Appplication Note

AN05320101

FIFO 应用笔记

V1.00 Date: 2007/06/24

产品应用笔记

文件信息

类别	内容
关键词	FIFO、FULL、控制器
摘要	介绍如何使用 ProASIC3/E 三种 FIFO，包括内部结构、时序波形以及简单的实例

修订历史

版本	日期	原因
V1.00	2007/06/24	创建文档。

目 录

1. 适用范围

此应用笔记适用于 Actel Flash 结构的 ProASIC3/E 系列的 FPGA，适用 Libero 7.3 和 Libero 8.0 IDE 开发环境。

2. 概述

随着设计需求地增加，对存储器的要求也越来越高，很多地设计都需要使用 FIFO，Actel 的 FPGA 在这方面有独特的设计，为了迎合不同的客户需求，Actel 设计了三种类型的 FIFO，分别是：内嵌硬件 FIFO 控制器的同步 FIFO、带有存储单元的软控制器的 FIFO、不带存储单元的软控制器的 FIFO，第一种不占用逻辑资源，和 RAM 一起使用；第二种控制器用逻辑资源搭建，存储器用内部的 RAM；第三种是独立的控制器，用逻辑资源搭建，不带有存储单元，但有读写信号和地址信号输出。下面将会介绍这些 FIFO 的使用。

3. FIFO 的特点

3.1 FIFO 内部结构

这里主要介绍内部带有硬件 FIFO 控制器的 FIFO 的结构，这是 Actel FPGA 的一大特点，可以节省很多的资源，它和内部的 RAM 一起配合使用，每个 4K 的 RAM 块内部都带有 FIFO 控制器。当只作为 RAM 使用时，控制器被旁路；当用作 FIFO 使用时，控制器被使能，并产生一些标志信号，例如：FULL、EMPTY、AFULL、AEMPTY 等，图 1 是带有控制器的 RAM 块的内部结构。

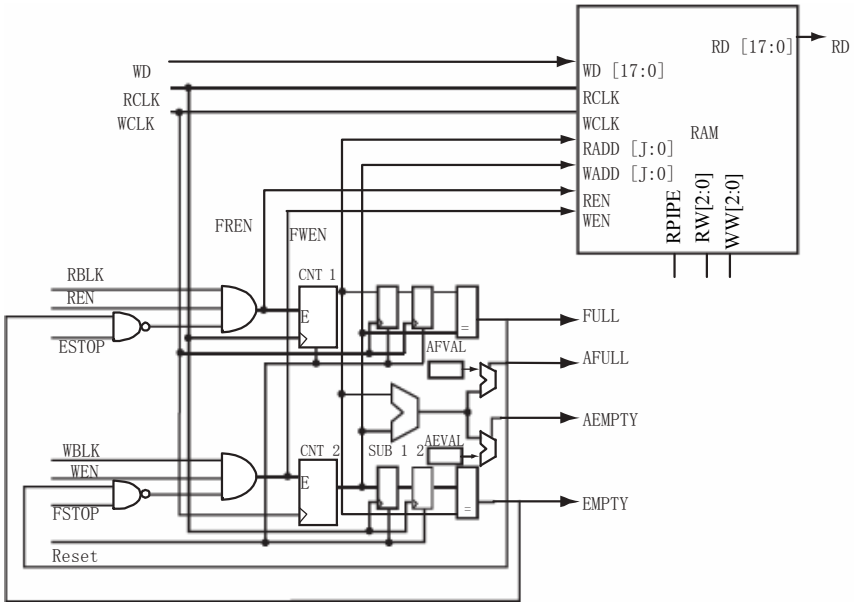


图 1 内嵌 FIFO 控制器的 RAM 块内部结构

FIFO 中的 RAM 块和一般的 RAM 的使用一样，只能是配置成 4,096×1，2048×2，1,024×4，512×9，256×18 这 5 种模式，虽然软件里可以设置各种各样的宽度和深度比，但是最终是用这几种模式中的一种来实现的，它是由控制器内部的 WW 和 RW 信号决定的，具体见表 1 所示。

表 1 WW 和 RW 配置宽度深度比

WW2, WW1, WW0	RW2, RW1, RW0	DxW
000	000	4k×1
001	001	2k×2
010	010	1k×4
011	011	512×9
100	100	256×18
101, 110, 111	101, 110, 111	保留

FIFO 的存储模块最高可以配置成 18 位的模式，所以在 FIFO 控制器的接口上预留有 18 位的数据接口，当使用的数据宽度达不到 18 位时，剩余的读写数据线必须接地。但是如果

是通过 SmartGen 软件生成的 FIFO，这些工作是 SmartGen 软件来自动完成；如果是直接例化一个 FIFO，这些地方是我们必须手动设置，数据总线的使用情况如表 2 所示。

表 2 不同宽度深度比的数据信号的使用情况

DxW	不使用的 WD/RD
4k×1	WD[17:1], RD[17:1]
2k×2	WD[17:2], RD[17:2]
1k×4	WD[17:4], RD[17:4]
512×9	WD[17:9], RD[17:9]
256×18	-

3.2 FIFO 控制器的内部信号

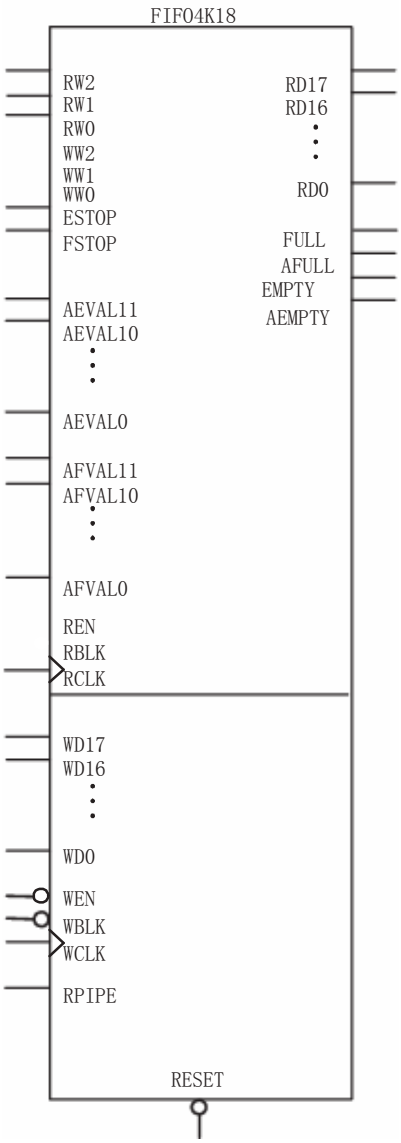


图 2 FIFO 控制器宏模块

这是 FIFO 宏模块的内部接口信号，SmartGen 软件中的设置参数也是最终影响这些信号来完成 FIFO 控制器的配置，我们在 SmartGen 软件生成时并不关心这些信号，但是用 HDL 例化的时候，我们必须非常的清楚这些信号的含义，如表 3 所示列出这些信号的含义，这些信号的含义仅仅针对于内嵌硬件控制器的 FIFO 来说的，其他两种会有一定的区别。

表 3 FIFO 的控制信号

信号名称	I/O 类型	含义
WW[2:0]	I	配置 FIFO 的写宽度和深度比，具体见表 1
RW[2:0]	I	配置 FIFO 的读宽度和深度比，具体见表 1
WBLK	I	写端口使能信号
RBLK	I	读端口使能信号
WEN	I	写使能信号，默认低有效，可以设置为高
REN	I	读使能信号，默认低有效，可以设置为高
WCLK	I	同步写数据时钟，可设置为上升或下降沿
RCLK	I	同步读数据时钟，可设置为上升或下降沿
RPIPE	I	指定流水线的读，为低为非流水线的读，数据在一个时钟周期内出现输出口上，高为流水线的读，数据出现在下个读时钟周期上
RESET	I	复位信号，低有效，将输出信号复位到零
WD	I	输入数据总线，18 位宽，但当配置成不同模式的 RAM 块时，这 18 位并非全部用到，不使用的信号必须接地
RD	O	输出数据总线，描述如上。
ESTOP	I	当 FIFO 为空时，该信号可以用来禁止读地址计数器继续计数（高有效），否则读地址计数器将从零开始
FSTOP	I	当FIFO 为满时，该信号可以来禁止写地址计数器继续计数（高有效），否则写地址计数器将从零开始
FULL	O	当 FIFO 已满不能再写入数据时，该标志置高 ，保持两个时钟周期时间，因为它需要和写地址进行比较
EMPTY	O	当 FIFO 已空不能再读取数据时，该标志置高，保持两个时钟周期时间，因为它需要和读地址进行比较
AFULL	O	近满标志，这和 AFVAL 的阀值有关，在写操作时，FIFO 存放的数据到达了 AFVAL 设定的值时，该标志变成高电平
AEMPTY	O	近空标志，这和 AEVAL 的阀值有关，在读操作时，FIFO 存放的数据到达了 AEVAL 设定的值时，该标志变成高电平
AFVAL	I	指定近满标志的阀值，12 位可设置，可寻址 4096bits
AEVAL	I	指定近空标志的阀值，12 位可设置，可寻址 4096bits

3.3 时序波形图

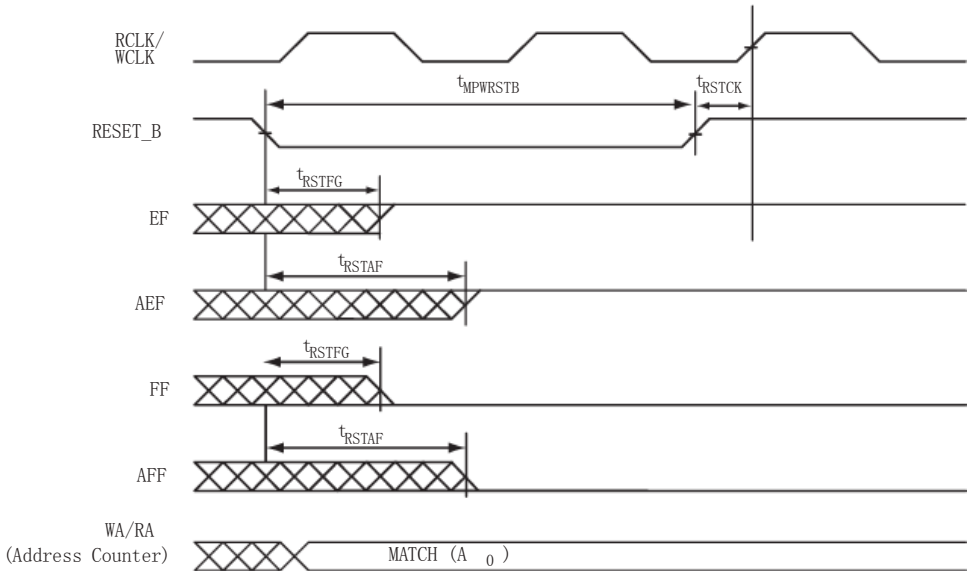


图 3 FIFO 复位

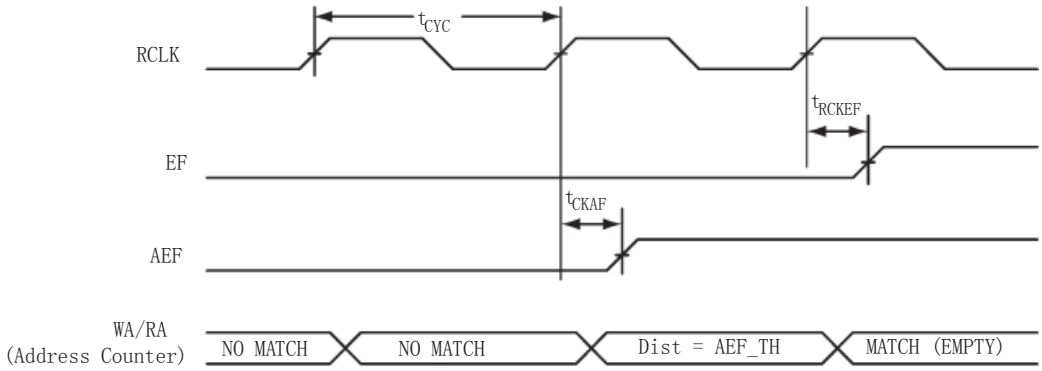


图 4 FIFO 空标志和近空标志

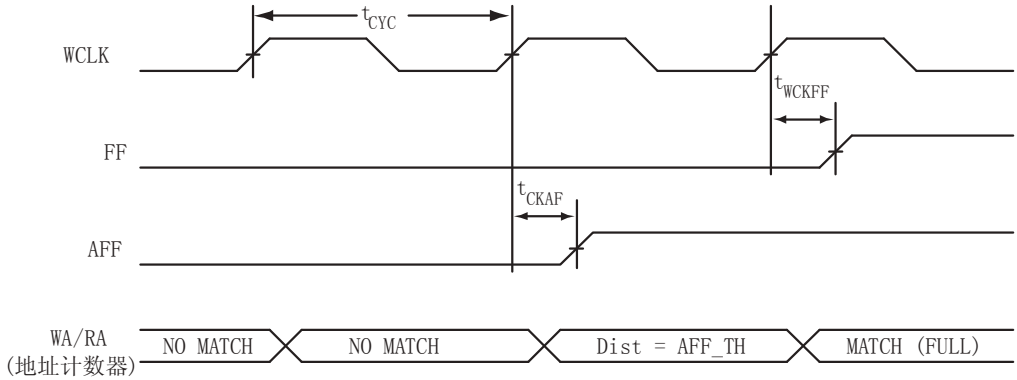


图 5 FIFO FULL 和 AFULL 标志

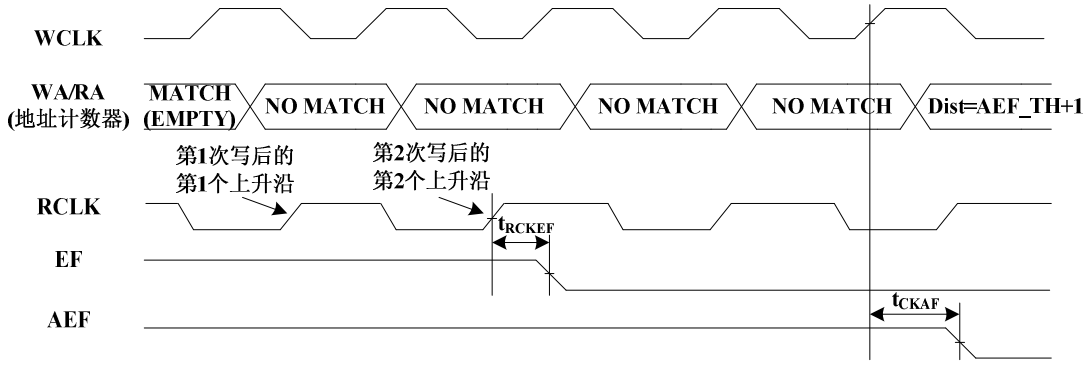


图 6 EMPTY 标志和 AEMPTY 标志变为无效

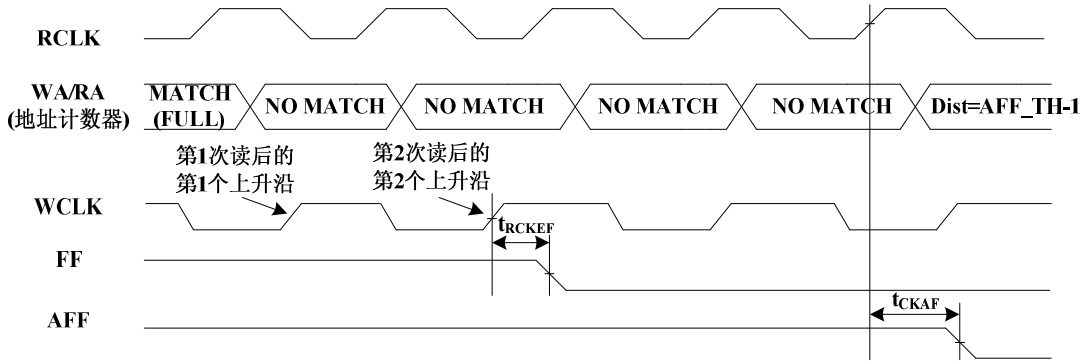


图 7 FULL 和 AFULL 变为无效

表 4 FIFO（（商业环境的条件：T_J=70℃， V_{CC}=1.425V）

参数	描述	–2	–1	标准值	单位
t_{ENS}	REN_B,WEN_B 建立时间	0.91	1.04	1.22	ns
t_{ENH}	REN_B, WEN_B 保持时间	0.00	0.00	0.00	ns
t_{BKS}	BLK_B 建立时间	0.25	0.29	0.34	ns
t_{BKH}	BLK_B 保持时间	0.00	0.00	0.00	ns
t_{DS}	输入数据（DI）建立时间	0.18	0.21	0.25	ns
t_{DH}	输入数据（DI）保持时间	0.00	0.00	0.00	ns
t_{CKQ1}	从时钟高电平到 DO 数据有效的时间（直通)	2.36	2.68	3.15	ns
t_{CKQ2}	从时钟高电平到 DO 数据有效的时间（流水线)	0.89	1.02	1.20	ns
t_{RCKEF}	从 RCLK 高电平到空（Empty）标志变有效的时间	1.72	1.96	2.30	ns
t_{WCKFF}	从 WCLK 高电平到满（Full）标志变有效的时间	1.63	1.86	2.18	ns
t_{CKAF}	从时钟高电平到近空（Almost-Empty）/近满（Almost Full）标志变有效的时间	3.72	4.24	4.99	ns
t_{RSTFG}	从 RESET_B 低电平到空/满标志变有效的时间	1.69	1.93	2.27	ns
t_{RSTAF}	从 RESET_B 低电平到近空/近满标志变有效的时间	3.66	4.17	4.90	ns
t_{RSTBQ}	从 RESET_B 低电平有效到 DO 上数据输出的时间（直通）	0.92	1.05	1.23	ns
	从 RESET_B 低电平有效到 DO 上数据输出的时间(流水线）	0.92	1.05	1.23	ns
$t_{REMRSTB}$	RESET_B 撤销	0.29	0.33	0.38	ns
$t_{RECRSTB}$	RESET_B 恢复	1.50	1.71	2.01	ns
$t_{MPWRSTB}$	RESET_B 的最小脉宽	0.30	0.30	0.30	ns
t_{CYC}	时钟周期	2.79	2.79	2.79	ns

4. 三种 FIFO 应用实例

4.1 应用场合

Actel FPGA 的 FIFO 有三种类型，可以适用于不同的应用场合：

(1) 同步的 FIFO（内嵌硬件 FIFO 控制器）

它不需要占用逻辑资源，只用内嵌 FIFO 控制器的 RAM 块实现，它的数据读写宽度必须是一样的，由同一个时钟驱动，适合逻辑资源紧张、需要同步设计、性能要求高的场合。

(2) 软 FIFO 控制器（带存储单元）

它的 FIFO 控制器是由逻辑单元搭建，所以会占用一定的资源，存储单元是由内部的 4KRAM 块实现，数据的读写宽度可以不同，适合读写数据宽度不一、灵活度高、标志信息多的场合。

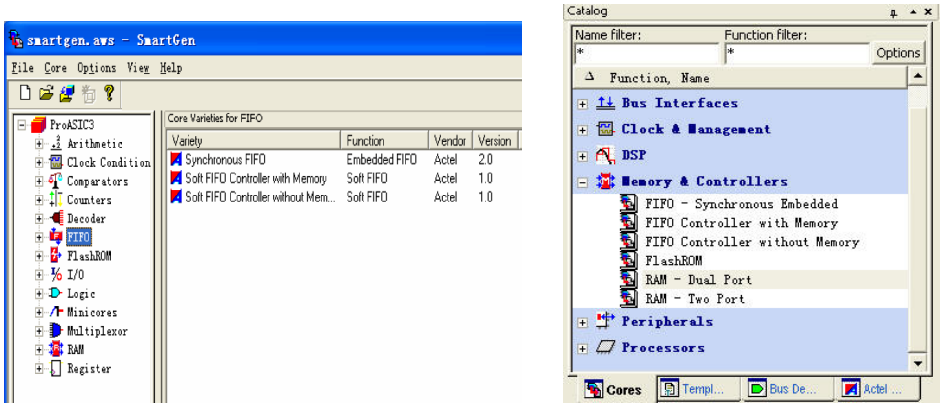
(3) 软 FIFO 控制器（不带存储单元）

它只有一个 FIFO 的控制器，不带有存储单元，不过它将存储器的地址和读写信号都引出，可以外接或者内部的存储器，而控制信号由 FIFO 控制器产生，适合于需要独立的控制器、自定义存储单元、灵活度高的场合。

4.2 同步的 FIFO（内嵌硬件控制器）

(1) 打开 SmartGen 软件，然后选择 FIFO，双击图 8 中的 Synchronous FIFO；

注意：在 Libero 7.3 中是独立调用的 SmartGen 软件，而 Libero 8.0 将该软件嵌在 Libreo 界面里面，可以直接调用里面的模块，图 8 就是两者类似的界面。



Libero 7.3　　Libero 8.0

图 8　SmartGen 软件中的三种 FIFO

(2) 配置同步的 FIFO

如图 所示，是同步 FIFO 的配置界面，这里的设置将最终影响表 3 中所示的 FIFO 内部信号，这些配置参数说明见表 5 所示。当设置完参数，点击“Generate”生成。

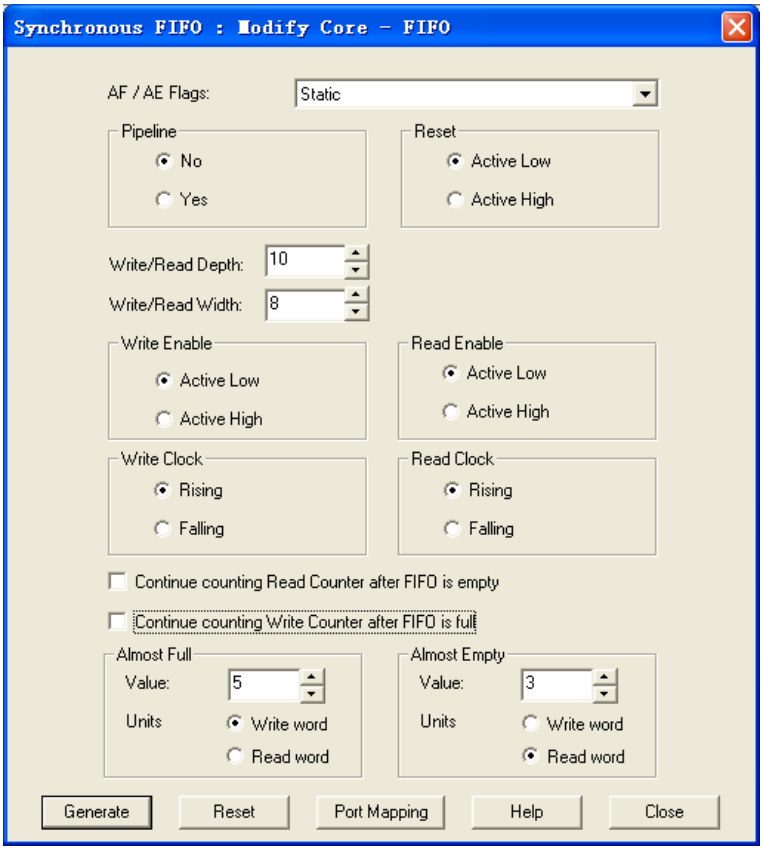


图 9　同步 FIFO 的配置界面

表 5　同步 FIFO 配置参数

配置参数	描述
AF/AE Flags	可以选择 Static、Dynamic、NoFlags 模式，Static 模式是指 AFVAL、AEVAL 的值都是通过软件设定好，不能在程序中修改；Dynamic 是指 AFVAL、AEVAL 信号引到顶层，可以动态的修改；NoFlags 指禁止使用 AFull 和 AEmpty 标志信号。
Pipeline	可以选择是否进行流水线的读，控制内部的 RPIPE 信号
Reset	选择复位的高或低电平，控制内部得 RESET 信号
Write/Read Depth	数据的读写深度设置，范围是 1~4096
Write/Read Width	数据的读写宽度设置，范围是 1~576，
Write Enable	选择写使能的电平，控制内部的 WEN 信号
Read Enable	选择读使能的电平，控制内部的 REN 信号
Write Clock	选择写时钟的边沿，控制内部的 WCLK 系信号
Read Clock	选择读时钟的边沿，控制内部的 RCLK 系信号
Continue counting Read Counting after FIFO is empty	这个选项可以设定当 FIFO 为空的时候，是否停止读地址计数器，控制内部的 ESTOP 信号
Continue counting Write Counting after FIFO is full	这个选项可以设定当 FIFO 为满的时候，是否停止写数器，控制内部的 FSTOP 信号

续表 4

Almost Full	Value：设定 AFULL 信号比较的阀值，当读地址计数器和写地址计数器的差值大于或等于这个值时就会把 AFULL 标志置有效； Units：选择指示读还是写
Almost Empty	Value：设定 AEMPTY 信号比较的阀值，当读地址计数器和写地址计数器的差值小于或等于这个值时就会把 AEMPTY 标志置有效； Units：选择指示读还是写

(3) 回到 Libero，查看 HDL 代码

从生成的 HDL 代码中可以看出，其实 SmartGen 软件所做的工作就是把设置的参数与 FIFO 控制器的内部信号进行一个连接，然后例化一个 FIFO4K×18 的 FIFO 块，这些 FIFO 的参数就是表 3 中的参数，如果我们理解这些参数的含义，也可以直接进行例化而不需要 SmartGen 的帮助，用户可以对比这些参数的关系进行理解。

```
`timescale 1 ns/100 ps
module FIFO(DATA,Q,WE,RE,WCLOCK,RCLOCK,FULL,EMPTY,RESET,AEMPTY,
        AFULL);
input [7:0] DATA;
output [7:0] Q;
input  WE, RE, WCLOCK, RCLOCK;
output  FULL, EMPTY;
input  RESET;
output  AEMPTY, AFULL;

    wire WEAP, VCC, GND;

    VCC VCC_1_net(.Y(VCC));
    GND GND_1_net(.Y(GND));
    INV REBUBBLEA(.A(RE), .Y(WEAP));
    FIFO4K18 FIFOBLOCK0(.AEVAL11(GND), .AEVAL10(GND), .AEVAL9(GND)
        , .AEVAL8(GND), .AEVAL7(GND), .AEVAL6(GND), .AEVAL5(GND),
        .AEVAL4(VCC), .AEVAL3(VCC), .AEVAL2(GND), .AEVAL1(GND),
        .AEVAL0(GND), .AFVAL11(GND), .AFVAL10(GND), .AFVAL9(GND),
        .AFVAL8(GND), .AFVAL7(GND), .AFVAL6(GND), .AFVAL5(VCC),
        .AFVAL4(GND), .AFVAL3(VCC), .AFVAL2(GND), .AFVAL1(GND),
        .AFVAL0(GND), .WD17(GND), .WD16(GND), .WD15(GND), .WD14(
        GND), .WD13(GND), .WD12(GND), .WD11(GND), .WD10(GND),
        .WD9(GND), .WD8(GND), .WD7(DATA[7]), .WD6(DATA[6]), .WD5(
        DATA[5]), .WD4(DATA[4]), .WD3(DATA[3]), .WD2(DATA[2]),
        .WD1(DATA[1]), .WD0(DATA[0]), .WW0(VCC), .WW1(VCC), .WW2(
        GND), .RW0(VCC), .RW1(VCC), .RW2(GND), .RPIPE(GND), .WEN(
        WE), .REN(WEAP), .WBLK(GND), .RBLK(GND), .WCLK(WCLOCK),
        .RCLK(RCLOCK), .RESET(RESET), .ESTOP(VCC), .FSTOP(VCC),
        .RD17(), .RD16(), .RD15(), .RD14(), .RD13(), .RD12(),
        .RD11(), .RD10(), .RD9(), .RD8(), .RD7(Q[7]), .RD6(Q[6]),
        .RD5(Q[5]), .RD4(Q[4]), .RD3(Q[3]), .RD2(Q[2]), .RD1(Q[1])
        , .RD0(Q[0]), .FULL(FULL), .AFULL(AFULL), .EMPTY(EMPTY),
        .AEMPTY(AEMPTY));

endmodule
```

图 10 同步 FIFO 的 HDL 代码

还是以这个例子说明一下，这些信号的连接关系，大家先看图 9，我们设置的“Almost Full”和“Almost Ampty”的值分别为 5 和 3，而且是 8 位的数据宽度，分别代表的阀值是 5*8=40 位和 3*8=24 位，因此 AFVAL 和 AEVAL 设置为 101000 和 11000；数据宽度为 8 位，WD 的低 8 位有效，其它接地；还是由于数据宽度为 8 位，使得 FIFO 配置成 512×9 的模式，所以 WW[2:0]=011,RW[2:0]=011，看表 2 可以找到对应关系；由于图 没有将“Continue counting Read Counting after FIFO is empty”和“Continue counting Write Counting after FIFO is full”打勾，所有 ESTOP 和 FSTOP 都接高电平（VCC），表明当 FIFO 满或者空的时候地址计数器都停止计数；其他标志相对简单，大家自行分析。

(4) 设置仿真测试文件

点击 WaverFormer 软件，编辑仿真测试文件，设置好波形，然后保存为 TestBench 文件，如何操作见《Libero 快速入门》文档。

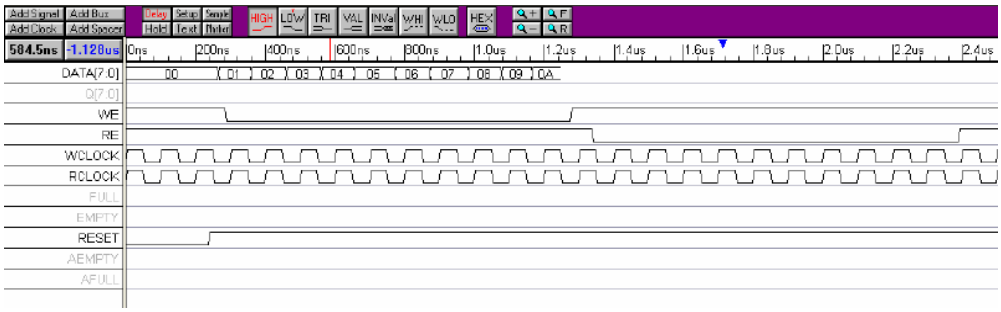


图 11　仿真测试文件生成

(5)　ModelSim 仿真

下面将列举各种不同的情况下仿真时序波形，目的是在实际操作的过程中能够明白为什么会出现这样的信号，我将详细解释图 12 的标志变化情况，其它简述注意点，大家可以自行分析。

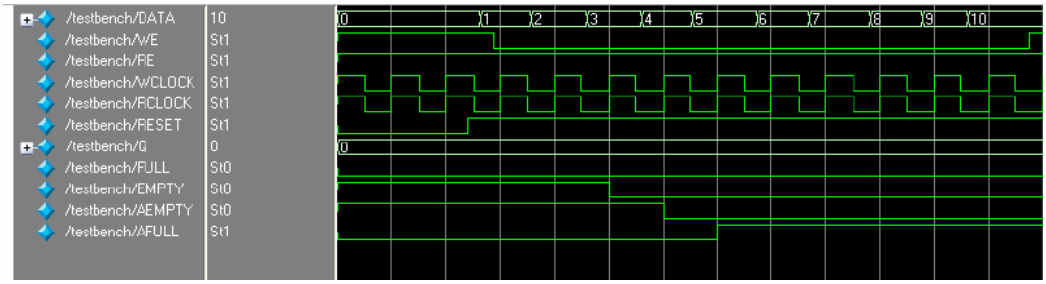


图 12　写数据过程

- FULL：只有当把 4K 的 RAM 写满时，该信号才会变高（高电平有效），因为仿真的时候没有写满 4K，所以一直保持无效状态；
- EMPTY：一开始，数据为空，该信号高（高电平有效），即使有写信号和写时钟在操作，但是该信号仍将保持两个时钟的周期才会变成无效，见表 3 说明，这是因为它需要时间来比较读地址；
- AEMPTY：在配置 FIFO 的时候我们设置的阀值 AEVAL 是 3，所以当写地址计数器和读地址计数器的差值小于或等于 3 时，该信号就会置高（高电平有效），所以这里它保持了三个周期时间；
- AFULL：在配置 FIFO 的时候我们设置的阀值 AFVAL 是 5，所以当写地址计数器和读地址计数器的差值大于或者等于 5 时，该信号就会变得高（高电平有效），所以这里在写信号和写时钟有效后 5 个周期，该信号变为高电平。

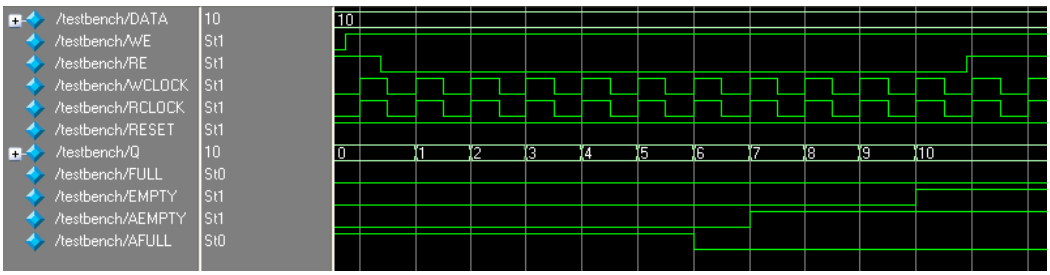


图 13　读数据过程

如图 13 所示是读取数据的过程，由于没写满，FULL 将一直无效；第 10 个上升沿后变空，EMPTY 变有效；AEMPTY 在第 7 个时钟的时候，读地址计数器和写地址计数器的差值为 3，该信号变有效；AFULL 在第 6 个时钟的时候，写地址计数器和读地址计数器差值小于 5，该信号变有效。

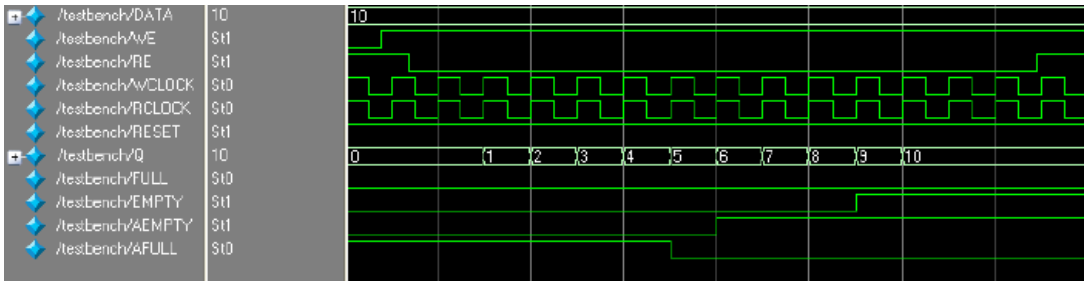


图 14 流水线读模式

重新回到 SmartGen 中 FIFO 配置界面，将 Pipeline 选择为“yes”，再生成一次，然后重新按照(3)、(4)、(5)步骤重新做一次，仿真的波形如图 14 所示，采用流水线读出的数据将延时一个周期的时间出现在数据线 Q 上，但是标志信号的位置都没有变，再注意一点，如果没有选择“Continue counting Read Counting after FIFO is empty”，当 FIFO 为空的时候，读地址计数器不会再计数，所以读出的数据一直是同一个数据，如上图所示。

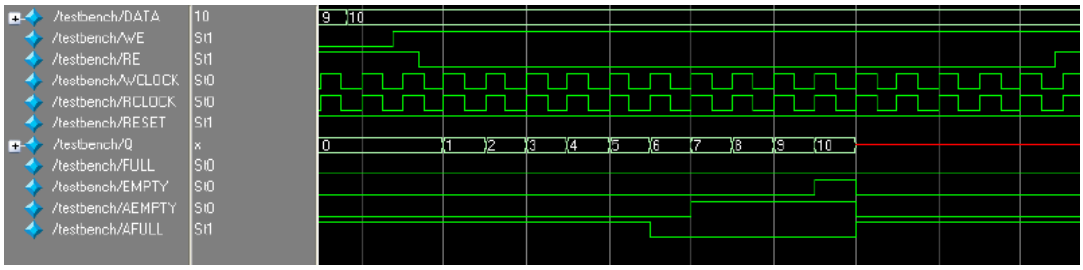


图 15 空之后继续可读数据模式

如图 15 所示是选择“Continue counting Read Counting after FIFO is empty”，不选择流水线，所以读地址计数器继续计数，由于 FIFO 采用的模式是 512×9，所以从 512 开始递减，而在这些地址里没有数据写入，出现了不定态。 当选择“Continue counting Write Counting after FIFO is Full”时，只有当计满 512 个字节的时候，如果再写入数据将会覆盖原来的值，否则这 512 个数据一直有效，并可以读出，这里就不附图，大家可以自己尝试。

4.3 软 FIFO 控制器（带存储单元）

(1) 打开 SmartGen 软件，然后选择 FIFO，双击图 8 中“Soft FIFO Controller with Memory”。

(2) 配置软 FIFO 控制器

如图 16 所示，是带存储单元的软 FIFO 控制器配置界面，这些配置参数说明见表 6 所示。当设置完参数，点击“Generate”生成。

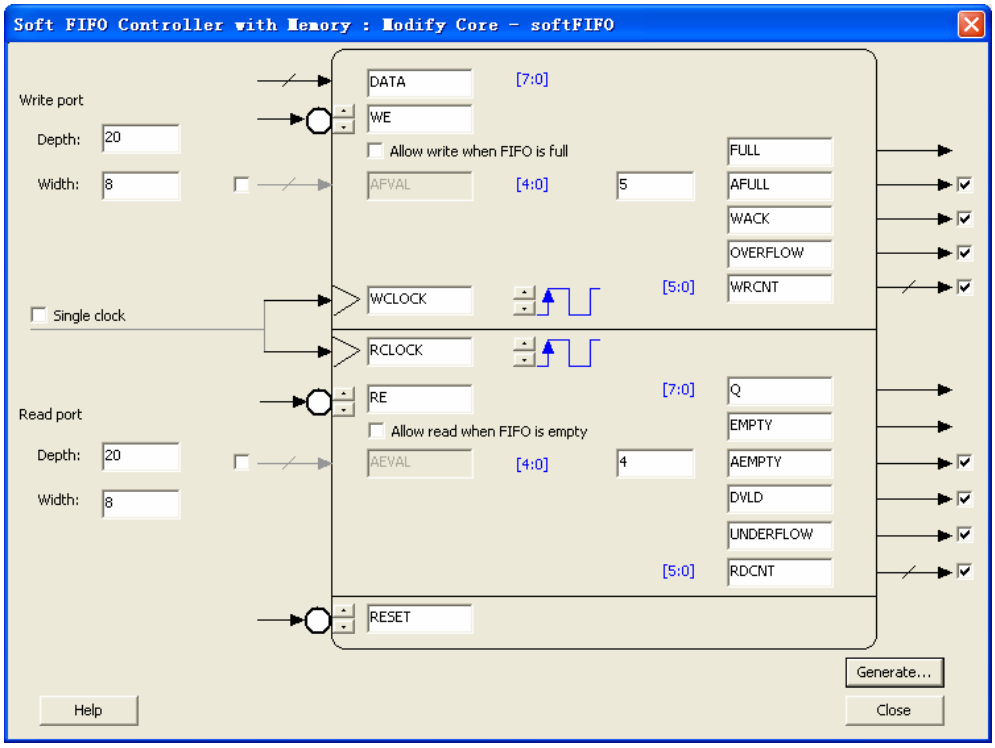


图 16　带存储单元的软 FIFO 配置界面

表 6　带存储单元的软 FIFO 配置参数

配置参数	描述
Write port	写数据口的宽度和深度，可以配置独立的读写数据宽度
Read port	读数据口的宽度和深度，可以配置独立的读写数据宽度
Single clock	可以设置同一个读写时钟，也可以独立的读写时钟
Allow Write when FIFO is Full	选择当满的时候是否允许继续写入，当选择此项时，OVERFLOW 无效
Allow read when FIFO is empty	选择当空的时候是否允许继续读出，当选择此项时，UNDERFLOW 无效
AFVAL	可以选择将近满的阀值设置引到顶层动态修改，也可以在后面独立设置
AEVAL	可以选择将近空的阀值设置引到顶层动态修改，也可以在后面独立设置
FULL	满标志
EMPTY	空标志
AFULL	近满标志
AEMPTY	近空标志
WACK	当数据写入到 FIFO 时，该标志高有效，表明数据写入有效
OVERFLOW	写数据失败，当写入的数据超过了 FIFO 的容量，标志变高有效
UNOVERLOW	读数据失败，当 FIFO 为空，继续读时，该标志变高有效
WRCNT	写地址计数器，每次写入数据时，计数器就会加 1，这个计数器的位数根据设定的数据深度软件自动生成，最大计数范围是深度的两倍。
Q	数据的输出信号线
DVLD	当读数据成功的时候，该标志被置高（DVLD 高电平有效）

续表 5

RDCNT	读地址计数器，读数据是递减的过程，写数据是递加的过程，这个计数器的位数根据设定的数据深度，软件自动生成，最大计数范围是深度的两倍。
WRCNT	写地址计数器，读数据是递减的过程 ，写数据时递加的过程，这个计数器的位数根据设定的数据深度，软件自动生成，最大计数范围是深度的两倍。

(3) 回到 Libero，查看 HDL 代码

由于 FIFO 控制器是由逻辑单元搭建的，所以代码比较的多，这里只显示使用 RAM 的一部分，里面的存储器模块还是调用 RAM4K9 的模块，图 17 就是 RAM4K9 的例化代码。

```
140     RAM4K9 RAM4K9_QXI_7_inst(.ADDRA11(GND), .ADDRA10(GND),
141         .ADDRA9(GND), .ADDRA8(GND), .ADDRA7(GND), .ADDRA6(GND),
142         .ADDRA5(GND), .ADDRA4(MEM_WADDR_4_net), .ADDRA3(
143         MEM_WADDR_3_net), .ADDRA2(MEM_WADDR_2_net), .ADDRA1(
144         MEM_WADDR_1_net), .ADDRA0(MEM_WADDR_0_net), .ADDRB11(GND),
145         .ADDRB10(GND), .ADDRB9(GND), .ADDRB8(GND), .ADDRB7(GND),
146         .ADDRB6(GND), .ADDRB5(GND), .ADDRB4(MEM_RADDR_4_net),
147         .ADDRB3(MEM_RADDR_3_net), .ADDRB2(MEM_RADDR_2_net),
148         .ADDRB1(MEM_RADDR_1_net), .ADDRB0(MEM_RADDR_0_net),
149         .DINA8(GND), .DINA7(DATA[7]), .DINA6(DATA[6]), .DINA5(
150         DATA[5]), .DINA4(DATA[4]), .DINA3(DATA[3]), .DINA2(
151         DATA[2]), .DINA1(DATA[1]), .DINA0(DATA[0]), .DINB8(GND),
152         .DINB7(GND), .DINB6(GND), .DINB5(GND), .DINB4(GND),
153         .DINB3(GND), .DINB2(GND), .DINB1(GND), .DINB0(GND),
154         .WIDTHA0(VCC), .WIDTHA1(VCC), .WIDTHB0(VCC), .WIDTHB1(VCC)
155         , .PIPEA(GND), .PIPEB(GND), .WMODEA(GND), .WMODEB(GND),
156         .BLKA(MEMWENEG), .BLKB(MEMRENEG), .WENA(GND), .WENB(VCC),
157         .CLKA(WCLOCK), .CLKB(RCLOCK), .RESET(RESET), .DOUTA8(),
158         .DOUTA7(RAM4K9_QXI_7__DOUTA7), .DOUTA6(
159         RAM4K9_QXI_7__DOUTA6), .DOUTA5(RAM4K9_QXI_7__DOUTA5),
160         .DOUTA4(RAM4K9_QXI_7__DOUTA4), .DOUTA3(
161         RAM4K9_QXI_7__DOUTA3), .DOUTA2(RAM4K9_QXI_7__DOUTA2),
162         .DOUTA1(RAM4K9_QXI_7__DOUTA1), .DOUTA0(
163         RAM4K9_QXI_7__DOUTA0), .DOUTB8(), .DOUTB7(QXI_7_net),
164         .DOUTB6(QXI_6_net), .DOUTB5(QXI_5_net), .DOUTB4(QXI_4_net)
165         , .DOUTB3(QXI_3_net), .DOUTB2(QXI_2_net), .DOUTB1(
166         QXI_1_net), .DOUTB0(QXI_0_net));
```

图 17 HDL 代码

(4) 设置仿真测试文件

点击 WaverFormer 软件，编辑仿真测试文件，设置好波形，然后保存为 TestBench 文件，如何操作见《Libero 快速入门》文档。

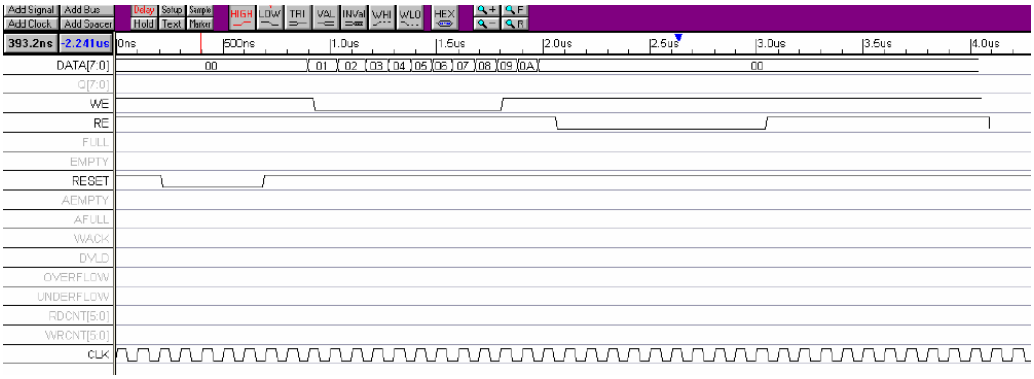


图 18 仿真测试文件生成

(5) ModelSim 仿真

下面将列举各种不同的情况下仿真时序波形，这里的时序波形和带硬件控制器的 FIFO 有所区别，具体在下面介绍，也同样会列出不同的情况下的时序，注意这里的数据读出都带有一级流水线。

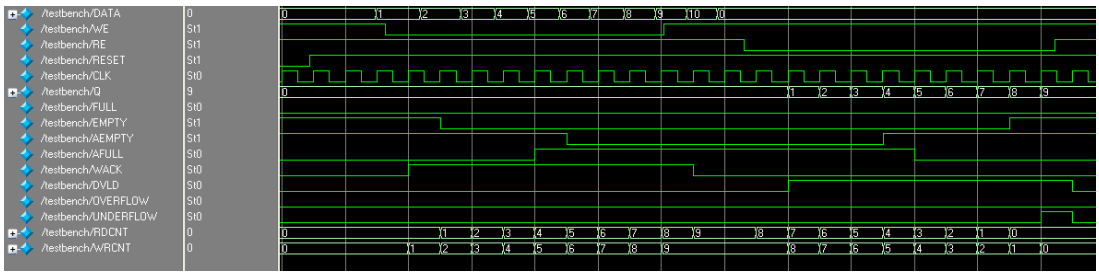


图 19　读写数据过程

- FULL：它和写时钟同步，由于未满（20 个数），一直为低；
- EMPTY：它和读时钟同步，从图 19 看出，它将在 RDCNT 开始计数的时候低无效；
- AEMPTY：近空标志和 RDCNT 有关，当 RDCNT 计数器小于等于设定的 AFVAL 时，该标志为高，否则为低，这里的 AFVAL 值为 5，所以 RDCNT 为 5 时变低；
- AFULL：近满标志和 WRCNT 有关，当 WRCNT 的计数器大于或等于设定的 AEVAL 时，该标志为高，否则为低，这里的 AFVAL 值为 4，所以 WRCNT 为 5 时变低；
- WACK：只要写信号有效的第一个时钟，该标志就为高有效，写信号无效的第一个时钟变为低无效，从图中可以对比；
- DVLD：当读操作时有效数据出现在输出总线上时，该信号就有效，图中显示在读信号有效的第二个时钟该信号变高有效。
- OVERFLOW：当 FIFO 为满的时候，如果继续写入，该信号将变高有效，这里没有多写入，所以该信号一直为低；
- UNOVERFLOW：当 FIFO 为空的时候，如果继续读数据，该信号将变高有效，这里有多读一个数据，所以可以看到有一个高电平的产生；
- RDCNT：需要注意的是，不论的读还是写，该计数器都有在动作，并且写的时候是递加的过程，落后于 WRCNT 一个周期；在读的时候是递减的过程，读信号有效后第一个有效时钟就开始递减，从图中可以对比看；
- WRCNT：和 RDCNT 差不多，在写的时候是递加的过程，写信号有效后第一个有效时钟就开始递加；在读的时候是递减的过程，落后于 RDCNT 一个周期。

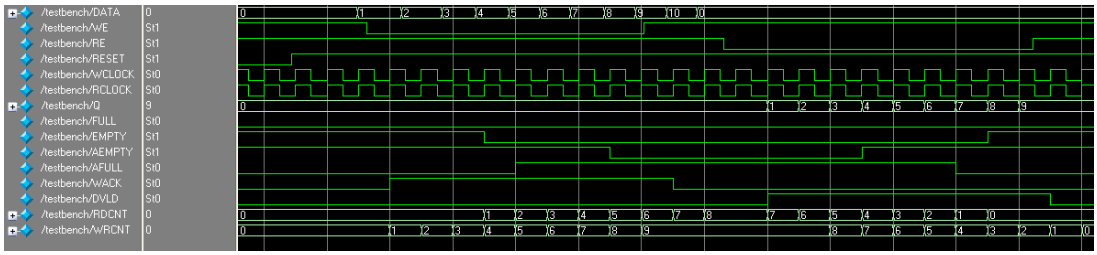


图 20　读写时钟相同的模式

回到 SmartGen 软件，将读写的时钟分开，也就是不选“single clock”，重新仿真，将得到如图 20 所示的时序，与前面有所不同的是，在写的过程中，RDCNT 落后与 WRCNT 三个周期时间，从而导致 EMPTY 和 AEMPTY 随之改变；同理，在读的时候，WRCNT 落后与 RDCNT 三个周期的时间，EMPTY 和 AEMPTY 随之改变，其他信号基本没变。

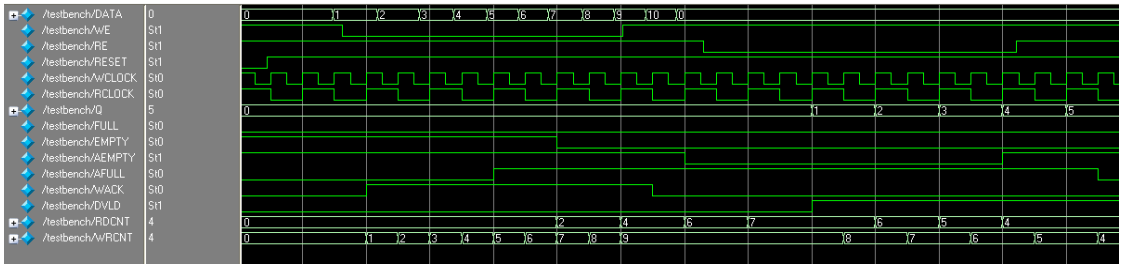


图 21　读写时钟频率不同的模式

回到 WaveFormer 软件，将读时钟改为写时钟频率的二分之一，然后保存后仿真，如图 21 所示，此时，除了在写的时候最后一个数据和写入的个数有关，其他的 RDCNT 计数都是以偶数变化， EMPTY 和 AEMPTY 也随之改变。

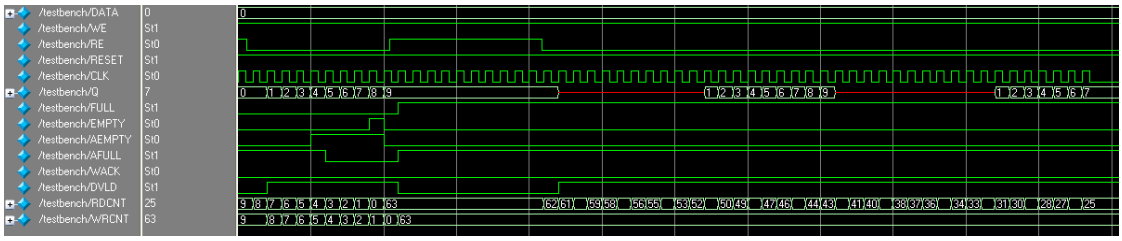


图 22　当空时允许继续读入模式

回到 SmartGen 软件的 FIFO 设置界面，选择单个时钟并将“Allow read when FIFO is empty”选中，如图 22 是仿真时序图，不同的是当 FIFO 为空的时候，能够继续进行读操作，由于 RDCNT 的位数为 6 位，所以当 RDCNT 为 0 后如果读信号有效，将从 63 递减，此时数据将重复出现，由于我们开始的时候只写入 10 个数据，所以后 10 个数据是不定态，还需要注意的是，由于 RDCNT 改变了计数的范围，EMPTY 和 AEMPTY 也改变；还有当 RDCNT 计到 0 反转的时候，WRCNT 就一直为最大值，并使 FULL 为有效禁止写入数据，只有 RDCNT 计数到 20 以内的时候，FULL 才变为低，使得可以再写入数据。

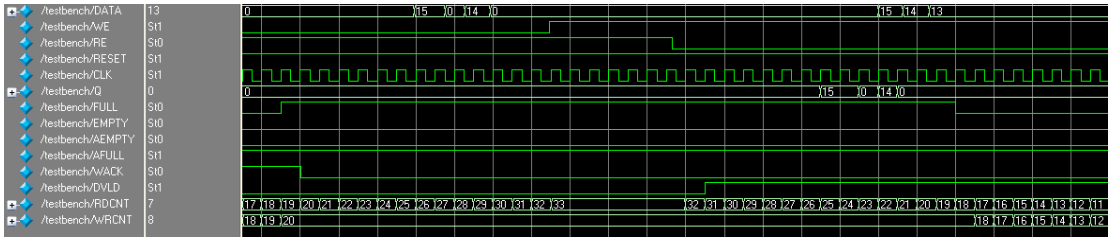


图 23　当满时允许继续写入模式

同样选择“Allow Write when FIFO is Full”时，如果 FIFO 已经满了，再写入的数据将覆盖前面的数据，从读出的数据可以看出，仿真波形如图 23 所示。

4.4 软 FIFO 控制器（不带存储单元）

(1) 打开 SmartGen 软件，然后选择 FIFO，双击图 8 中“Soft FIFO Controller without Memory”。

(2) 配置软 FIFO 控制器

如图 24 所示，是不带存储单元的软 FIFO 控制器配置界面，这些配置参数说明见表 6 所示。当设置好参数后，点击“Generate”生成。

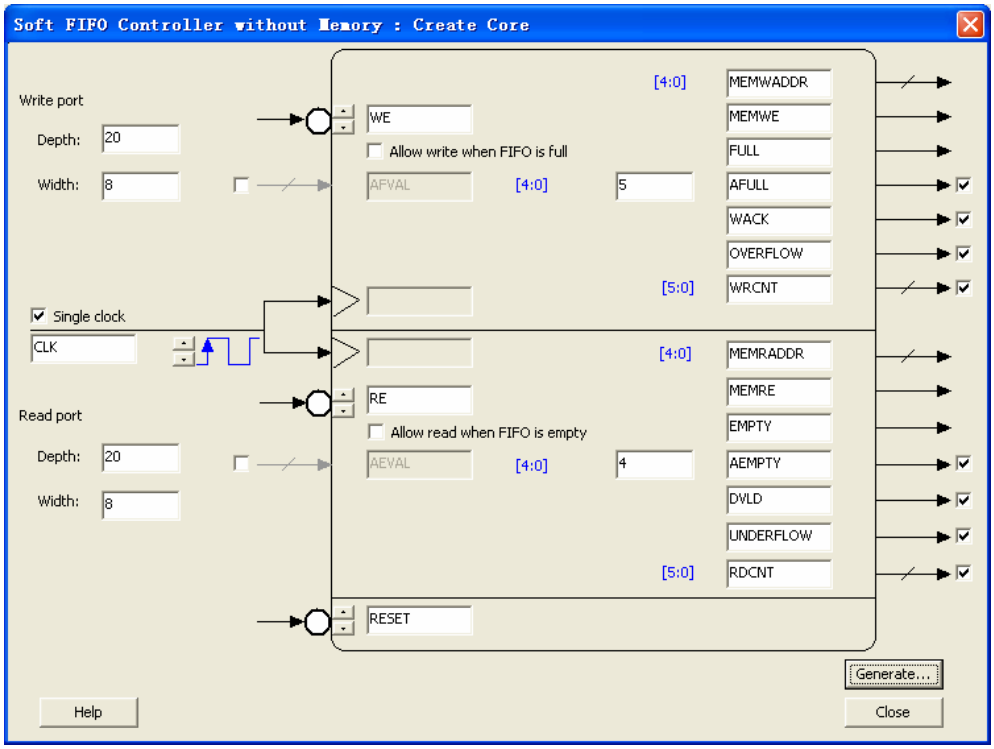


图 24 不带存储单元的 FIFO 配置界面

大部分的配置参数都和表 6 相同，只多了 MEMWADDR（外存储器写地址）、MEMWE（外存储器的写信号）、MEMRADDR（外存储器读地址）、MEMRE（外存储器的读信号）；通过这里信号可以外接用户的存储模块，非常的灵活。

(3) 回到 Libero，查看 HDL 代码

同样，FIFO 控制器是由逻辑单元搭建，所以会占用一定的资源，与带存储单元的软 FIFO 控制器相比，少了存储器单元，多了存储器的控制信号，这里只显示一部分代码。

```
001
002
003  `timescale 1 ns/100 ps
004
005  module softFIFO_2(WE, RE, WCLOCK, RCLOCK, FULL, EMPTY, RESET, AEMPTY,
006          AFULL, WACK, DVLD, OVERFLOW, UNDERFLOW, RDCNT, WRCNT, MEMWADDR,
007          MEMRADDR, MEMWE, MEMRE);
008  input WE, RE, WCLOCK, RCLOCK;
009  output  FULL, EMPTY;
010  input  RESET;
011  output  AEMPTY, AFULL, WACK, DVLD, OVERFLOW, UNDERFLOW;
012  output [4:0] RDCNT, WRCNT;
013  output [3:0] MEMWADDR, MEMRADDR;
014  output MEMWE, MEMRE;
015
016      wire WEP, REP, RBIN_0_net, RBINNXT_0_net, WBINSYNC_0_net,
017          WBIN_0_net, WBINNXT_0_net, RBINSYNC_0_net, RBIN_1_net,
018          RBINNXT_1_net, WBINSYNC_1_net, WBIN_1_net, WBINNXT_1_net,
019          RBINSYNC_1_net, RBIN_2_net, RBINNXT_2_net, WBINSYNC_2_net,
020          WBIN_2_net, WBINNXT_2_net, RBINSYNC_2_net, RBIN_3_net,
021          RBINNXT_3_net, WBINSYNC_3_net, WBIN_3_net, WBINNXT_3_net,
022          RBINSYNC_3_net, RBIN_4_net, RBINNXT_4_net, WBINSYNC_4_net,
023          WBIN_4_net, WBINNXT_4_net, RBINSYNC_4_net,
024          FULLCONSTVALUE_0_net, FULLCONSTVALUE_1_net, WDIFF_0_net,
025          WDIFF_1_net, WDIFF_2_net, WDIFF_3_net, WDIFF_4_net,
026          MEMORYWE;
027      wire AFVALCONST_0_net = FULLCONSTVALUE_1_net;
028      wire AFVALCONST_1_net = FULLCONSTVALUE_0_net;
029      wire WGRY_0_net, WGRY_1_net, WGRY_2_net, WGRY_3_net,
030          WGRY_4_net, RGRYSYNC_0_net, RGRYSYNC_1_net,
031          RGRYSYNC_2_net, RGRYSYNC_3_net, RGRYSYNC_4_net;
032      wire EMPTYVALUECONST_0_net = FULLCONSTVALUE_0_net;
033      wire RDIFF_0_net, RDIFF_1_net, RDIFF_2_net, RDIFF_3_net,
034          RDIFF_4_net, MEMORYRE;
```

图 25 HDL 代码

(4) 设置仿真测试文件

点击 WaverFormer 软件，编辑仿真测试文件，设置好波形，然后保存为 TestBench 文件，如何操作见《Libero 快速入门》文档。

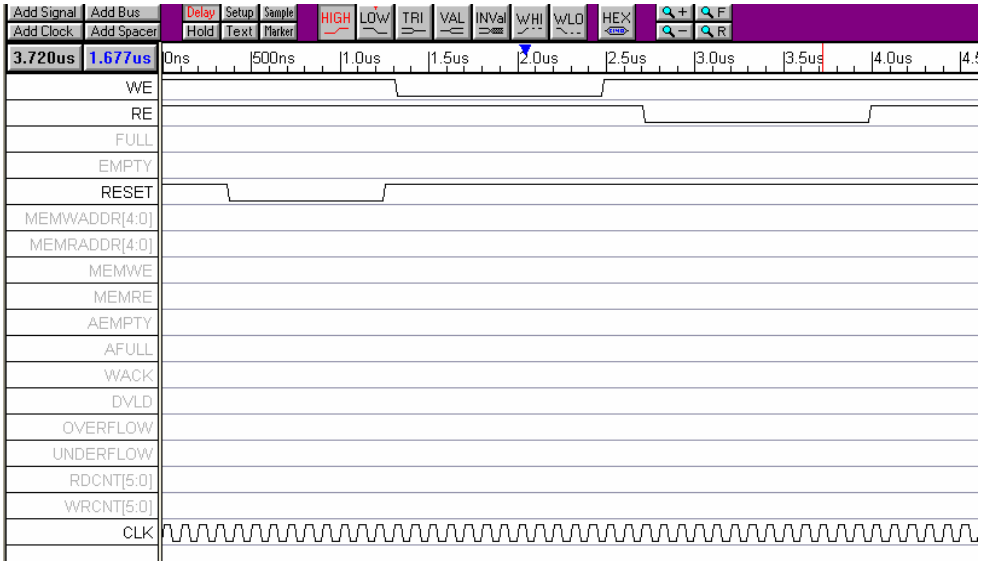


图 26 仿真测试文件生成

(5) ModelSim 仿真

这个 FIFO 控制器的时序基本上都和前面的带存储单元的 FIFO 一样，只不过多了存储器读写地址和读写信号的时序，都是和读写时钟同步的信号，这些信号可以控制外部的存储器的读和写。这里只列出一种情况的时序图，其它的可以参考前面的时序仿真图，相对比较简单，在这就不重复了，大家可以自行分析。

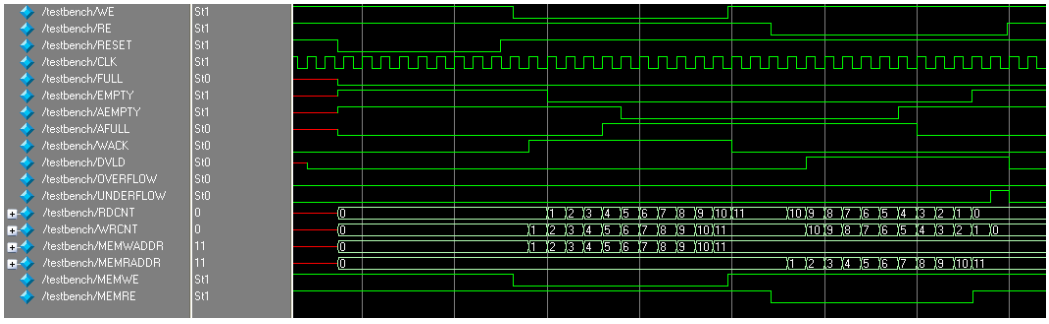


图 27 读写数据过程

5. 特别注意事项

5.1 存储模块的匹配

这点需要充分的理解，因为 FIFO 的存储器模块（4608bits）只能配置成 4,096×1，2048×2，1,024×4，512×9，256×18 五种模式，然而在 SmartGen 软件设置数据宽度和深度的时候，我们可以任意的设置（不超过最大的范围），其实这里的转换工作都交给了软件去实现转换，大致有以下几种情况：

(1) 首先根据我们的需要，在软件里设置宽度和深度，例如设置深度为 256，宽度为 8；

(2) 软件会选择最佳的匹配方式，即使用最少的 RAM 块原则。这里软件会先匹配宽度，找到大于或者等于这个宽度的模块去实现，例如：这里设置了宽度为 8，那会把 4608bits 的 RAM 存储块配置成 512×9 的模式，这样一个块的深度就被限制在 512 字节了；然后再匹配深度，这里的深度为 256 位，没有超过 512 位，那只要一个块就能实现，如果这里的深度超过了 512，那就会使用两个或者两个以上的 RAM 块来实现，其实这里有一部分的 RAM 资源被浪费了，原因就是宽度没有达到 9 位；

(3) 有些时候也会先匹配深度然后再匹配宽度，例如：如果设置的深度为 4096，宽度为 8，那首先会选择 4k×1 的模式实现 4096 的深度，然后再用 8 个相同的 RAM 块实现 8 位的宽度。

(4) 如果是读写数据宽度不同，软件会挑选读写宽度中最大的值和读写深度中最大值来匹配，然后用现有的 RAM 块实现，例如：如果写数据宽度为 2，读数据宽度为 4，深度都为 256，那会用 1k×4 的模块实现，而不是 2k×2 的模式。不同的深度也是如此规律。

5.2 FIFO 标志的匹配

(1) 12 位的 AEVAL 和 AFVAL 来指定 AEMPTY 和 AFULL 的门限值，它们和写地址计数器和读地址计数进行比较来控制 AEMPTY 和 AFULL 信号；

(2) 为了避免半字的写入或读出，FIFO 会尽快将 FULL 和 EMPTY 拉高，保持有一个字不能被写入或者读出的最低限度，例如，如果写入的是一个 2 位字而读出是 4 位，则当写入第一个 2 位字的时候，FIFO 将保持空状态，即使 FIFO 并没有实际完全变为空，因为这个情况下不能读出一个完整的 4 位字；

(3) FULL 信号变化根据不同的 FIFO 有不同的深度，如果是硬件的 FIFO，只有当 4Kbit 的全部填满的时候该信号才有效，和实际设置深度的没有关系；当使用其他两种软件的 FIFO 时，FULL 信号的变化只有达到实际设置的深度才会变化。

5.3 读写的数据宽度匹配

如果在配置的时候将读写数据宽度配置成不同的模式，那会有一个寻址有效的问题，例如：如果将写数据宽度配置成 9 位的模式，读配置成 4 位的模式，那在读操作中，只有可以读取 9 位数据中的前 8 位，第 9 位是不能访问的；相反，如果写数据宽度为 4 位，读数据宽度为 9 位，那读出的第 9 位数据是不正确的。

6. 总结

本文档主要是介绍 FIFO 一些使用情况，目的是让用户能够尽快掌握 Actel FPGA 的内部资源，列举了 FIFO 常用的使用步骤，由于编者水平有限，文档中难免有错漏的地方，敬请谅解。如果您在使用的过程中有不明白的或者发现有错漏的地方，请与本公司联系，我们将会竭尽全力为您解决问题。如果您有更好的方法或见解，也欢迎您和我们联系，共同探讨。

7. 参考资料

- Actel，PA3E_DS，pdf，2005.7
- Actel，PA3_E_SRAMFIFO_AN，pdf，2005.1

8. 免责声明

广州致远电子有限公司随附提供的软件或文档资料旨在提供给您(本公司的客户)使用，仅限于且只能在本公司制造或销售的产品上使用。

该软件或文档资料为本公司和/或其供应商所有，并受适用的版权法保护。版权所有。如有违反，将面临相关适用法律的刑事制裁，并承担违背此许可的条款和条件的民事责任。本公司保留在不通知读者的情况下，修改文档或软件相关内容的权利，对于使用中所出现的任何效果，本公司不承担任何责任。

该软件或文档资料“按现状”提供。不提供保证，无论是明示的、暗示的还是法定的保证。这些保证包括（但不限于）对出于某一特定目的应用此文档的适销性和适用性默示的保证。在任何情况下，公司不会对任何原因造成的特别的、偶然的或间接的损害负责。

公　　司：广州致远电子有限公司　测控事业部
地　　址：广州市天河区车陂路黄洲工业区七栋二楼（研发部）
邮　　编：510660
网　　址：www.embedtools.com
销售电话：+86 (020) 2887-2349
技术支持：+86 (020) 2887-2345
传　　真：+86 (020) 3860-1859

9. 销售与服务网络

广州致远电子有限公司

地址：广州市天河区车陂路黄洲工业区 3 栋 2 楼　邮编：510660
电话：(020) 22644249　28872524　22644399
　　28872342　28872349　28872569　28872573
传真：(020) 38601859
网站：http://www.embedtools.com　http://www.embedcontrol.com　http://www.ecardsys.com

广州周立功单片机发展有限公司

地址：广州市天河北路 689 号光大银行大厦 12 楼 F4 邮编：510630
电话：(020)38730972　38730976　38730916　38730917　38730977
传真：(020)38730925
网址：http://www.zlgmcu.com

广州专卖店

地址：广州市天河区新赛格电子城 203-204 室
邮编：510630
电话：(020)87578634　87578842　87569917
传真：(020)87578842
E-mail：guangzhou@zlgmcu.com

北京周立功

地址：北京市海淀区知春路 113 号银网中心 712 室
邮编：100086
电话：(010)62536178　62536179　82628073
传真：(010)82614433
E-mail：beijing@zlgmcu.com

杭州周立功

地址：杭州市登云路 428 号浙江时代电子商城 205 号
邮编：310000
电话：(0571)88009205　88009932　88009933
传真：(0571)88009204
E-mail：hangzhou@zlgmcu.com

深圳周立功

地址：深圳市深南中路 2070 号电子科技大厦 A 座 24 楼 2403 室　邮编：518031
电话：(0755)83783298　83781768　83781788
传真：(0755)83793285
E-mail：shenzhen@zlgmcu.com

上海周立功

地址：上海市北京东路 668 号科技京城东座 7E 室
邮编：200001
电话：(021)53083452　53083453　53083496
传真：(021)53083491
E-mail：shanghai@zlgmcu.com

南京周立功

地址：南京市珠江路 280 号珠江大厦 2006 室
邮编：210018
电话：(025)83613221　83613271　83603500
传真：(025)83613271
E-mail：nanjing@zlgmcu.com

重庆周立功

地址：重庆市九龙坡区石桥铺科园一路二号大西洋国际大厦(赛格电子市场) 1611 室
邮编：400039
电话：(023)68796438　68796439　68797619
E-mail：chongqing@zlgmcu.com

成都周立功

地址：成都市一环路南一段 57 号金城大厦 612 室
邮编：610041
电话：(028) 85439836　85437446
传真：(028)85437896
E-mail：chengdu@zlgmcu.com

武汉周立功

地址：武汉市洪山区广埠屯珞瑜路 158 号 12128 室（华中电脑数码市场）　邮编：430079
电话：(027)87168497　87168397　87168297
传真：(027)87163755
E-mail：wuhan@zlgmcu.com

西安办事处

地址：西安市长安北路 54 号太平洋大厦 1201 室
邮编：710061
电话：(029)87881296　87881295　83063000
传真：(029)87880865
E-mail：XAagent@zlgmcu.com