ADI 7bit地址8bit数据三线制SPI总线驱动IP核

用户使用说明书

目录

[1 简介 3](#_Toc116206065)

[2 IP核介绍 3](#_Toc116206066)

[2.1 特性 3](#_Toc116206067)

[2.2 信号和参数列表 3](#_Toc116206068)

[3 IP核应用参考 5](#_Toc116206069)

[3.1 ISE中纯逻辑驱动例化参考 5](#_Toc116206070)

[3.1.1 仿真例程 6](#_Toc116206071)

[3.2 VIVADO中纯逻辑驱动例化参考 7](#_Toc116206072)

[3.3 VIVADO中带AXI总线的IP包装器 8](#_Toc116206073)

[3.3.1 IP寄存器描述 8](#_Toc116206074)

[3.3.2 C语言读写访问例程 10](#_Toc116206075)

[3.4 在DSP + FPGA构架的应用 10](#_Toc116206076)

[3.4.1 IP寄存器描述 10](#_Toc116206077)

[3.4.2 C语言读写访问例程 12](#_Toc116206078)

# 简介

为提高FPGA设计效率，对于常用且重复性很强的功能块，不需要每次都进行设计，通过形成IP核的方式，可提高设计模块的重用性和设计效率。至此开始着手对常用的协议以尽可能完善的形式组建用户IP核。

# IP核介绍

该系列IP主要为ADI公司芯片提供的三线制SPI总线驱动，由于ADI公司的SPI接口驱动的时序以及位的含义相对比较固定，最多为地址位宽的不同，所以该系列IP核主要以地址位宽和数据位宽来进行命名，例如本IP命名为“ADI 7bit地址8bit数据三线制SPI总线驱动”。

器件适配表如下，已经在该器件上进行了验证，后续可能会更新更多的适配器件：

|  |  |  |
| --- | --- | --- |
| 适配器件列表 | 名称 | 型号 |
| DAC | AD9704/AD9705/AD9706/AD9707 |
|  |  |

*注：此驱动只支持每次配置一个字节的数据，7BIT地址为通用定义，请根据具体芯片手册，给定不同的地址组合（特别需要注意）如AD9707的7BIT地址组合如下：*

*|N1|N0|A4|A3|A2|A1|A0| 其中N1N0的值不是地址，在AD970X系列中控制的是单次SPI传输的数据的字节数，由于本驱动每次只传输一个字节，所以在使用时上层NIN0的值只能给两个0。 由于每个芯片对7BIT地址的定义不同，所以需要结合实际情况来编写上层驱动，以上是AD970X列举了一个示例。*

## 特性

◆ **运行主时钟频率可配置；**

◆ **SPI接口速率可配置**；

## 信号和参数列表

|  |  |  |  |
| --- | --- | --- | --- |
| 参数 | 名称 | 含义 | 示例 |
| CLK\_FRE | 运行时钟频率 | 配置IP所实际运行使用的主时钟频率；  默认100000000 ->100Mhz |
| SCLK\_FRE | SPI总线速率 | 配置数据接口总线速率，最大不超过20Mhz  默认 1000000 –> 1Mhz |

**参数配置的注意事项:**

1. SCLK \_FRE： 配置的BUS \_FRE频率必须能被CLK\_FRE整除，否则实际运行时钟频率将大于设置的频率（除不尽有小数，小数不被逻辑执行）。（不关心该频率也可忽略）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 端口 | 名称 | 属性 | 位宽 | 说明 |
| 系统接口 | clk | in | 1 | 系统时钟输入 |
| rst\_n | in | 1 | 系统复位输入：低电平有效 |
| user\_wr\_data | user\_wr\_en | in | 1 | 写使能，一个时钟周期 |
| user\_wr\_addr | in | 7 | 写地址 |
| user\_wr\_data | in | 8 | 写数据 |
| user\_rd\_en | in | 1 | 读使能，一个时钟周期 |
| user\_rd\_addr | in | 7 | 读地址 |
| user\_rd\_data | out | 8 | 读数据 |
| user\_op\_busy | out | 1 | 1:驱动忙  0:驱动空闲 |
| user\_wr\_vild | out | 1 | 写完成信号（一个时钟周期高电平） |
| user\_rd\_vild | out | 1 | 读完成信号（一个时钟周期高电平） |
| AD970X 数据接口 | rst\_spi\_pin | out | 1 | rst\_spi\_pin为特殊管脚，不是SPI总线的一部分，其时序为在上电以后只工作一次，顺序为低电平200us、高电平600us之后一直处于低电平。  以AD970x系列为例：该管脚主要用于使AD970x芯片进入SPI模式，并复位寄存器到默认值  如果别的芯片有类似的功能，可将该管脚引出，如果没有可将该管脚悬空即可（可作为高电平的复位引脚使用）。 |
| scb | out | 1 | 芯片的SPI总线接口 |
| sclk | out | 1 |
| sdio | inout | 1 |
| 方向控制接口 | dir | out | 1 | 外围电平转换控制芯片接口，SPI sdio输出时为高电平，sdio输入时为低电平 |

# IP核应用参考

该IP在ISE中使用仅提供一种源码例化，使用纯逻辑进行驱动的方式。在VIVADO中使用时，提供纯逻辑驱动的IP包装器形式和带AXI总线的包装器形式。

## ISE中纯逻辑驱动例化参考

**adi\_spi\_driver\_7\_8bit**

**#(**

**.CLK\_FRE ( 100\_000\_000 ) ,**

**.SCLK\_FRE ( 1\_000\_000 )**

**)**

**adi\_spi\_driver\_7\_8bit\_inst**

**(**

**.clk ( clk ),**

**.rst\_n ( rst\_n ),**

**.user\_wr\_en ( user\_wr\_en ),**

**.user\_wr\_addr ( user\_wr\_addr ),**

**.user\_wr\_data ( user\_wr\_data ),**

**.user\_rd\_en ( user\_rd\_en ),**

**.user\_rd\_addr ( user\_rd\_addr ),**

**.user\_rd\_data ( user\_rd\_data ),**

**.user\_op\_busy ( user\_op\_busy ),**

**.rst\_spi\_pin ( rst\_spi\_pin ),**

**.csb ( csb ),**

**.sclk ( sclk ),**

**.sdio ( sdio ),**

**.dir ( dir )**

**);**

### 仿真例程

仿真工程仿真对设计分别进行了一次写访问和读访问。

//generate signal

initial begin

#1;

//initialization

user\_wr\_en = 0;

user\_wr\_addr = 7'h0;

user\_wr\_data = 8'h0;

user\_rd\_en = 0;

user\_rd\_addr = 7'h0;

sdio\_vlg = 1'bz ;

#(10\*CYCLE);

//generate

#(100000\*CYCLE);

user\_wr\_en = 1;

user\_wr\_addr = 7'h4A;

user\_wr\_data = 8'h55;

#(1\*CYCLE);

user\_wr\_en = 0;

#(2500\*CYCLE);

user\_rd\_en = 1;

user\_rd\_addr = 15'h5A;

#(1\*CYCLE);

user\_rd\_en = 0;

#(800\*CYCLE);

sdio\_vlg = 1'b0 ;

for(i=0;i<8;i=i+1)begin //产生一个数据0xAA

sdio\_vlg <= ~sdio\_vlg;

#(100\*CYCLE);

end

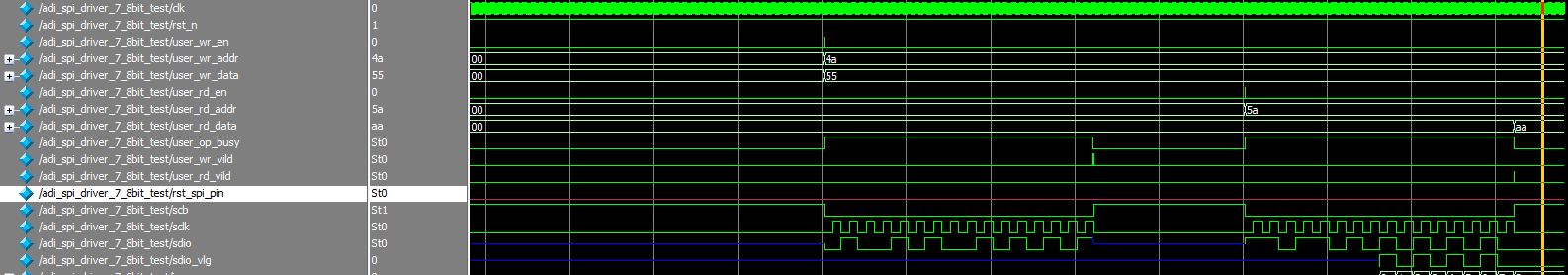
#(300\*CYCLE);

$display("---------test finish--------------");

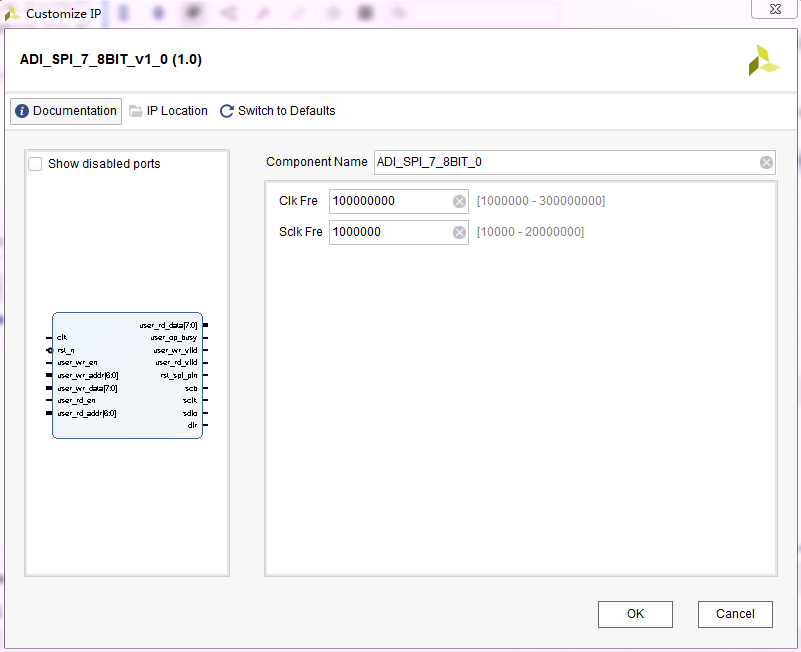
$stop;

End

仿真结果如下图：



## VIVADO中纯逻辑驱动例化参考



**adi\_spi\_driver\_7\_8bit adi\_spi\_driver\_7\_8bit\_inst**

**(**

**.clk ( clk ),**

**.rst\_n ( rst\_n ),**

**.user\_wr\_en ( user\_wr\_en ),**

**.user\_wr\_addr ( user\_wr\_addr ),**

**.user\_wr\_data ( user\_wr\_data ),**

**.user\_rd\_en ( user\_rd\_en ),**

**.user\_rd\_addr ( user\_rd\_addr ),**

**.user\_rd\_data ( user\_rd\_data ),**

**.user\_op\_busy ( user\_op\_busy ),**

**.rst\_spi\_pin ( rst\_spi\_pin ),**

**.csb ( csb ),**

**.sclk ( sclk ),**

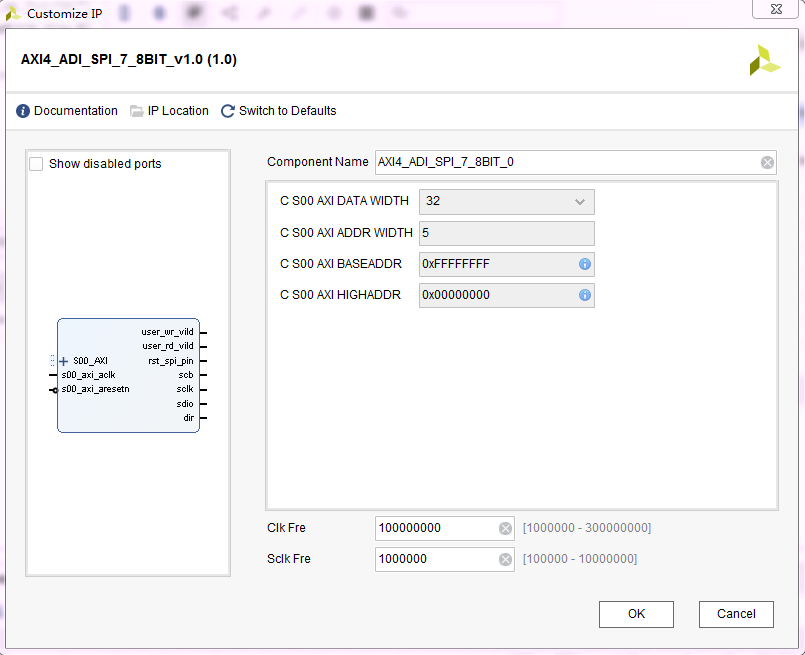
**.sdio ( sdio ),**

**.dir ( dir )**

**);**

仿真使用方法同ISE纯逻辑驱动，不再赘述。

## VIVADO中带AXI总线的IP包装器



使用该形式的IP可以将IP挂在在AXI总线上，从而使用C语言对外设进行读写访问。

### IP寄存器描述

这些寄存器在访问时需要加上IP在AXI总线上被实际分配的基地址。

#### 写地址寄存器(0x0000\_0000)(读写)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** | 0x0000\_0000 | **属性** | 读写 |
| **位** | **含义** | | **复位值** |
| 31~7 | -- | | 0 |
| 6~0 | 此处写入需要写访问的总线的地址，地址长度为7bit; | |  |

#### 写数据寄存器 (0x0000\_0004)(读写)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** | 0x0000\_0004 | **属性** | 读写 |
| **位** | **含义** | | **复位值** |
| 31~8 | -- | | 0 |
| 7~0 | 此处写入需要写访问的总线的数据，数据长度为8bit; | |  |

#### 写命令寄存器 (0x0000\_0008)(读写)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** | 0x0000\_0008 | **属性** | 读写 |
| **位** | **含义** | | **复位值** |
| 31~0 | 分两次前后写入0xCCCC\_CCCC-->0x0000\_0000为一个组合发起一个写命令 | | 0 |

#### 读地址寄存器(0x0000\_000C)(读写)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** | 0x0000\_000C | **属性** | 读写 |
| **位** | **含义** | | **复位值** |
| 31~7 | -- | | 0 |
| 6~0 | 此处写入需要读访问的总线的地址，地址长度为7bit; | |  |

#### 读命令寄存器 (0x0000\_0010)(读写)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** | 0x0000\_0010 | **属性** | 读写 |
| **位** | **含义** | | **复位值** |
| 31~0 | 分两次前后写入0xDDDD\_DDDD-->0x0000\_0000为一个组合发起一个读命令 | | 0 |

#### 读数据寄存器 (0x0000\_0014)(只读)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** | 0x0000\_0014 | **属性** | 只读 |
| **位** | **含义** | | **复位值** |
| 31~8 | -- | | 0 |
| 7~0 | 总线的读数据，数据长度为8bit; | |  |

#### 总线状态寄存器(0x0000\_0018)(只读)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** | 0x0000\_0018 | **属性** | 只读 |
| **位** | **含义** | | **复位值** |
| 31~1 | -- | |  |
| 0 | 1: 总线忙 0: 总线\_空闲 | | 0 |

注：发起总线访问之前 或 要读取总线读数据时，必须确保对应的总线处于空闲状态

同时可以通过user\_wr\_vild user\_rd\_vild 信号为CPU提供读写完成的中断，提高访问的效率。

### C语言读写访问例程

C语言读写流程驱动，见路径

ADI\_SPI\_7\_8BIT\AXI4\_ADI\_SPI\_7\_8BIT\_1.0\drivers\AXI4\_ADI\_SPI\_7\_8BIT\_v1\_0\src\driver\_src参考此例程完成IP的操作。

## 在DSP + FPGA构架的应用

DSP+FPGA构架下，驱动例化均采用纯逻辑的例化方式，开发环境包含ISE和VIVADO, 交互机制的实现与AXI4 IP核的内部逻辑保持一致，只需要在《reg\_block寄存器标准文件》中定义AXI4 IP一样的寄存器即可，用以产生IP的标准用户接口的时序即可。

### IP寄存器描述

这些寄存器被定义为标准的交互逻辑寄存器，在《reg\_block寄存器标准文件》中定义并使用这些寄存器产生IP的标准用户接口的时序。 这些寄存器的具体偏移地址不进行强制规定，按照具体工程进行合理的分配即可

#### 写地址寄存器( )(读写)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** |  | **属性** | 读写 |
| **位** | **含义** | | **复位值** |
| x~7 | -- | | 0 |
| 6~0 | 此处写入需要写访问的总线的地址，地址长度为7bit; | |  |

#### 写数据寄存器 ()(读写)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** |  | **属性** | 读写 |
| **位** | **含义** | | **复位值** |
| x~8 | -- | | 0 |
| 7~0 | 此处写入需要写访问的总线的数据，数据长度为8bit; | |  |

#### 写命令寄存器 ()(读写)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** |  | **属性** | 读写 |
| **位** | **含义** | | **复位值** |
| x~0 | 分两次前后写入0xCCCC\_CCCC-->0x0000\_0000为一个组合发起一个写命令 | | 0 |

#### 读地址寄存器()(读写)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** |  | **属性** | 读写 |
| **位** | **含义** | | **复位值** |
| x~7 | -- | | 0 |
| 6~0 | 此处写入需要读访问的总线的地址，地址长度为7bit; | |  |

#### 读命令寄存器 ()(读写)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** |  | **属性** | 读写 |
| **位** | **含义** | | **复位值** |
| x~0 | 分两次前后写入0xDDDD\_DDDD-->0x0000\_0000为一个组合发起一个读命令 | | 0 |

#### 读数据寄存器 ()(只读)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** |  | **属性** | 只读 |
| **位** | **含义** | | **复位值** |
| x~8 | -- | | 0 |
| 7~0 | 总线的读数据，数据长度为8bit; | |  |

#### 总线状态寄存器( )(只读)

|  |  |  |  |
| --- | --- | --- | --- |
| **寄存器地址** |  | **属性** | 只读 |
| **位** | **含义** | | **复位值** |
| x~1 | -- | |  |
| 0 | 1: 总线忙 0: 总线\_空闲 | | 0 |

注：发起总线访问之前 或 要读取总线读数据时，必须确保对应的总线处于空闲状态

同时可以通过user\_wr\_vild user\_rd\_vild 信号为CPU提供读写完成的中断，提高访问的效率。

### C语言读写访问例程

C语言读写流程驱动，见路径

ADI\_SPI\_7\_8BIT\AXI4\_ADI\_SPI\_7\_8BIT\_1.0\drivers\AXI4\_ADI\_SPI\_7\_8BIT\_v1\_0\src\driver\_src参考此例程完成IP的操作。

参考此例程完成SJA1000总线读写操作。

注：在DSP+FPGA的构架中，总线忙状态查询时，建议在此函数中加入超时等待，防止外部物理总线出现问题时，程序进入死循环出现不可控的意外。