FPGA 片内 FIFO 读写例程

黑金动力社区 2020-03-13

1 实验简介

本实验将为大家介绍如何使用 FPGA 内部的 FIFO 以及程序对该 FIFO 的数据读写操作。

2 实验原理

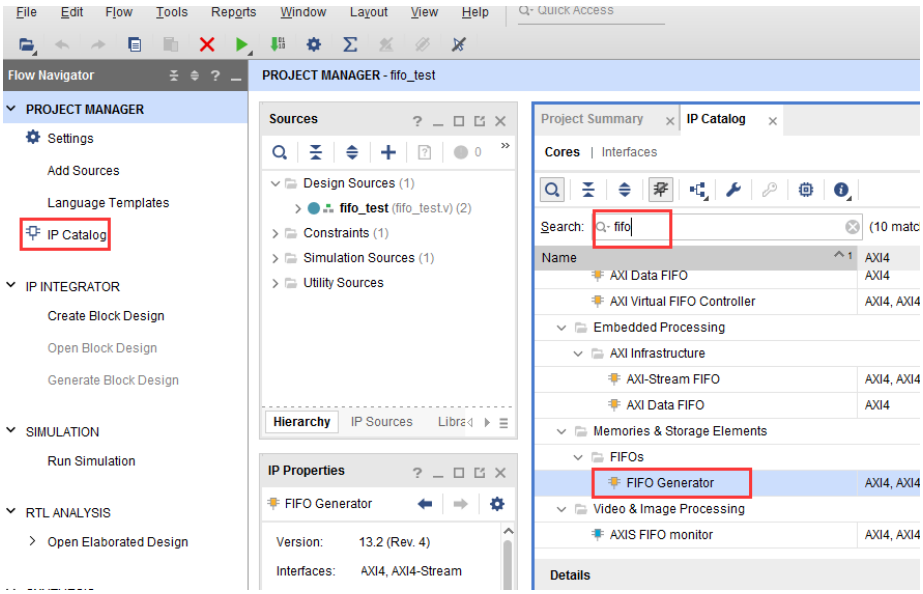
FIFO: First in, First out 代表先进的数据先出，后进的数据后出。Xilinx 在 VIVADO 里为我们已经提供了 FIFO 的 IP 核，我们只需通过 IP 核例化一个 FIFO，根据 FIFO 的读写时序来写入和读取 FIFO 中存储的数据。实验中会通过 VIVADO 集成的在想逻辑分析仪 ila，我们可以观察 FIFO 的读写时序和从 FIFO 中读取的数据。

3 程序设计

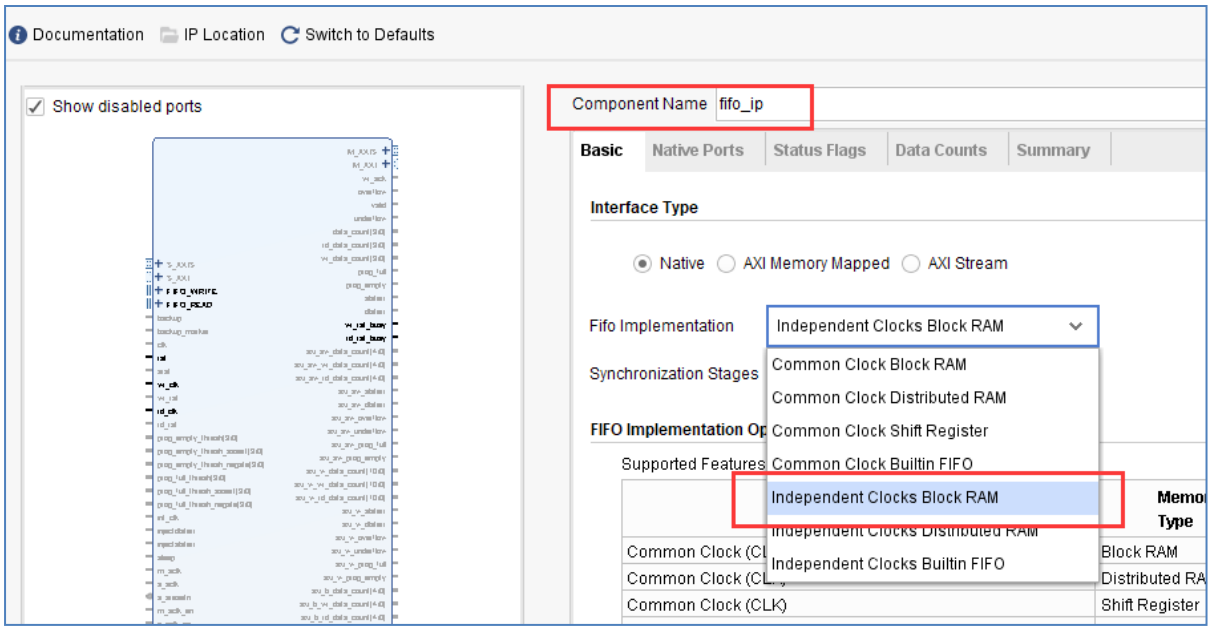
3.1 FIFO IP 的添加和配置

在添加 FIFO IP 之前先新建一个 fifo_test 的工程，然后在工程中添加 FIFO IP，方法如下：

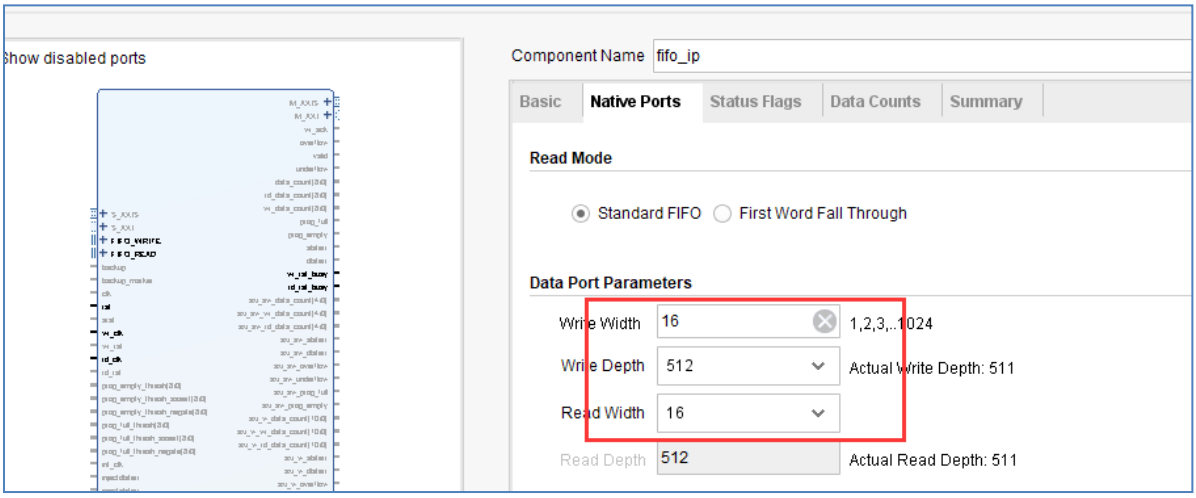
1. 点击下图中 IP Catalog，在右侧弹出的界面中搜索 fifo，找到 FIFO Generator,双击打开。



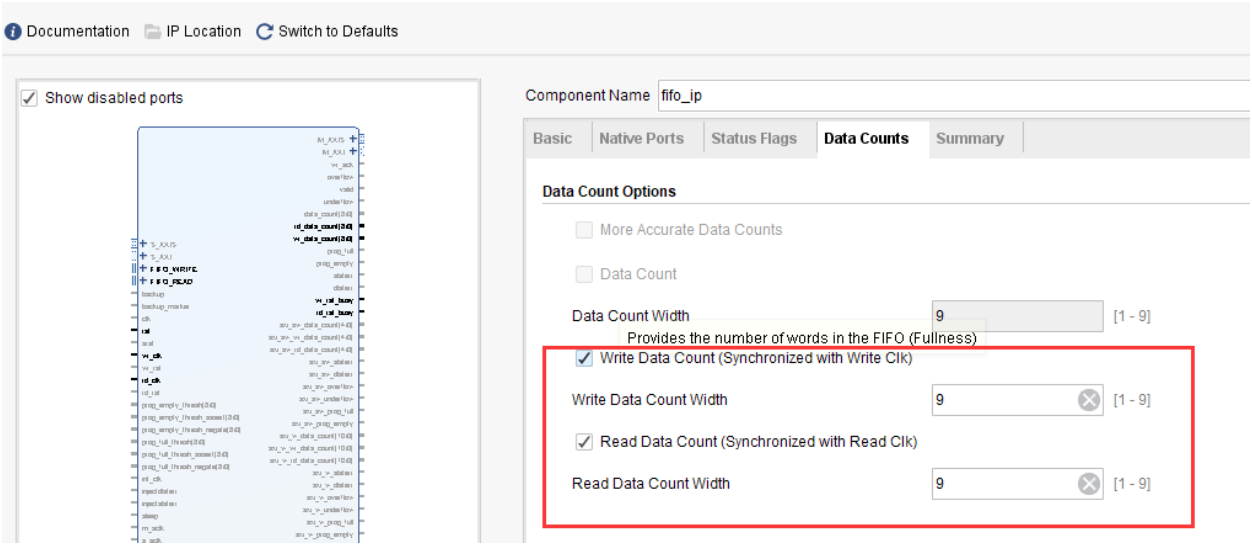
2. 弹出的配置页面中，这里可以选择读写时钟分开还是用同一个，一般来讲我们使用 FIFO 为了缓存数据，通常两边的时钟速度是不一样的。所以独立时钟是最常用的，我们这里选择“Independent Clocks Block RAM”，然后点击“Next”到下一个配置页面。



3. 切换到 Native Ports 栏目下，选择数据位宽 16；FIFO 深选择 512，实际使用大家根据需要自行设置就可以。其他配置默认即可。



4. 切换到 Data Counts 栏目下，使能 Write Data Count 和 Read Data Count，这样我们可以通过这两个值来看 FIFO 内部的数据多少。点击 OK,Generate 生成 FIFO IP。

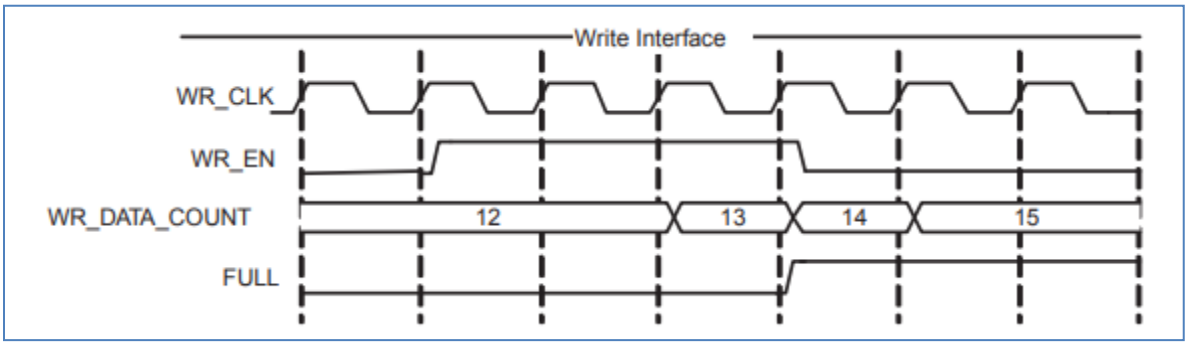


3.2 FIFO 的端口定义和时序

Simple Dual Port RAM 模块端口的说明如下：

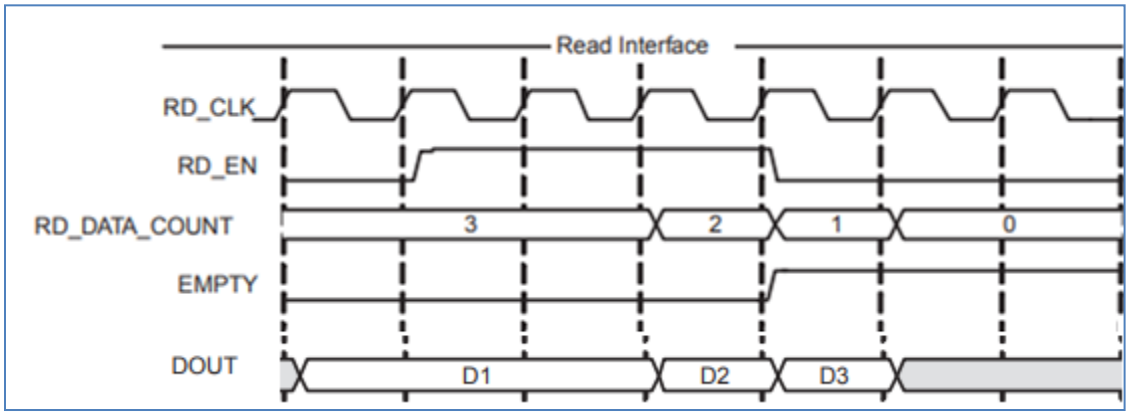
信号名称	方向	说明
rst	in	复位信号，高有效
wr_clk	in	写时钟输入
rd_clk	in	读时钟输入
din	in	写数据
wr_en	in	写使能，高有效
rd_en	in	读使能，高有效
dout	out	读数据
full	out	满信号
empty	out	空信号
rd_data_count	out	可读数据数量
wr_data_count	out	可写数据数量

FIFO 的数据写入和读出都是按时钟的上升沿操作的，WR_EN 信号为高时写入 FIFO 数据，WR_DATA_COUNT 的数据不会马上生效，比如下图第 3 个时钟开始写入 FIFO 数据，WR_DATA_COUNT 的值要在第 4 个时钟的时候才发生变化。下图为 FIFO 写的时序图。



FIFO 写时序

RD_EN 信号为高时读 FIFO 数据，RD_DATA_COUNT 的数据不会马上生效，比如下图第 3 个时钟开始写入 FIFO 数据，RD_DATA_COUNT 的值要在第 4 个时钟的时候才发生变化。下图为 FIFO 读的时序图。



FIFO 读时序

3.3 FIFO 测试程序编写

添加一个首先对 FIFO IP 的实例化，FIFO IP 的实例化及程序设计如下:

```
/*****************************************************************************
call xilinx fifo IP
*****************************************************************************/
fifo_ip fifo_ip_inst (
.rst            (~rst_n       ),   // input rst
.wr_clk         (clk          ),   // input wr_clk
.rd_clk         (clk          ),   // input rd_clk
.din            (w_data       ),   // input [15 : 0] din
.wr_en          (wr_en        ),   // input wr_en
.rd_en          (rd_en        ),   // input rd_en
.dout           (r_data       ),   // output [15 : 0] dout
.full           (full         ),   // output full
.empty          (empty        ),   // output empty
.rd_data_count  (rd_data_count),   // output [8 : 0] rd_data_count
.wr_data_count  (wr_data_count)    // output [8 : 0] wr_data_count
);
```

FIFO 写的过程会判断 FIFO 是否为空，如果是空，开始写数据，一直写到满为止。

```
/*****************************************************************************
Generating Write Fifo State Machine
*****************************************************************************/
always@(*)
begin
    case(write_state)
        W_IDLE:
            if(empty == 1'b1)              //FIFO is empty,  start writing FIFO
                next_write_state <= W_FIFO;
            else
                next_write_state <= W_IDLE;
        W_FIFO:
            if(full == 1'b1)               //FIFO is full
                next_write_state <= W_IDLE;
            else
                next_write_state <= W_FIFO;
        default:
            next_write_state <= W_IDLE;
    endcase
end

assign wr_en = (next_write_state == W_FIFO) ? 1'b1 : 1'b0;
```

FIFO 读的过程会判断 FIFO 是否为满，如果满，开始读数据，一直读到空为止。

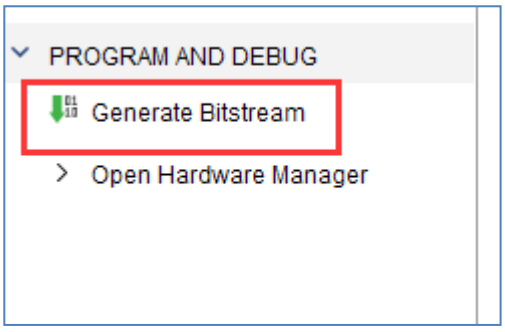
```
/*****************************************************************************
Generating read fifo State Machine
*****************************************************************************/
always@(*)
begin
    case(read_state)
        R_IDLE:
            if(full == 1'b1)              //FIFO is full, starting read FIFO
                next_read_state <= R_FIFO;
            else
                next_read_state <= R_IDLE;
        R_FIFO:
            if(empty == 1'b1)             //FIFO is  empty
                next_read_state <= R_IDLE;
            else
                next_read_state <= R_FIFO;
        default:
            next_read_state <= R_IDLE;
    endcase
end

assign rd_en = (next_read_state == R_FIFO) ? 1'b1 : 1'b0;
/*---------------------------------------------------------------------------*/
```

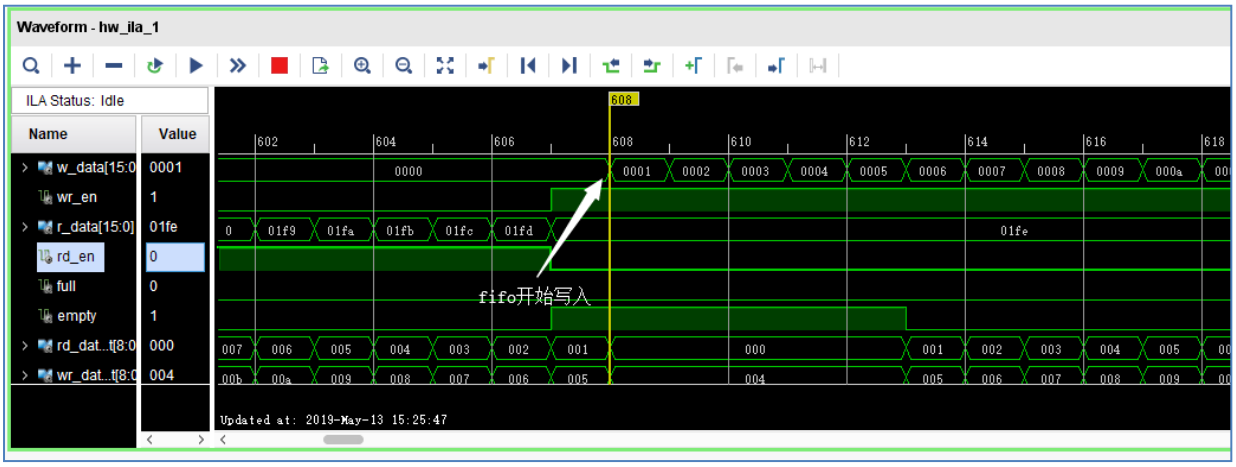
为了能实时看到 FIFO 数据的读写，我们这里添加了 ila 工具来观察 FIFO 的数据信号和控制信号。关于如何生成 ila 大家请参考"I2C 接口 EEPROM 实验.pdf"教程。

4 实验现象

生成 bit 文件并下载到 FPGA。我们通过 ila 来观察 FIFO 的读写数据。



在 Waveform 的窗口我们可以看到当 empty 变高时，FIFO 开始写入数据（0~1FE）。



在 FIFO 的 full 信号变高时，FIFO 停止写入，开始读出数据（0~1FE）。