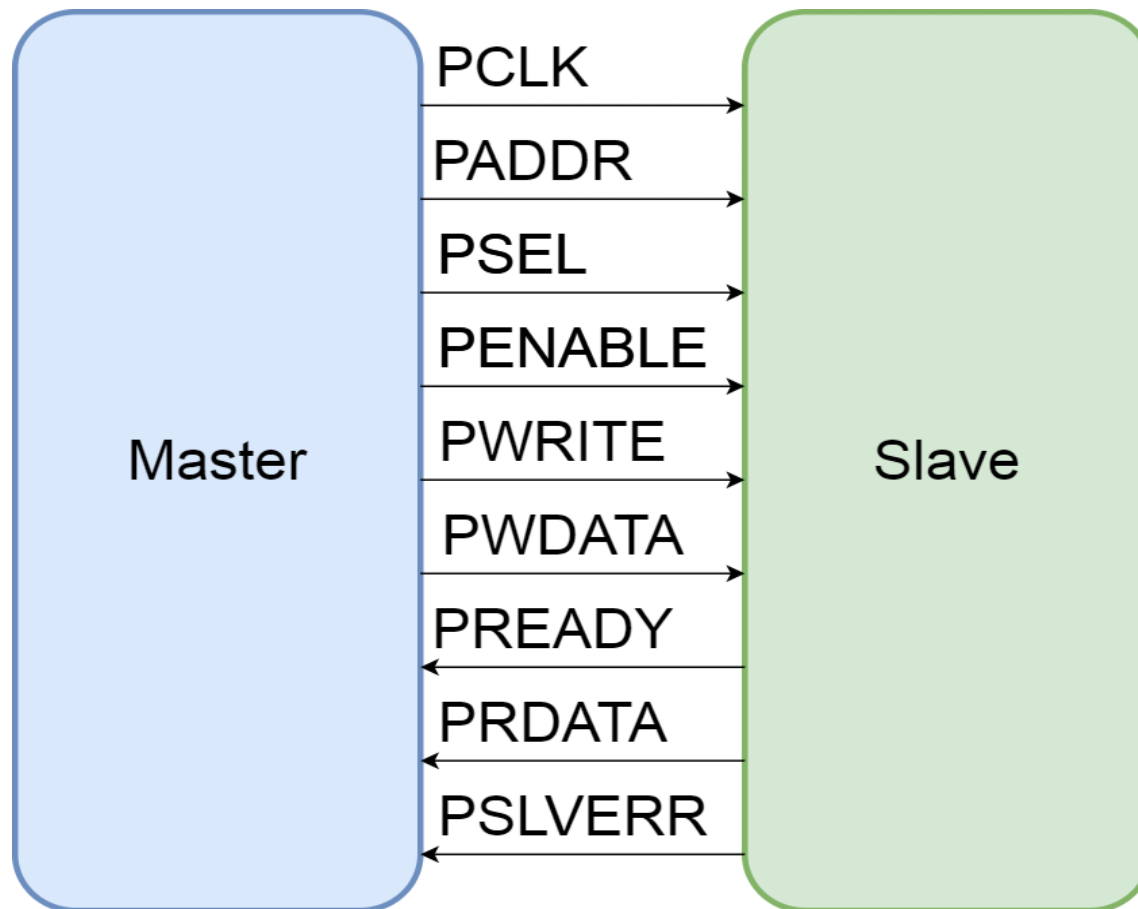
```
19. always_ff @( negedge spi.clk or negedge spi.rst)
20.  if (spi.rst)
21.      miso_pkg <= 8'h00;
22.  else if ( cnt == 4'h7 && mosi_pkg[5] )
23.      miso_pkg <= {mosi_pkg[7:4],
memory[mosi_pkg[7:6]]};
24.  else
25.      miso_pkg <= miso_pkg << 1;
26.
27.  always_comb spi.MISO = miso_pkg[7];
```

# Описание ТВ SPI на SV

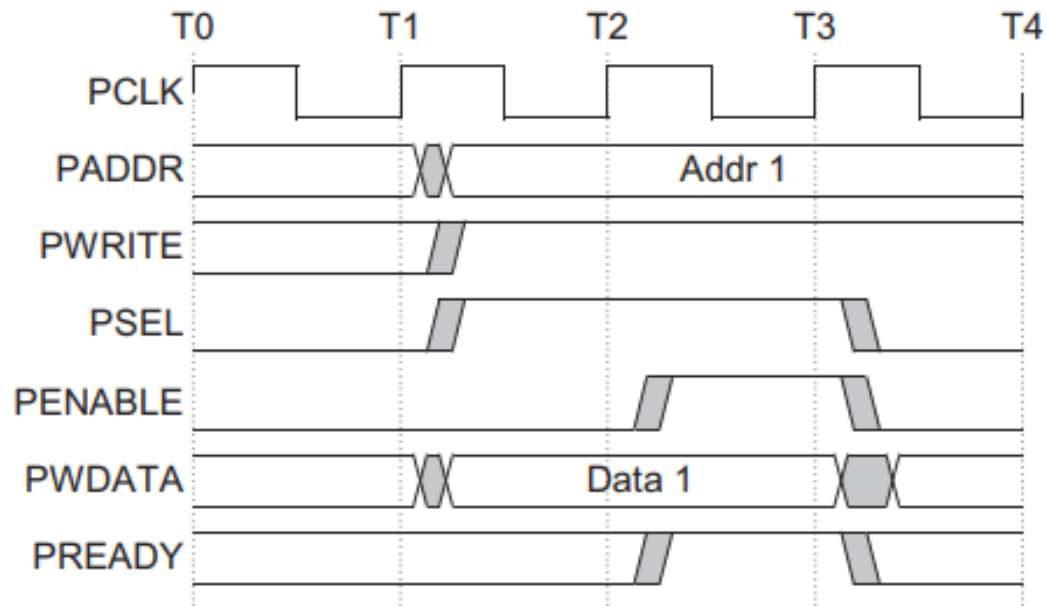
```
1. module tb;
2.
3.   logic    clk  = '0;
4.   logic    rstn = '0;
5.   logic [7:0] mosi_pkg;
6.   logic [7:0] miso_pkg;
7.
8.   SPI my_spi( clk, rstn );
9.   Slave udb_slave( my_spi );
10.
11.  initial begin
12.    #5ns;
13.    rstn = 1;
14.    spi_do( 2'b01, 1'b0, 1'b1, 4'b1011 );
15.    #15ns $stop;
16.  end
17.
18.  always #10ns clk = ~clk;
19.
20. endmodule : tb
```

```
1. task spi_do (
2.   input [1:0]   addr,
3.   input         r,
4.   input         w,
5.   input [3:0]   data
6. );
7.
8.   logic [7:0] pkg;
9.   my_spi.SS   = 1;
10.  pkg          = {addr, r, w, data };
11.
12.  assign my_spi.MOSI = pkg[7];
13.
14.  repeat(8) @( negedge clk ) begin
15.    my_spi.MOSI = pkg[7];
16.    pkg          = pkg << 1;
17.  end
18.
19. endtask
```

# Интерфейс AMBA APB3

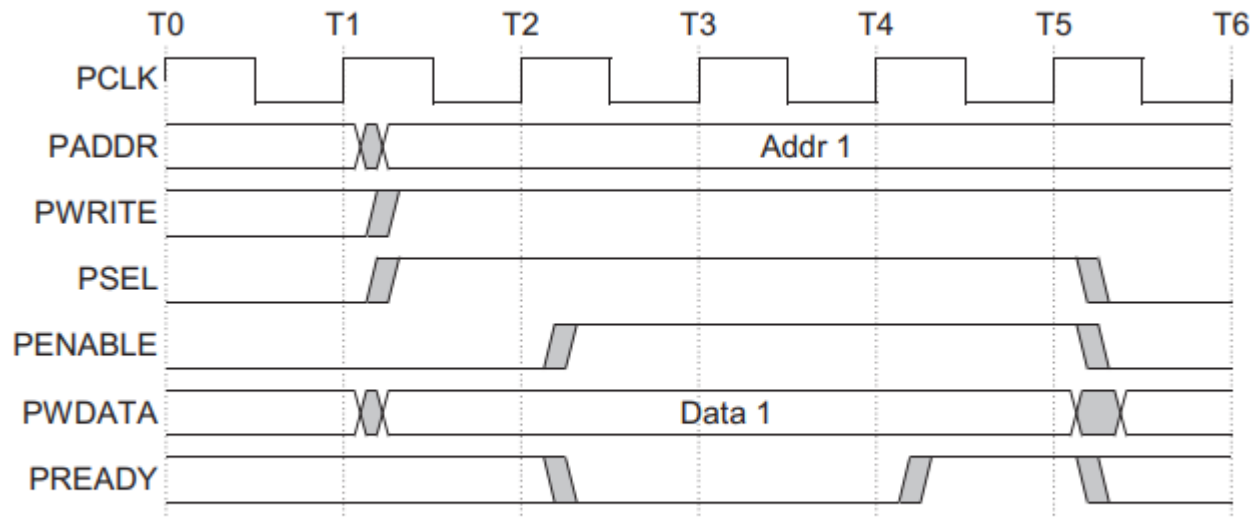


# Интерфейс APB, запись



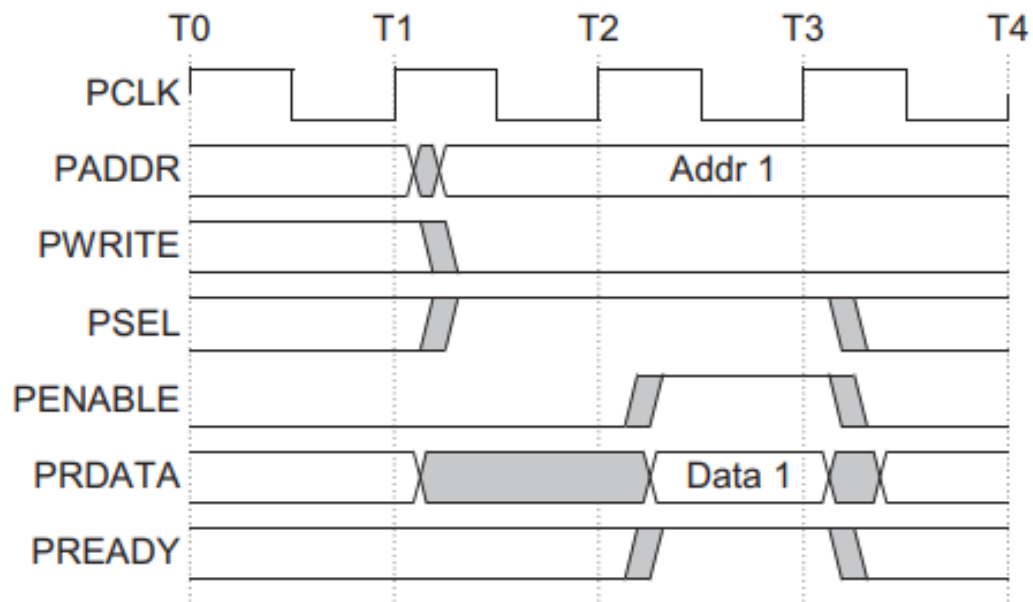
**Figure 3-1 Write transfer with no wait states**

# Интерфейс APB, запись (wait)



**Figure 3-2 Write transfer with wait states**

# Интерфейс APB, чтение



**Figure 3-4 Read transfer with no wait states**



# Интерфейс APB, чтение (wait)

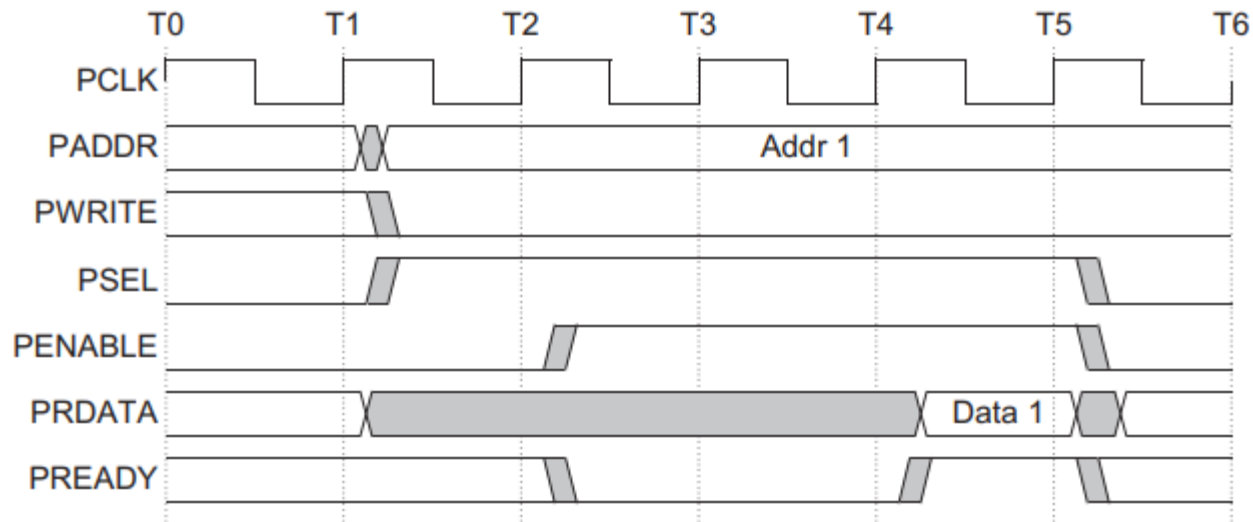


Figure 3-5 Read transfer with wait states

# Цели лабораторной работы

- 1) Реализовать APB3 SystemVerilog интерфейс;
- 2) Написать модуль, реализующий протокол APB3 с использованием написанного вами интерфейса (без адресной логики – достаточно 1 регистра-памяти);
- 3) Сделать тестбенч для вашего модуля, продемонстрировать процессы записи и чтения.